

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías



División de Tecnologías para la Integración CiberHumana

Ingeniería en Computación

Programación de Bajo Nivel

D02 - IL358 - 209850

3. Manipulación de Números

Profesor: José Juan Meza Espinoza

Alumno: Alan Yahir Juárez Rubio

Código: 218517809

Correo: alan.juarez5178@alumnos.udg.mx

Este documento contiene información sensible.
No debería ser impreso o compartido con terceras entidades.

02 de octubre de 2024



Índice

1. Introducción	4
2. Implementación	5
2.1. Código	5
2.2. Ejecución del Programa	8
3. Conclusión	9



Índice de figuras

1.	Suma de dos números	8
----	-------------------------------	---



Índice de códigos

1.	Entrada, salida y suma de números	5
----	---	---



3. Manipulación de Números

1. Introducción

La manipulación de números dentro de ensamblador es de suma importancia debido a que es uno de las bases fundamentales que practicamente cualquier programa necesita. Si bien es sencillo aplicar operaciones aritméticas, el hecho de representarlas al usuario tiene su complejidad a bajo nivel, debido a que es necesario la conversión de valores a ASCII y viceversa.

2. Implementación

2.1. Código

```
1 ; Constantes
2 SYS_EXIT    equ 1
3 SYS_READ    equ 3
4 SYS_WRITE   equ 4
5 STDIN       equ 0
6 STDOUT      equ 1
7
8 NUM_DIGITS  equ 4
9 NUM_CHARS   equ NUM_DIGITS + 1
10
11 ; Variables inicializadas
12 section .data
13     msg_1 DB "Digita el primer número: ", 0
14     len_1 equ $- msg_1
15
16     msg_2 DB "Digita el segundo número: ", 0
17     len_2 equ $- msg_2
18
19     msg_3 DB "El resultado de la suma es: ", 0
20     len_3 equ $- msg_3
21
22 ; Variables sin inicializar
23 section .bss
24     num_1_str RESB NUM_CHARS
25     num_2_str RESB NUM_CHARS
26
27     num_1 RESB 1
28     num_2 RESB 1
29
30     res_str RESB NUM_DIGITS
31     res RESB 1
32
33 section .text
34     global _start
35
36 _start:
37     ; Impresión de msg_1
38     PUSH DWORD msg_1
39     PUSH DWORD len_1
40     CALL printf
41
42     ; Ingreso de num_1_str
43     PUSH DWORD num_1_str
44     PUSH DWORD NUM_CHARS
45     CALL scanf
46
47     ; Conversión de num_1_str a valor
48     CALL ascii_to_num_1
49
50     ; Impresión de msg_2
51     PUSH DWORD msg_2
52     PUSH DWORD len_2
53     CALL printf
54
55     ; Ingreso de num_2_str
56     PUSH DWORD num_2_str
57     PUSH DWORD NUM_CHARS
```



```
58     CALL scanf
59
60     ; Conversión de num_2_str a valor
61     CALL ascii_to_num_2
62
63     ; Suma: num_1_str + num_2
64     PUSH DWORD [num_1]
65     PUSH DWORD [num_2]
66     CALL sum
67
68     ; Impresión de msg_3
69     PUSH DWORD msg_3
70     PUSH DWORD len_3
71     CALL printf
72
73     ; Convertir resultado en ascii
74     CALL num_to_ascii
75
76     ; Impresión de resultado
77     PUSH DWORD res_str
78     PUSH DWORD NUM_DIGITS
79     CALL printf
80
81 exit:
82     MOV eax, SYS_EXIT
83     MOV ebx, 0
84     INT 0x80
85
86 ; Funciones
87
88 printf:
89     PUSH ebp
90     MOV ebp, esp
91
92     MOV eax, SYS_WRITE
93     MOV ebx, STDOUT
94     MOV ecx, [ebp + 12]
95     MOV edx, [ebp + 8]
96     INT 0x80
97
98     POP ebp
99     RET
100
101 scanf:
102     PUSH ebp
103     MOV ebp, esp
104
105     MOV eax, SYS_READ
106     MOV ebx, STDIN
107     MOV ecx, [ebp + 12]
108     MOV edx, [ebp + 8]
109     INT 0x80
110
111     MOV BYTE [ecx + 3], 0
112
113     POP ebp
114     RET
115
116 ascii_to_num_1:
117     centenas_1:
118         MOV al, [num_1_str]
```



```
119         SUB al, '0'
120
121         MOV cl, 100
122         MUL cl
123         ADD bl, al
124
125     decenas_1:
126         MOV al, [num_1_str + 1]
127         SUB al, '0'
128
129         MOV cl, 10
130         MUL cl
131         ADD bl, al
132
133     unidades_1:
134         MOV al, [num_1_str + 2]
135         SUB al, '0'
136         ADD bl, al
137
138     MOV [num_1], bl
139
140     RET
141
142 ascii_to_num_2:
143     centenas_2:
144         MOV al, [num_2_str]
145         SUB al, '0'
146
147         MOV cl, 100
148         MUL cl
149         ADD bl, al
150
151     decenas_2:
152         MOV al, [num_2_str + 1]
153         SUB al, '0'
154
155         MOV cl, 10
156         MUL cl
157         ADD bl, al
158
159     unidades_2:
160         MOV al, [num_2_str + 2]
161         SUB al, '0'
162         ADD bl, al
163
164     MOV [num_2], bl
165
166     RET
167
168 num_to_ascii:
169     MOV eax, [res]
170
171     MOV BYTE [res_str], '0'
172
173     centenas_3:
174         MOV ebx, 100
175         MOV edx, 0
176         DIV ebx
177         ADD eax, '0'
178         MOV [res_str + 1], eax
179
```



```
180     decenas_3:
181         MOV eax, edx
182         MOV ebx, 10
183         MOV edx, 0
184         DIV ebx
185         ADD eax, '0'
186         MOV [res_str + 2], eax
187
188     unidades_3:
189         ADD edx, '0'
190         MOV [res_str + 3], edx
191
192     RET
193
194 sum:
195     PUSH ebp
196     MOV ebp, esp
197
198     MOV al, [ebp + 12]
199     MOV bl, [ebp + 8]
200
201     ADD bl, al
202     MOV [res], bl
203
204     POP ebp
205     RET
```

Código 1: Entrada, salida y suma de números

2.2. Ejecución del Programa

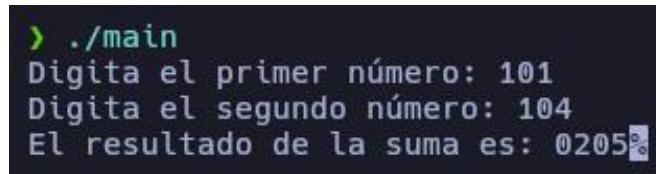


Fig. 1: Suma de dos números



3. Conclusión

Para finalizar, me es importante mencionar que si bien esta actividad parece ser un tanto sencilla, la realidad es que el hecho de trabajar en bajo nivel es lo que lo convierte en laborioso. Cuestiones como conversión de valor a ASCII, almacenamiento de múltiples dígitos, entre otras, fueron un desafío que sin duda me ayudó a comprender muchas cosas, especialmente la manipulación e interpretación de datos numéricos a través de texto.