Shrikrushna S Zirape
21286 - (1)

PICT, PUNE

Assignment-01

DOP:- 16/02/2021

Problem Statement:-

Write an x86/64 ALP to accept a 64 bit hexadecimal number from user and store them in an array and display the accepted numbers.

SW/HW Requirements:-

intel is 8th gen 64-bit processor
OS:- Linux 64 bit OS
Editor:- VS Code / Gedit.
Assembler:- Nasm
Debugger:- GDB/TD

Theory:-

1) System calls:-

i) Write System Call:-

```
mov rax, 01
mov rdi, 01
mov rsi, msg1
mov rdx, 0BH
syscall
```

ii) Read System Call

```
mov rax, 00
mov rdi, 01
mov rsi, rbx
mov rdx, 17
syscall.
```

PICT, PUNE

iii) Exit System Call :-

```
mov rax,60
mov rdi,00
Syscall.
```

2) Instructions used

i) mov = move memory into resister
ii) add = add two value and store them into resistor.
iii) dec = decrement the byte [count] variable.
iv) jnz = If value is not zero jump to the location.

Algorithm :-

1 Start.
2 declare msg1 for input. in .data section
3 declare msg2 for output in .data section.
4 declare cnt variable as counter in .bss.
5 declare array of 85 bytes in .bss
6 // input of array
6 Write system call to print msg1
7 move 05 to counter
8 move base location of arry to rsi.
9 declare a breakpoint.
10 Read system call for (input)
11 add 17 to rbx and decrement counter.
12 Jnz to step 9 if counter is not equal to 0

// output of array.

13 Write system call for msg 2
14 declare a breakpoint.
15 Follow 7, 8 again.
16 Write system call to display nos
17 add 17 to rbx
18 decrease Counter.
19 jump to 13 if Counter is non zero.
20 Exit system call.

Memory Dig :-

Quadword : 8 byte

0123456789ABCDEF

	0000	0001	← 2000 location in memory
	0010	0011	01
	0100	0101	02
	0110	0111	03
Quadword →	1000	1010	04
(64-bit no)	1010	1011	05
stored in memory.	1100	1101	06
	1110	1111	07

PICT, PUNE

21286 → (4)

While accepting nos from user ascii values of input digit are stored.

eg:- No is ABCDEF0123456789.

9	0011	1001	39
8	0011	1000	38
7	0011	0111	37
6	0011	0110	36
5	0011	0101	35
4	0011	0100	34
3	0011	0011	33
2	0011	0010	32
1	0011	0001	31
0	0011	0000	30
F	0100	0110	46
E	0100	0101	45
D	0100	0100	44
C	0100	0011	43
B	0100	0010	42
A	0100	0001	41

Hence we have to secure 80 bytes for 6 quad words and not 40 byte we have reserved 6 extra bytes for end of line character.

PICT, PUNE

21286 → (5)

Conclusion:-

Hense He are successfully able to create an array and store 6 quad word in it we used loop for redding and printing data.