

**Project Report**

**Project Name: Digital Clock**

**COURSE TITLE: Microprocessors & Microcontrollers Lab COURSE CODE: 304**

**Date of performance : 14 – 10 – 2020 Date of submission : 27 – 10 – 2020**

|  |  |  |
| --- | --- | --- |
| Submitted To  **ABDULLAH AL-ARIF**  Lecturer  Department of CSE |  | Submitted By  **Name: Parvej Hossain**  ID: 181002085  Section: 181\_DC  Department of CSE |

**Introduction:**

A digital clock is a type of clock that displays the time digitally, i.e. in numerals or other symbols, as opposed to an analog clock, where the time is indicated by the positions of rotating hands. Digital Clock in assembly language is developed as an educational project to learn assembly language in more detail and have practical experience.

**Features:**

1. The clock shows the time of the computer system.
2. It shows hours, minutes.
3. It is displayed in a classy, stylish and attractive font.
4. The interface is simplistic with white text on black background.

**How to use:**

* Download TASM for Windows or emu8086 and install.
* Build the program and run it.

**Code:**

; Parvej Hossain

; 8086 Assembly, emu8086

data segment

zero db " 000000000 ", 10

db " 00:::::::::00 ", 10

db " 00:::::::::::::00 ", 10

db "0:::::::000:::::::0", 10

db "0::::::0 0::::::0", 10

db "0:::::0 0:::::0", 10

db "0:::::0 0:::::0", 10

db "0:::::0 000 0:::::0", 10

db "0:::::0 000 0:::::0", 10

db "0:::::0 0:::::0", 10

db "0:::::0 0:::::0", 10

db "0::::::0 0::::::0", 10

db "0:::::::000:::::::0", 10

db " 00:::::::::::::00 ", 10

db " 00:::::::::00 ", 10

db " 000000000 ", 10, "$"

one db " 1111111 ", 10

db " 1::::::1 ", 10

db "1:::::::1 ", 10

db "111:::::1 ", 10

db " 1::::1 ", 10

db " 1::::1 ", 10

db " 1::::1 ", 10

db " 1::::l ", 10

db " 1::::l ", 10

db " 1::::l ", 10

db " 1::::l ", 10

db " 1::::l ", 10

db "111::::::111", 10

db "1::::::::::1", 10

db "1::::::::::1", 10

db "111111111111", 10, "$"

two db " 222222222222222 ", 10

db "2:::::::::::::::22 ", 10

db "2::::::222222:::::2 ", 10

db "2222222 2:::::2 ", 10

db " 2:::::2 ", 10

db " 2:::::2 ", 10

db " 2222::::2 ", 10

db " 22222::::::22 ", 10

db " 22::::::::222 ", 10

db " 2:::::22222 ", 10

db "2:::::2 ", 10

db "2:::::2 ", 10

db "2:::::2 222222", 10

db "2::::::2222222:::::2", 10

db "2::::::::::::::::::2", 10

db "22222222222222222222", 10, "$"

three db " 333333333333333 ", 10

db "3:::::::::::::::33 ", 10

db "3::::::33333::::::3", 10

db "3333333 3:::::3", 10

db " 3:::::3", 10

db " 3:::::3", 10

db " 33333333:::::3 ", 10

db " 3:::::::::::3 ", 10

db " 33333333:::::3 ", 10

db " 3:::::3", 10

db " 3:::::3", 10

db " 3:::::3", 10

db "3333333 3:::::3", 10

db "3::::::33333::::::3", 10

db "3:::::::::::::::33 ", 10

db " 333333333333333 ", 10, "$"

four db " 444444444 ", 10

db " 4::::::::4 ", 10

db " 4:::::::::4 ", 10

db " 4::::44::::4 ", 10

db " 4::::4 4::::4 ", 10

db " 4::::4 4::::4 ", 10

db " 4::::4 4::::4 ", 10

db "4::::444444::::444", 10

db "4::::::::::::::::4", 10

db "4444444444:::::444", 10

db " 4::::4 ", 10

db " 4::::4 ", 10

db " 4::::4 ", 10

db " 44::::::44", 10

db " 4::::::::4", 10

db " 4444444444", 10, "$"

five db "555555555555555555 ", 10

db "5::::::::::::::::5 ", 10

db "5::::::::::::::::5 ", 10

db "5:::::555555555555 ", 10

db "5:::::5 ", 10

db "5:::::5 ", 10

db "5:::::5555555555 ", 10

db "5:::::::::::::::5 ", 10

db "555555555555:::::5 ", 10

db " 5:::::5", 10

db " 5:::::5", 10

db "5555555 5:::::5", 10

db "5::::::55555::::::5", 10

db " 55:::::::::::::55 ", 10

db " 55:::::::::55 ", 10

db " 555555555 ", 10, "$"

six db " 66666666 ", 10

db " 6::::::6 ", 10

db " 6::::::6 ", 10

db " 6::::::6 ", 10

db " 6::::::6 ", 10

db " 6::::::6 ", 10

db " 6::::::6 ", 10

db " 6::::::::66666 ", 10

db "6::::::::::::::66 ", 10

db "6::::::66666:::::6 ", 10

db "6:::::6 6:::::6", 10

db "6:::::6 6:::::6", 10

db "6::::::66666::::::6", 10

db " 66:::::::::::::66 ", 10

db " 66:::::::::66 ", 10

db " 666666666 ", 10, "$"

seven db "77777777777777777777", 10

db "7::::::::::::::::::7", 10

db "7::::::::::::::::::7", 10

db "777777777777:::::::7", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db " 7::::::7 ", 10

db "77777777 ", 10, "$"

eight db " 888888888 ", 10

db " 88:::::::::88 ", 10

db " 88:::::::::::::88 ", 10

db "8::::::88888::::::8", 10

db "8:::::8 8:::::8", 10

db "8:::::8 8:::::8", 10

db " 8:::::88888:::::8 ", 10

db " 8:::::::::::::8 ", 10

db " 8:::::88888:::::8 ", 10

db "8:::::8 8:::::8", 10

db "8:::::8 8:::::8", 10

db "8:::::8 8:::::8", 10

db "8::::::88888::::::8", 10

db " 88:::::::::::::88 ", 10

db " 88:::::::::88 ", 10

db " 888888888 ", 10, "$"

nine db " 999999999 ", 10

db " 99:::::::::99 ", 10

db " 99:::::::::::::99 ", 10

db "9::::::99999::::::9", 10

db "9:::::9 9:::::9", 10

db "9:::::9 9:::::9", 10

db " 9:::::99999::::::9", 10

db " 99::::::::::::::9", 10

db " 99999::::::::9 ", 10

db " 9::::::9 ", 10

db " 9::::::9 ", 10

db " 9::::::9 ", 10

db " 9::::::9 ", 10

db " 9::::::9 ", 10

db " 9::::::9 ", 10

db " 99999999 ", 10, "$"

line db 0

column db 0

page\_number db 0

digit\_unit db 0

digit\_ten db 0

time db 0

minute db 0

second db 0

current\_minute db 0

current\_second db 0

digit\_pointer dw 10 dup(?)

ends

stack segment

dw 128 dup(?)

ends

extra segment

ends

code segment

start:

; set segment registers:

mov ax, data

mov ds, ax

mov ax, extra

mov es, ax

call set\_digit\_pointer

main\_loop:

call load\_time

mov al, current\_second

cmp second, al

jne do\_print

mov al, current\_minute

cmp minute, al

jne do\_print

jmp main\_loop

do\_print:

mov al, current\_minute

mov minute, al

mov al, current\_second

mov second, al

call clear\_screen

call print

jmp main\_loop

jmp fim

print:

; print minute -----

mov al, current\_minute

mov time, al

call parse\_time

; hour ten

mov al, digit\_ten

call set\_digit

mov column, 0

call print\_digit

; hour unit

mov al, digit\_unit

call set\_digit

mov column, 20

call print\_digit

; print second -----

mov al, current\_second

mov time, al

call parse\_time

; minute ten

mov al, digit\_ten

call set\_digit

mov column, 40

call print\_digit

; minute unit

mov al, digit\_unit

call set\_digit

mov column, 60

call print\_digit

ret

clear\_screen: ; get and set video mode

mov ah, 0fh

int 10h

mov ah, 0

int 10h

ret

load\_time: ; save CH = hour, CL = minute, DH = second

mov ah, 2Ch

int 21h

mov current\_minute, cl

mov current\_second, dh

ret

set\_digit\_pointer:

mov digit\_pointer[0], offset zero

mov digit\_pointer[2], offset one

mov digit\_pointer[4], offset two

mov digit\_pointer[6], offset three

mov digit\_pointer[8], offset four

mov digit\_pointer[10], offset five

mov digit\_pointer[12], offset six

mov digit\_pointer[14], offset seven

mov digit\_pointer[16], offset eight

mov digit\_pointer[18], offset nine

ret

set\_digit: ; set digit from al to si

mov bl, 2

mul bl

mov si, ax

mov si, digit\_pointer[si]

ret

parse\_time: ;parse time in "time", loading "digit\_ten" and "digit\_unit"

mov ah, 0

mov al, time

mov bl, 10

div bl

mov digit\_ten, al

mov digit\_unit, ah

ret

print\_digit: ;print digit in SI until find "$", set line = 4 and column = column

mov line, 4

call set\_cursor

print\_main:

mov dh, 0

mov dl, ds:[si]

cmp dx, "$"

je end\_print

cmp dx, 10

je new\_line

mov ah, 2

int 21h

inc si

jmp print\_main

new\_line:

inc line

call set\_cursor

inc si

jmp print\_main

end\_print:

ret

set\_cursor:

mov ah, 2

mov bh, page\_number

mov dh, line

mov dl, column

int 10h

ret

fim:

mov ax, 4c00h ; exit to operating system.

int 21h

code ends

end start

Result:



**REFERENCE:**

[1] wineyard.in/wp-content/uploads/2012/12/WYARD9.pdf

[2] <http://www.pelnet.co.uk/elect/projects/clock/Clock_a.pdf>

[3] http://en.wikipedia.org/wiki/Digital\_clock

[4] [www.ee.duke.edu/~jmorizio/ece261/F09/projects/clk.pdf](http://www.ee.duke.edu/~jmorizio/ece261/F09/projects/clk.pdf)

[5] <https://www.hni.uni-paderborn.de/uploads/media/doc2486_01.pdf>

[6] datasheets.maximintegrated.com/en/ds/DS1307.pdf

**Discussion:**

In this Project, I learn about various Micro-Processor & Micro-Controller things which is very useful to knowing Assembly language. This Projects is very informative.