FACULTAD DE INGENIERÍA DEL EJÉRCITO

TRABAJO PRÁCTICO NRO 1

técnicas digitales II

CUESTIONARIO

ct HORACIO FUENTES

2021

# Implemente el problema 2 (Dividir un número alojado en R1 por 8) en una subrutina. Analice el comportamiento del microcontrolador cuando se produce el llamado a la subrutina

Diagrama de bloques:

Implementación del problema en Assembler

;----------------------CÓDIGO ASSEMBLER -------------------

; Fecha:21May21

;

; Programador: HORACIO FUENTES

;

; Descripción: Dividir un número alojado en R1 por 8.

;

; Modificaciones: Ninguna

;

;----------------------------------------------------------

;------------------Archivos Include------------------------

$bitdef.h

$ioAT89C52.h

;----------------------------------------------------------

;----------------- DECLARACIÓN DEL PROGRAMA ---------------

NAME template

PUBLIC main

;----------------------------------------------------------

;---- Prototipos de ISRs ----------------------------------

PUBLIC myISR

;----------------------------------------------------------

;---- Prototipos de subrutinas ----------------------------

PUBLIC MiSubrutina

;----------------------------------------------------------

;---- Constantes ------------------------------------------

MI\_CONSTANTE EQU 4

CONSTANTE\_BIN EQU b'00001000

CONSTANTE\_HEX EQU 0x45

;Constantes para los segmentos

SEG\_CODIGO EQU 0030h

SEG\_STACK EQU 00C0h

LONG\_STACK EQU 10

;----------------------------------------------------------

;---SEGMENTO DE CODIGO-------------------------------------

ASEG

org 0000h

ljmp main

ASEG

org 000BH

ljmp myISR

;---SEGMENTO DE STACK--------------------------------------

ASEG

org SEG\_STACK

stack\_init DS LONG\_STACK

;---SEGMENTO DE DATOS -------------------------------------

RSEG IDATA\_Z:DATA:NOROOT

my\_variable ds 1

Tabla2Pow db 1,2,4,8,16,32,64

;---SEGMENTO DE CÓDIGO-------------------------------------

RSEG RCODE

ORG SEG\_CODIGO

main

mov SP,#stack\_init-1

Loop\_ppal:

CALL MiSubrutina

NOP

salir: jmp Loop\_ppal

;--------FIN MAIN -----------------------------------------

;-------Subrutinas-----------------------------------------

RSEG NEAR\_CODE:CODE:NOROOT(0)

;===========SUBRUTINAS=====================================

;----------------------------------------------------------

; Nombre: MiSubrutina

; Parametros: ninguno

; Devuelve: nada

; Modifica: nada

; Nivel de Stack:

; Descripción:

; Dividide un número alojado en R1 por 8.

;----------------------------------------------------------

MiSubrutina:

MOV R1, #16

MOV B,#8

MOV A, R1

DIV AB

MOV R0, A

ret

;===ISRs===================================================

;----------------------------------------------------------

; Nombre: myISR

; Parametros:ninguno

; Devuelve:nada

; Modifica:

; xxxxx

; Nivel de Stack:

;

; Descripción:

; Rutina de servicio de interrupción del <periférico x>

;----------------------------------------------------------

myISR:

reti

;--------FIN MAIN -----------------------------------------

END main

Análisis del comportamiento:

# Aplicar la formula f(x)=3x+7 a las 10 posiciones de RAM interna a partir de la 30h y colocar el resultado a partir de la 40h (recuerde que el resultado será en dos bytes).

Diagrama de bloques:

Implementación del problema en Assembler:

;----------------------CÓDIGO ASSEMBLER -------------------

; Fecha:21May21

;

; Programador: HORACIO FUENTES

;

; Descripción: Aplica f(x)=3x+7 a las 10 posiciones de RAM interna a partir de la 30h y colocar el resultado a partir de la 40h;

; Modificaciones: Ninguna

;

;----------------------------------------------------------

;------------------Archivos Include------------------------

$bitdef.h

$ioAT89C52.h

;----------------------------------------------------------

;----------------- DECLARACIÓN DEL PROGRAMA ---------------

NAME template

PUBLIC main

;----------------------------------------------------------

;---- Prototipos de ISRs ----------------------------------

PUBLIC myISR

;----------------------------------------------------------

;---- Prototipos de subrutinas ----------------------------

PUBLIC MiSubrutina

;----------------------------------------------------------

;---- Constantes ------------------------------------------

MI\_CONSTANTE EQU 4

CONSTANTE\_BIN EQU b'00001000

CONSTANTE\_HEX EQU 0x45

;Constantes para los segmentos

SEG\_CODIGO EQU 0030h

SEG\_STACK EQU 00C0h

LONG\_STACK EQU 10

;----------------------------------------------------------

;---SEGMENTO DE CODIGO-------------------------------------

ASEG

org 0000h

ljmp main

ASEG

org 000BH

ljmp myISR

;---SEGMENTO DE STACK--------------------------------------

ASEG

org SEG\_STACK

stack\_init DS LONG\_STACK

;---SEGMENTO DE DATOS -------------------------------------

RSEG IDATA\_Z:DATA:NOROOT

my\_variable ds 1

Tabla2Pow db 1,2,4,8,16,32,64

;---SEGMENTO DE CÓDIGO-------------------------------------

RSEG RCODE

ORG SEG\_CODIGO

main

mov SP,#stack\_init-1

Loop\_ppal:

;Mueve valores a la memoria para comprobación

MOV 030H, #255

MOV 031H, #2

MOV 032H, #253

MOV 033H, #252

MOV 034H, #251

MOV 035H, #250

MOV 036H, #249

MOV 037H, #248

MOV 038H, #247

MOV 039H, #246

;Mueve a R0 y R1 las direcciones de memoria de donde toma los datos y donde almacena los resultados

MOV R0, #030H

MOV R1, #040H

MOV R3, #010 ;Asigna el número de iteraciones a realizar

;Asignación de los valores de la función lineal

MOV R4, #3 ;Asigna el valor de la pendiente de la función

MOV R5, #7 ;Asigna el valor de la ordenada al origen de la función

FUNC: MOV A, @R0 ;Mueve al acumulador el contenido de la posición de memoria a donde apunta R0

INC R0

MOV B, R4 ;Mueve a B el valor de la pendiente

MUL AB

ADD A,R5 ;Suma el valor de la ordenada al origen

INC R1

MOV @R1, A ;Almacena la parte baja del resultado una posición de memoria por delante de la posición inicial

DEC R1

MOV A,B

ADDC A,#0H ;En caso de que el carry haya quedado en 1, incrementa el valor de la parte alta

MOV @R1, A ;Mueve a la posición de memoria inicial la parte alta del resultado

INC R1

INC R1

DJNZ R3,FUNC ;Produce la interación según la cantidad de oportunidades determinadas en R3

NOP

salir: jmp Loop\_ppal

;--------FIN MAIN -----------------------------------------

;-------Subrutinas-----------------------------------------

RSEG NEAR\_CODE:CODE:NOROOT(0)

;===========SUBRUTINAS=====================================

;----------------------------------------------------------

; Nombre: MiSubrutina

; Parametros: ninguno

; Devuelve: nada

; Modifica: nada

; Nivel de Stack:

; Descripción:

;----------------------------------------------------------

MiSubrutina:

ret

;===ISRs===================================================

;----------------------------------------------------------

; Nombre: myISR

; Parametros:ninguno

; Devuelve:nada

; Modifica:

; xxxxx

; Nivel de Stack:

;

; Descripción:

; Rutina de servicio de interrupción del <periférico x>

;----------------------------------------------------------

myISR:

reti

;--------FIN MAIN -----------------------------------------

END main

# CONCLUSIONES