

**(6609) LABORATORIO DE MICROCOMPUTADORAS**

Proyecto:  
*( tp1 prender apagar un led)*

<b>Profesor:</b>	<b>Ing. Jorge A. Alberto</b>
<b>Cuatrimestre / Año:</b>	1ro/2020
<b>Turno de clases prácticas:</b>	Miércoles
<b>Jefe de Trabajos Prácticos:</b>	Pedro Martos
<b>Docente guía:</b>	

Autores			Seguimiento del proyecto									
Nombre	Apellido	Padrón										
Cristian	Simonelli	87879										

**Observaciones:**

---

---

---

---

---

---

---

---

---

---

<b>Fecha de aprobación</b>

<b>Firma J.T.P.</b>

<b>COLOQUIO</b>	
<b>Nota final</b>	
<b>Firma Profesor</b>	

<b>Objetivo:</b>	<b>2</b>
<b>Desarrollo.</b>	<b>2</b>
Instrucciones:	2
DEC:	2
BRNE:	3
<b>Listado de componentes:</b>	<b>4</b>
<b>Diagrama de flujos: (solo de la función delay)</b>	<b>5</b>
<b>Código:</b>	<b>6</b>
<b>Diagrama en bloque:</b>	<b>7</b>
<b>Resultado:</b>	<b>8</b>
<b>Conclusiones:</b>	<b>8</b>

## Objetivo:

El objetivo de este primer trabajo práctico es encender y apagar un led con el microcontrolador. Para ello controlar puertos de entrada y salida que el mismo posee. Utilizando un arduino uno, la placa ya tiene un led conectado al pin 13 (bit 5 del puerto b).

Por lo tanto es necesario poner ese pin en estado alto y bajo para prender y apagar el led.

## Desarrollo.

El microcontrolador opera a 8Mhz, cada instrucción requieren de determinados ciclos de cómputo. Como queremos hacer una función de delay exacta debemos tener en cuenta los ciclos de cada instrucción.

Utilizaremos (si bien no es la mejor manera y es solo como ejemplo de primer trabajo práctico) una rutina de retardo, que lo que hace es decrementar registros y comparar los mismos con 0. Una determinada cantidad de veces.

Nota, no hace falta usar cpi ya que dec setea el flag z.

## Instrucciones:

### DEC:

(i)  $Rd \leftarrow Rd - 1$

Syntax:

Operands:

Program Counter:

(i) **DEC** Rd

$0 \leq d \leq 31$

$PC \leftarrow PC + 1$

16-bit Opcode:

1001	010d	dddd	1010
------	------	------	------

1 ciclo..

## BRNE:

- (i) If  $R_d \neq R_r$  ( $Z = 0$ ) then  $PC \leftarrow PC + k + 1$ , else  $PC \leftarrow PC + 1$

Syntax:

Operands:

Program Counter:

- (i) BRNE k

$-64 \leq k \leq +6$

$PC \leftarrow PC + k + 1$

$PC \leftarrow PC + 1$ , if condition is false

16-bit Opcode:

1111	01kk	kkkk	k001
------	------	------	------

### Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
—	—	—	—	—	—	—	—

Example:

```
eor r27,r27 ; Clear r27
loop: inc r27 ; Increase r27
...
cpi r27,5 ; Compare r27 to 5
brne loop ; Branch if r27<>5
nop ; Loop exit (do nothing)
```

**Words** 1 (2 bytes)

**Cycles** 1 if condition is false

2 if condition is true

Como se ve en las figuras anteriores, las operaciones necesitan 1 y 2 ciclos respectivamente.

Como el microcontrolador funciona a 8Mh.

$8\text{Mhz}/3\text{clicos} = 2.666.666$

Cada registro es de 1 byte, y por lo tanto tiene 256 valores posibles. Como se decrementa y luego se pregunta son 255 valores.

$2.666.666/255 = 10.457$

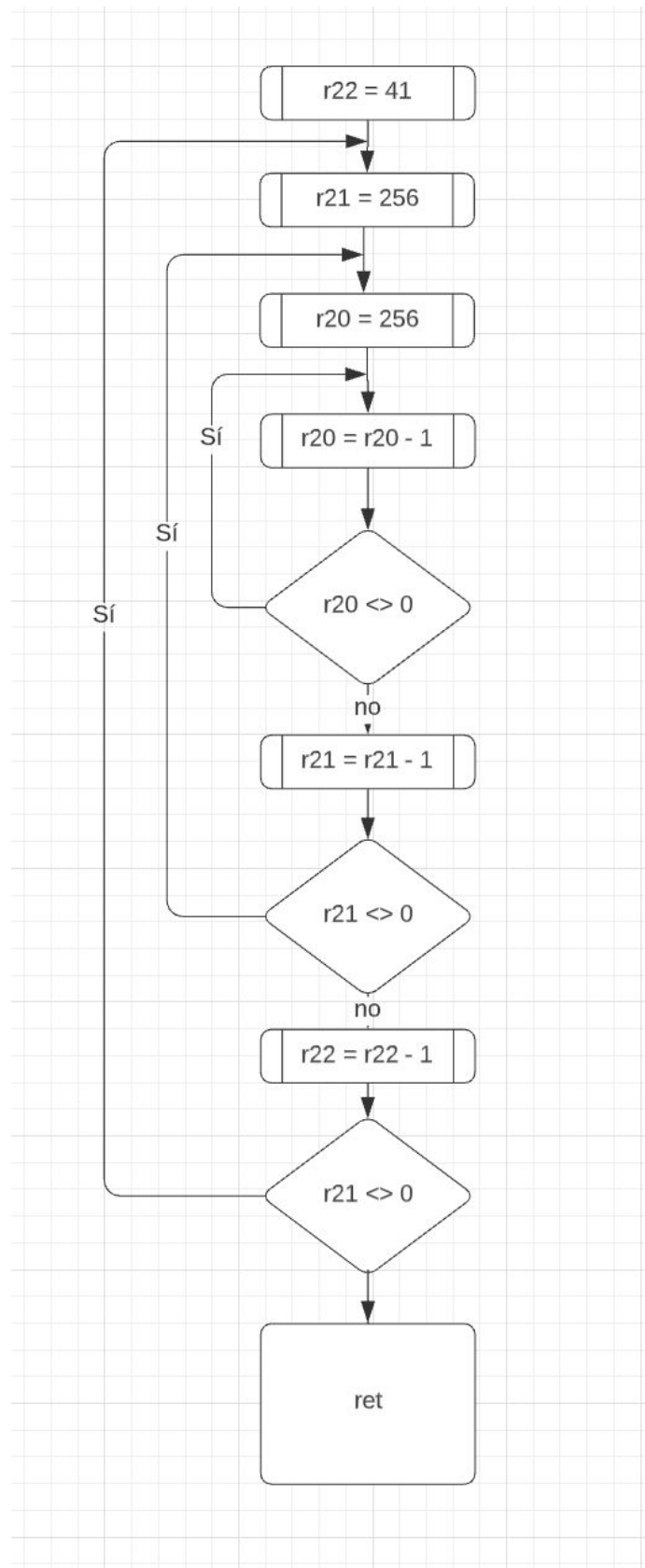
$10.457/255 = 41$

O sea se necesitan 3 registros.

## **Listado de componentes:**

Placa arduino UNO

## Diagrama de flujos: (solo de la función delay)



## Código:

<https://github.com/csimonelli/labo-micro/> (es privado)

```
.include "m328pdef.inc" ; Valid definitions to 238p

.equ pin_lead = 5 ; the built in led is the pin 13
(5th pin in B port)

.org 0x000 ; The next instruction has to be
written to add 0x0000

        rjmp      main ; Relative jump to main
.org INT_VECTORS_SIZE ; inter vector

main:

        ldi        r20, HIGH(RAMEND) ; Load r20 with the last ram address
higher byte
        out        sph, r20 ; Load higher byte in sp with r20
        ldi        r20, LOW(RAMEND) ; Load r20 with the last ram address
lower byte
        out        spl, r20 ; Load lower byte in sp with r20

        ldi        r20, 0xff ; B port as output
        out        DDRB, r20 ; B port as output

led:     cbi        PORTB, pin_lead ; Led loop turn off led
        call       delay ; Delay
        sbi        PORTB, pin_lead ; Turn on led
        call       delay ; Delay
        jmp        led ; Loop 4 ever

delay: ; Delay procedure

        push       r20 ; Save the r20 value in the stack
        push       r21 ; Save the r21 value in the stack
        push       r22 ; Save the r22 value in the stack
        ldi        r22, 41 ; 41 * 255 * 255 aprox 8000000/3
loop1:   ldi        r21, 255 ;
loop2:   ldi        r20, 255 ;
loop3:   dec        r20 ; decrement r20 by 1
        brne       loop3 ; If r20 had reached 0, z flag would
have been seted ; and we will jump to loop 3

        dec        r21 ; The same as above
        brne       loop2
        dec        r22
        brne       loop1
        pop        r22 ; Set r22, r21, r20 to the same value
that
```

```

pop      r21                ; it had before entere this proc
pop      r20
ret      ; go back to main

```

## Diagrama en bloque:

Figure 2-1. Block Diagram

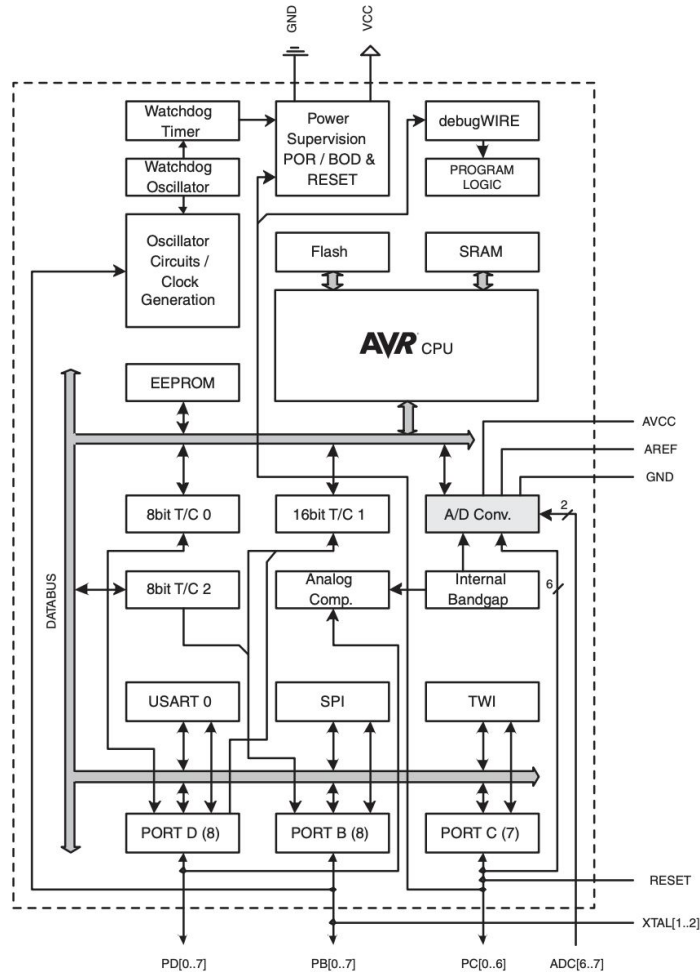


Diagrama en bloque probablemente no aplique para este tp. Lo importante de aquí y a los efectos de este primer tp es el puerto b pin 5, la memoria de programa y la memoria ram.



## **Resultado:**

Se logró controlar el encendido del led en un tiempo controlado.

## **Conclusiones:**

Se entiende que es primer tp y la idea es la puesta a punto del entorno de desarrollo.  
Hace una rutina de retardo decrementando registros seguramente no sea la mejor manera de hacer lo que se requiere, básicamente por consumo energético.