Universidad de Buenos Aires
Facultad de Ingeniería

86.07 - Laboratorio de microprocesadores

TP Integrador

Berard, Lucia Magdalena
101213 - lberard@fi.uba.ar

1er cuatrimestre de 2021

En el siguiente trabajo práctico consiste en estudiar la carga y descarga en condensadores. Se utilizan conceptos de PWM, timers, conversores analógicos-digitales y comunicación con el puerto serie.

Índice

1. Introducción

2. Desarrollo

2.1. Conexión y configuración de los puertos

Se utilizó un capacitor de 10uF y una resistencia de 220 Ω para armar el circuito RC y se conectó de la siguiente forma:

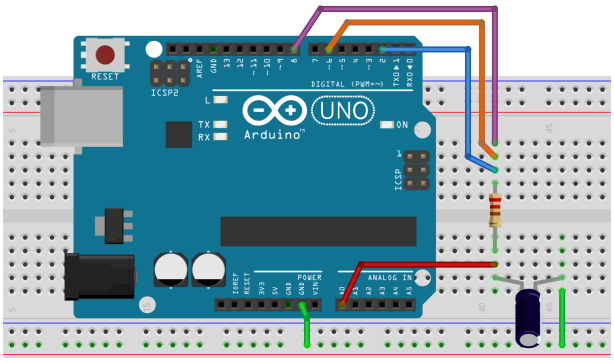

Figura 1: Diagrama esquemático

2.2. Programa

2.2.1. Generar con FAST PWM una señal de 50 Hz con un DC de 50 %

La frecuencia mínima a la que se puede llegar con la señal es de aproximadamente 61 Hz dado que la frecuencia de trabajo del Arduino UNO es de 16 MHz, el prescaler máximo del Timer es 1024, y que el Timer puede contar desde 0 hasta 255.

Se generó un clock de 15625 y para lograr una señal de 50Hz habria que contar hasta aproximadamente 312, el problema es que solo se puede contar hasta 255. Por eso la mínima es 61 Hz.

Para generar esta frecuencia se configuraron los registrs TCCR0A y TCCR0B, se inicializo en cero el timer y se configuro el duty cycle de la siguiente manera:

```
configPWM:
        ;Fast PWM
        ldi             R16, (1<<COM0A1) | (1<<WGM00) | (1<<WGM01) | (1<<WGM02) ;10000011
        out             TCCR0A, R16
        ;Prescaler en 1024
        ldi             R16, (1<<CS02) | (1<<CS00) ;00000101
        out             TCCR0B, R16
        ;Timer en cero
        clr             R16
        out             TCNT0, R16
        ;Duty Cicle del 50%
        ldi             R16, 126
        out             OCR0A, R16
    ret
```

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

2.2.2. Verificar la frecuencia generada con el modo de captura del timer 1

Se configuró el timer 1 en modo normal, con prescaler 1024 y flanco ascendente y la interrupcion del timer 1 por captura configurando el registro TIMSK1 :

```
configTimer1
        ldi             R16,((1<<ICES1) | (1<<CS10) | (1<<CS12))
        sts             TCCR1B, R16
    ret

configINT_Timer1
        ldi             R16,(1<<ICIE1)
        sts             TIMSK1, R16
    ret
```

En cuanto al resto de las interrupciones se las configuró de la siguiente manera:

```
habilitarINTRecep:
;Modifico UCSR0B para habilitar las interrupciones por recepcion
        lds             R16, UCSR0B
        sbr             R16, (1<<RXCIE0) ;0b10000000
        sts             UCSR0B, R16
    ret

configINT0:
        ldi             R16, 0 ;(1<<ISC00)
        sts             EICRA, R16
        ;Habilito interrupcion
        ldi             R16, (1<<INT0) ; o INTF0?
        out             EIMSK, R16
        ;Borro flags
        ldi             R16, (1<<INTF0)
        out             EIFR, R16
    ret
```

2.2.3. Armar un circuito RC y Adquirir con el ADC valores de voltaje sobre el capacitor , VC .

En el PIN de salida de la señal PWM se conectó una resistencia en serie con un capacitor para hacer un circuito RC, y entre ellos se conectó el conversor AD, como se puede observar en la Figura 1.

Como se trabaja con una frecuencia de 61 Hz, un semiperíodo de la señal PWM sera de aproximadamente 8.2 ms, por lo que se eligieron valores de resistencia y capacitancia de tal forma que la constante τ entre en el semiperiodo y se pueda observar la carga y descarga del capacitor. Los valores elegidos fueron 220 Ω y 10 uF.

$$\tau = RC = 2,2ms$$

por lo que 5τ estara en el orden de los 11ms.

Para configurar el ADC se modificaron los registros ADMUX y ADCSRA.

```
configADC:
        ;Registro de salida a la izquierda y configuro la entrada ADC0
        ldi             R16, 0b01100000         ;(1<<ADLAR) | (1<<REFS0)
        sts             ADMUX, R16
    ret

initADC:
        ;Habilito el ADC, el inicio de la conversion y un prescaler de 128 para el clock interno
        ldi             R16, 0b11100111;(1<<ADEN) | (1<<ADSC) | (1<<ADPS2) | (1<<ADPS1)| (1<<ADPS0)
        sts             ADCSRA, R16
    ret
```

ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

2.2.4. Transmitir por vía serial los valores obtenidos a una PC

```
configSerialPort:
        ;Habilito transmision y recepcion
        ldi             R16, (1<<TXEN0) | (1<<RXEN0) ;0b00001000
        sts             UCSR0B, R16

        ;Asincronico, sin paridad y 8 bits de datos
        ldi             R16, (1<<UCSZ00) | (1<<UCSZ01) ;0b00000110
        sts             UCSR0C, R16

        ;Configuro Baud rate
        clr             R16
        sts             UBRR0H, R16
        ldi             R16, BPS
        sts             UBRR0L, R16
    ret
```

2.3. Diagramas

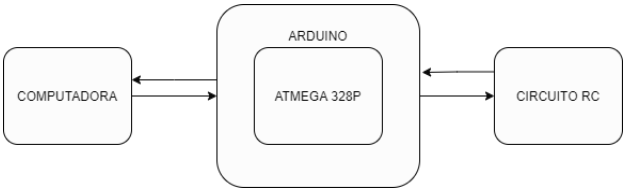

Figura 2: Diagrama en bloques

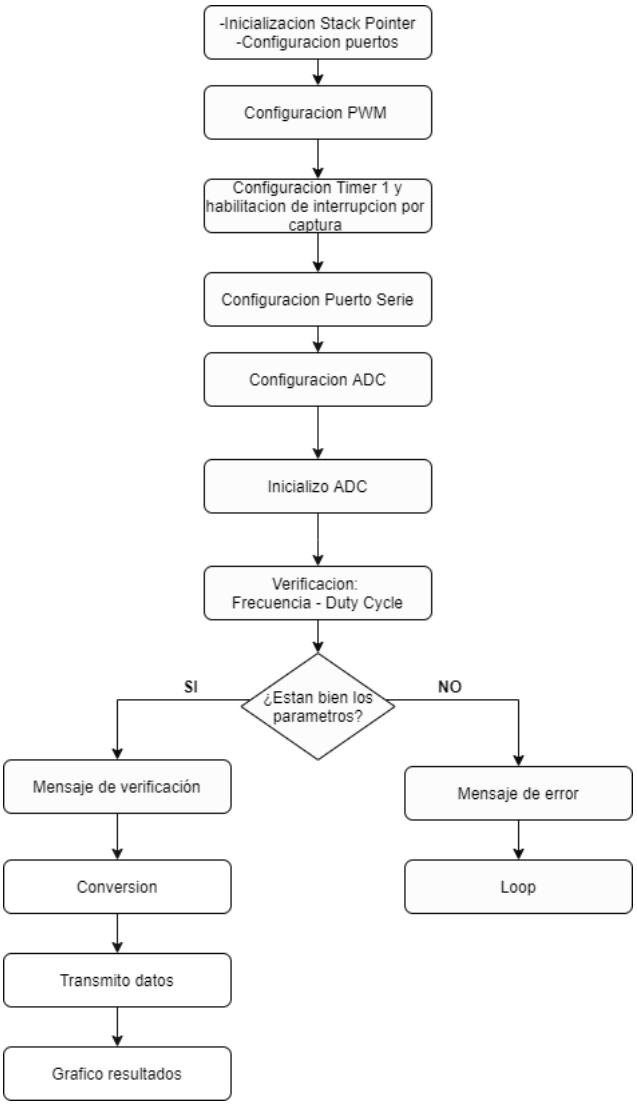

Figura 3: Diagrama de flujo

3. Resultados

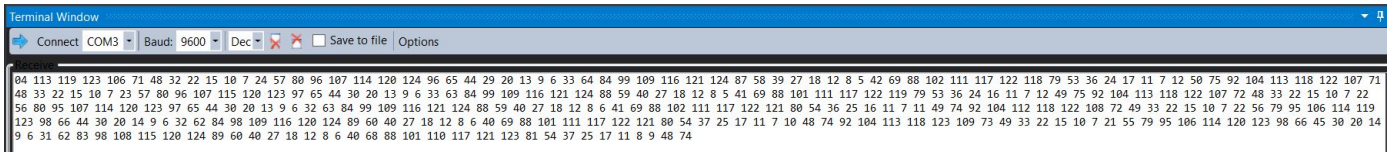

Figura 4: Mensaje serial: Transmision de los datos

Se utilizó el lenguaje de programación Python para graficar los valores obtenidos:

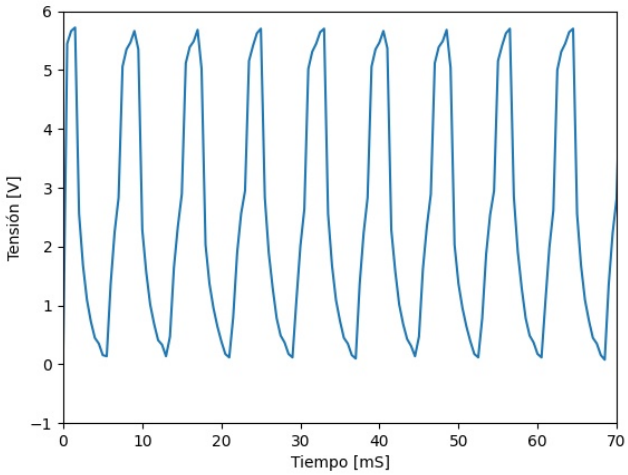

Figura 5: Carga y descarga del capacitor

4. Conclusiones

Si bien se logró obtener y analizar la curva de carga y descarga del capacitor, se tuvo problemas en la verificación de la frecuencia y el duty cycle generados. Para verificarla la señal generada se conectó el pin de salida del PWM al pin de INT0 para poder tener flancos consecutivos y mediante el modo de captura del Timer1 poder detectarlo de la siguiente manera:

- Primer captura: se guarda el contador del Timer1
- Segunda captura: se compara el contador del Timer1 con el de la primera captura para verificar Duty Cycle
- Tercera captura: se compara nuevamente para verificar el período de la señal y por consecuencia su frecuencia.

En comparación a los trabajos prácticos anteriores, no se utilizaron macros en este trabajo ya que se consideró que no era necesario. Las macros se las podria considerar como una "metainstruccion de assembler", es decir algo que se asemeja a una instruccion a la cual se le pueden ingresar parametros pero en realidad es un conjunto de instrucciones. En cambio las rutinas serían como "funciones"que logran la legibilidad del codigo y permiten la iteración y repetición de instrucciones. De esa forma el programa ocupa menos memoria.

5. Anexo

5.1. Código principal

```
.equ FCPU = 16*10^6 ;frecuencia de oscilacion 16MHz
.equ BAUD = 9600
.equ BPS = 103 ;((FCPU/16/BAUD) - 1)
.def count = R17

.include "m328Pdef.inc"

;Interrupciones---------
.org 0x0016
        rjmp transmitir
;-----------------------

.org 0x0000
        rjmp config
.org INT_VECTORS_SIZE
;---------------------------------------------------------------------
;                       Configuracion de los puertos, timer e interrupciones
;---------------------------------------------------------------------

config:
```

```
21          ;Inicializo el Stack Pointer
22          rcall initSP
23
24          ;PORTB pin 0: (puerto 8 arduino) ICP1
25          sbi                 DDRB, 0
26          ;PORTD pin 6: (puerto 6 arduino) PWM
27          sbi                 DDRD, 6
28          ;PORTC pin 0: (puerto A0 arduino) Entrada analogica ADC
29          cbi                 DDRC, 0
30          ;PORTD pin 2: (puerto 2 arduino) Entrada INT0
31          cbi                 DDRD, 2
32          ;PORTB pin 1: Activa int por captura
33          ;sbi PORTB, 1
34
35          ;PWM: Fast PWM, Prescaler en 1024, Duty Cicle del 50%
36          rcall configPWM
37
38          ;Timer1 Modo normal, prescaler 1024, flanco ascendente
39          rcall configTimer1
40
41          ;Timer1 interrupcion por captura
42          ;rcall configINT_Timer1
43
44          ;Puerto serie: BaudRate, Velocidad normal, habilito transmision y recepcion.
45          ;Asincronico, sin paridad y 8 bits de datos
46          rcall configSerialPort
47
48          ;ADC: Registro de salida a la izquierda y configuro la entrada ADC0
49          rcall configADC
50
51          ;ADC: Habilito ADC, inicio, prescaler de 128 y deshabilito el buffer de la entrada
52          rcall initADC
53
54          ;Interrupcion: Cualquier cambio en el PIN activa la interrupción, habilito y borro flags
55          ;rcall configINT0
56
57          ;Activo interrupciones globales
58          sei
59  ;-----------------------------------------------------------------------
60  ;                                                           Programa
61  ;-----------------------------------------------------------------------
62  transmitir:
63          lds                 R16, UCSR0A
64          sbrs        R16, UDRE0
65          rjmp        transmitir
66
67          lds                 R16, ADCH
68          sts                 UDR0, R16
69          jmp                 transmitir
70          sei
71  ;-----------------------------------------------------------------------
72  ;                                                           Rutinas
73  ;-----------------------------------------------------------------------
74  ;Inicializacion del Stack Pointer--------------------------------
75  initSP:
76          ldi                 R16, HIGH(RAMEND)
77          out                 SPH, R16
78          ldi                 R16, LOW(RAMEND)
79          out                 SPL, R16
80          ret
81
82  ;PWM-------------------------------------------------------------
83  configPWM:
84          ;Fast PWM
85          ldi                 R16, (1<<COM0A1) | (1<<WGM00) | (1<<WGM01) | (1<<WGM02) ;10000011
86          out                 TCCR0A, R16
87          ;Prescaler en 1024
88          ldi                 R16, (1<<CS02) | (1<<CS00) ;00000101
89          out                 TCCR0B, R16
90          ;Timer en cero
91          clr                 R16
92          out                 TCNT0, R16
93          ;Duty Cicle del 50%
94          ldi                 R16, 126
95          out                 OCR0A, R16
96          ret
```

```
97
98  ;Configuracion Timer-------------------------------------------
99  configTimer1:
100         ldi             R16, (1<<ICES1) | (1<<CS10) | (1<<CS12) ;0b01000101
101         sts             TCCR1B, R16
102         ret
103
104 configINT_Timer1:
105         ldi             R16,(1<<ICIE1) ;0b00100000
106         sts             TIMSK1, R16
107         ret
108 ;Puerto serie-----------------------------------------------------
109 configSerialPort:
110
111         ;Habilito transmision y recepcion
112         ldi             R16, (1<<TXEN0) | (1<<RXEN0) ;0b00001000
113         sts             UCSR0B, R16
114
115         ;Asincronico, sin paridad y 8 bits de datos
116         ldi             R16, (1<<UCSZ00) | (1<<UCSZ01) ;0b00000110
117         sts             UCSR0C, R16
118
119         ;Configuro Baud rate
120         clr             R16
121         sts             UBRR0H, R16
122         ldi             R16, BPS
123         sts             UBRR0L, R16
124         ret
125
126 habilitarINTRecep:
127 ;Modifico UCSR0B para habilitar las interrupciones por recepcion
128         lds             R16, UCSR0B
129         sbr             R16, (1<<RXCIE0) ;0b10000000
130         sts             UCSR0B, R16
131         ret
132
133 ;ADC-------------------------------------------------------------
134 configADC:
135         ;Registro de salida a la izquierda y configuro la entrada ADC0
136         ldi             R16, 0b01100000         ;(1<<ADLAR) | (1<<REFS0)
137         sts             ADMUX, R16
138         ret
139
140 initADC:
141         ;Habilito el ADC, el inicio de la conversion y un prescaler de 128 para el clock interno
142         ldi             R16, 0b11100111;(1<<ADEN) | (1<<ADSC) | (1<<ADPS2) | (1<<ADPS1)| (1<<ADPS0)
143         sts             ADCSRA, R16
144         ret
145
146 ;Interrupcion---------------------------------------------------
147 configINT0:
148         ldi             R16, (1<<ISC00)
149         sts             EICRA, R16
150         ;Habilito interrupcion
151         ldi             R16, (1<<INT0) ; o INTF0?
152         out             EIMSK, R16
153         ;Borro flags
154         ldi             R16, (1<<INTF0)
155         out             EIFR, R16
156         ret
157 ;-----------------------------------------------------------------------
158 ;                                                       Mensajes
159 ;-----------------------------------------------------------------------
160 /*
161 Mensaje_error:
162         .db "Error en Frecuencia o Duty Cycle", 0
163 Mensaje_verificacion:
164         .db "Frecuencia y Duty Cycle correctos", 0
165 */
```

5.2. Repositorio Github

`https://github.com/fiuba-labo-de-micro-miercoles/2021_1c_trabajos_practicos-lmberard/tree/master/TP_INTEGRADOR`

5.3. Bibliografía

- AVR Datasheet
- AVR Manual de instrucciones
- Esquemático Arduino UNO

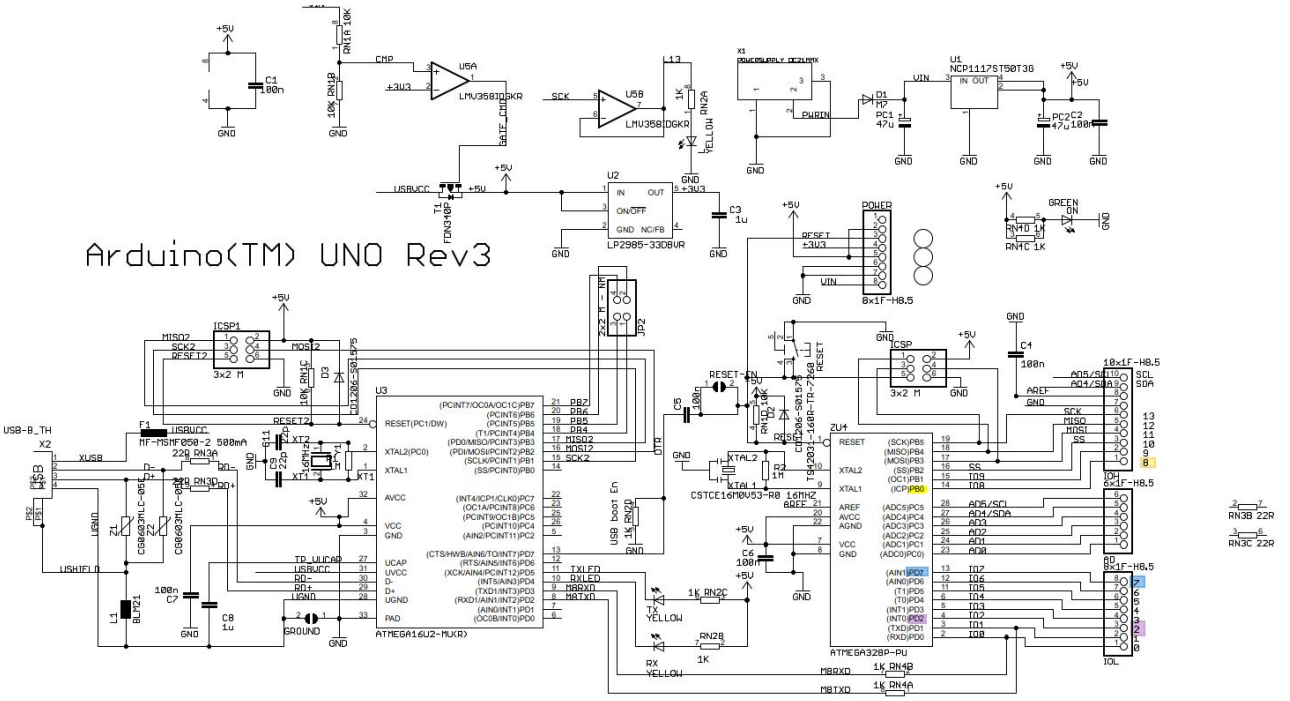

Figura 6: Esquemático Arduino UNO