Universidad de Buenos Aires
Facultad De Ingeniería
Año 2021 – 1.er Cuatrimestre

(82.07) Laboratorio de microprocesadores

Entrega N.º 6: Integrador
Curso 01

Docentes a cargo:
Stola, Gerardo Luis
Salaya, Juan Guido
Cofman, Fernando

Integrante	Padrón	Correo electrónico
Lützelschwab, Nahila	100686	nlützelschwab@fi.uba.ar

Índice

1. Objetivos del proyecto **3**

2. Descripción del proyecto **3**
2.1. Lista de componentes 3

3. Esquemático **3**

4. Software **4**
4.0.1. Fast PWM 4
4.0.2. Verificación de la frecuencia 5
4.0.3. Cálculo de la resistencia y capacidad 5
4.0.4. Comunicación puerto serie 5
4.0.5. Conversor A/D 5
4.1. Diagrama de flujo 5

5. Resultados **9**

6. Conclusiones **10**

7. Anexo **10**
7.1. Código fuente 10

8. Bibliografía **17**

1. Objetivos del proyecto

El objetivo del presente trabajo práctico es el estudio de la carga y descarga de los condensadores, integrando algunas de las herramientas adquiridas a lo largo del curso.

2. Descripción del proyecto

El presente trabajo práctico consistió en diseñar un programa que permita muestrear la tensión proveniente de la carga y descarga de un condensador conectado a un canal conversor analógico digital del microcontrolador ATMega328p. Para ello fue necesario generar una señal de entrada de 50Hz con un duty cycle de 50 % con el modo Fast PWM; y verificar la frecuencia generada con el modo de captura del Timer1 mediante el uso de la interrupción externa Int0. A partir de ello se graficó el comportamiento de los valores digitalizados de la salida del circuito RC.

2.1. Lista de componentes

A continuación se listan los componentes utilizados para la implementación del proyecto.

- Placa Arduino UNO
- Protoboard de 830 puntos
- Cables macho-macho para protoboard
- Dos resistencias de valor 180Ω
- Una resistencias de valor 150Ω
- Un capacitor de $100\mu F$

3. Esquemático

Como muestra el esquemático 1, se hizo uso de una placa Arduino UNO basada en el microcontrolador ATmega328p, a la cual se conectó un circuito RC de manera tal que sea alimentado por la señal generada de 50Hz con un duty cycle de 50 % a taves del pin PD6, quedando de esta manera el capacitor como salida del circuito. A su vez, se utilizaron como entradas los pines PD2(INT0), PB0(ICP) y PC0(ADC0) como se muestra en el esquematico.

En el diagrama de bloques 2 se representa la conexión necesaria para poder realizar la comunicación serie entre el circuito RC, el microcontrolador y la PC (de la cual se observan los datos).

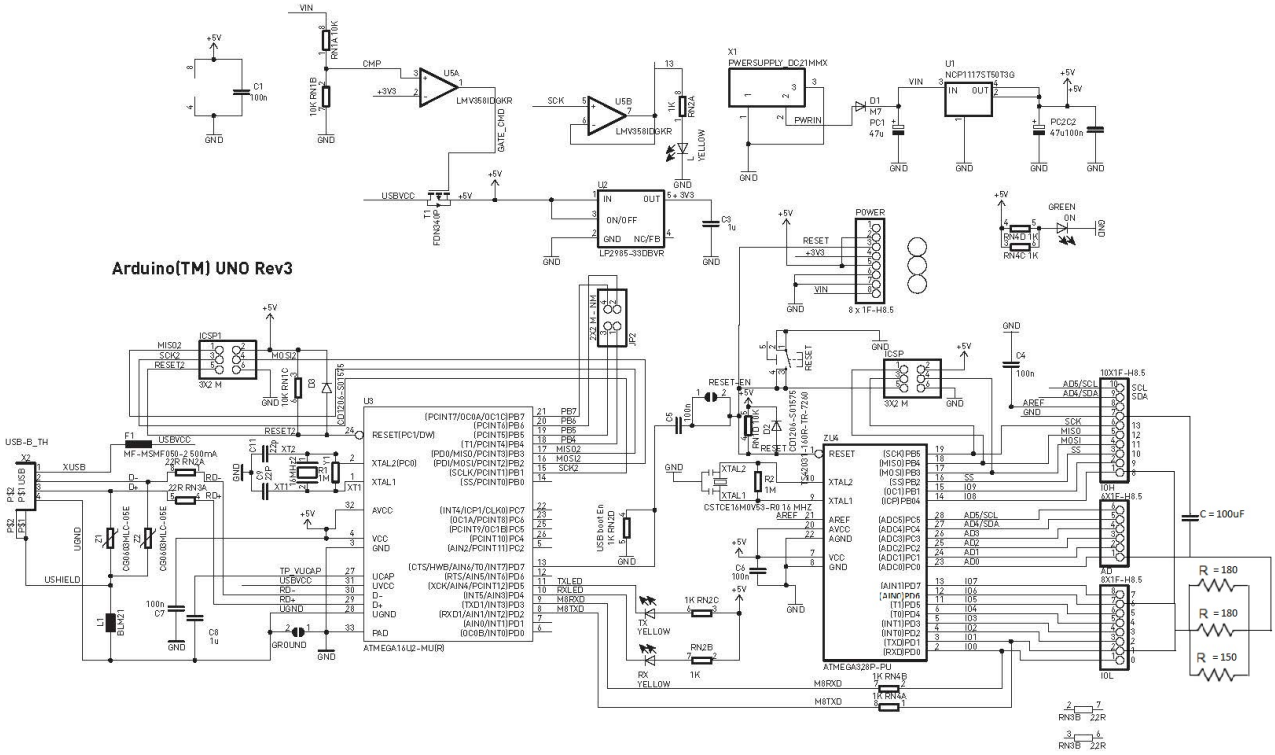

Figura 1: Esquematico completo del circuito implementado

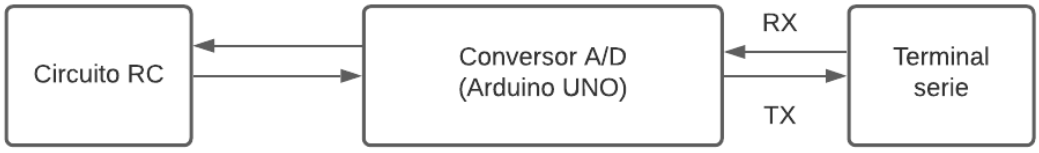

Figura 2: Diagrama en bloques

4. Software

Mediante el software Microchip Studio, se desarrolló el programa a implementar en código Assembly.

Para llevar a cabo la implementación del algoritmo se tuvieron en cuenta las siguientes funcionalidades:

- Generar la señal cuadrada de $50Hz$ con un duty-cycle de 50 % haciendo uso de la funcionalidad de Timer del microcontrolador.
- Verificar los la frecuencia generada con el modo de captura del Timer1 usando la interrupción Int0.
- Calcular los valores de resistencia y capacidad para una lograr un τ que sea 3 veces menor el periodo de la señal de entrada, es decir, $\tau < \frac{T}{3}$.
- Adquirir con el conversor A/D valores de voltaje sobre el capacitor V_C.
- Transmitir por comunicación serie USART los valores obtenidos a una PC.

4.0.1. Fast PWM

Para generar la señal cuadrada de $50Hz$ con un duty-cycle de 50 % a partir del modo PWM del Timer0 de 8-bits del microcontrolador, se calculó el prescaler necesario considerando que el ATMega328p contiene un

oscilador de cristal de 16MHz:

$$T = \frac{2^8}{16MHz} \cdot prescaler = \frac{1}{50Hz} \Rightarrow prescaler = 1250$$

Dado que los valores de prescaler ya estan determinados, se tomó el valor que más se aproxima, tomando de esta manera $prescaler = 1024$. Para lograr un duty-cycle de 50 % se configuró el Timer0 en modo comparador seteando el registro comparador de salida (OCR0A) a 127.

4.0.2. Verificación de la frecuencia

Para poder verificar la frecuencia y el duty-cycle de la señal generada por el Timer0 se hizo uso de la funcionalidad de captura del Timer1. Es decir, se configuró el Timer1 en modo captura con un prescaler de 1024 al igual que para el modo Fast PWM,y se habilitó la interrupcion de captura ICP1(PB0). A su vez, se configuró INT0 para transferir los datos mediante puerto serie, de manera que cualquier cambio produzca la interrupción.

Se implementó un algoritmo que verifique la frecuencia a partir de la detección de flancos de la señal cuadrada utilizando las interrupciones. Cuando se activa la interrupción externa se impone una interrupción de captura ICP1 por flanco ascendiente, realizando la verificación de las características de la señal. Esto lo realiza guardando los parte alta y baja del ICR1 cada vez que se produce una interrupción y llevando la cuenta de la cantidad de flancos detectados. Para calcular el período se utilizan los valores del primer y tercer flanco; y para calcular el duty-cycle el primer y segundo flanco.

4.0.3. Cálculo de la resistencia y capacidad

Para calcular los valores de resistencia y capacidad que logren un $\tau = R \cdot C$ que sea tres veces menor que el período de la señal de entrada, se fijó un valor de capacidad de $100\mu F$ y se calculó la resistencia necesaria.

$$\tau < \frac{T}{3} \Rightarrow R \cdot C < \frac{2 \cdot 10^{-2}}{3} = 6,66 \cdot 10^{-3}$$

$$R = 66,6\Omega$$

Para la implementación del circuito RC físico se utilizaron dos resistencias de 180Ω y una de 150Ω. Quedando de esta manera un $\tau \approx 5,6ms$, y una carga del capacitor máxima de 99 % luego de un tiempo $5\tau = 28ms$. En la práctica el capacitor no logra cargarse y descargarse por completo, y que pasado el régimen transitorio, el sistema se estabiliza alrededor de un valor medio de $2,5V$ devido a que los semiciclos de la señal generada por PWM toman los valores $0V$ y $5V$.

4.0.4. Comunicación puerto serie

Los valores de tensión sobre el capacitor V_C que se obtienen mediante el conversor A/D, son transmitidos por comunicación serie de 8-bits de datos, sin bit de paridad y un bit de stop. Para ello se tomó un boud rate de 76800, UBRR = 127.

4.0.5. Conversor A/D

Para la conversión de la señal analógica del PWM a digital, se configuró la mínima frecuencia de conversión posible de $125kHz$ y se seteó AVCC con capacitor externo, que ajuste el resultado hacia la izquierda y con única entrada ADC0. A su vez, fue necesario configurar el Timer2 en modo normal con un prescaler de 32 y habilitando la interrupción por desbordamiento. Por lo tanto, cuando se activa la interrupción lee la entrada ADC y se transmiten los datos por el puerto serie. Una vez que se generaron las 64 muestras se apaga el Timer2. Para que los datos sean transmitidos en forma correcta, hay que tener en cuenta que el tiempo de transmisión sea menor al de muestreo.

4.1. Diagrama de flujo

En la figura 6 se puede observar el diagrama de flujo correspondiente al software principal.

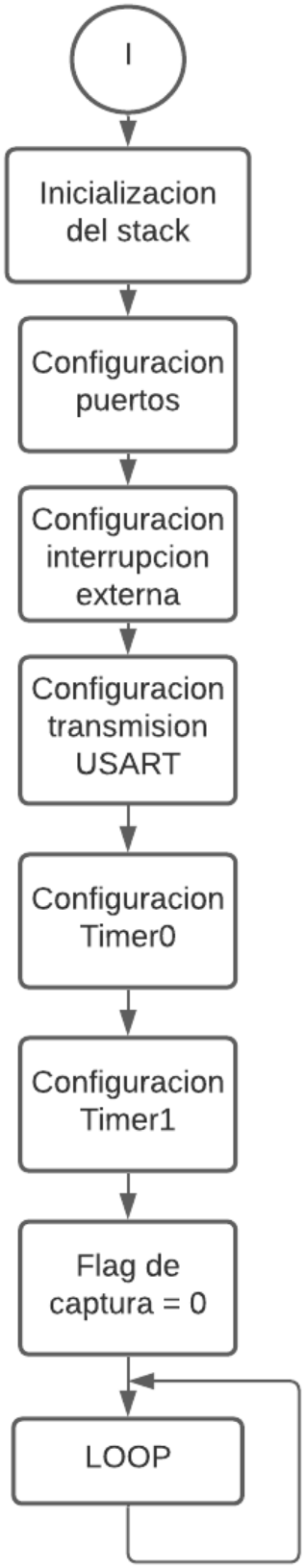

Figura 3: Diagrama de flujo completo

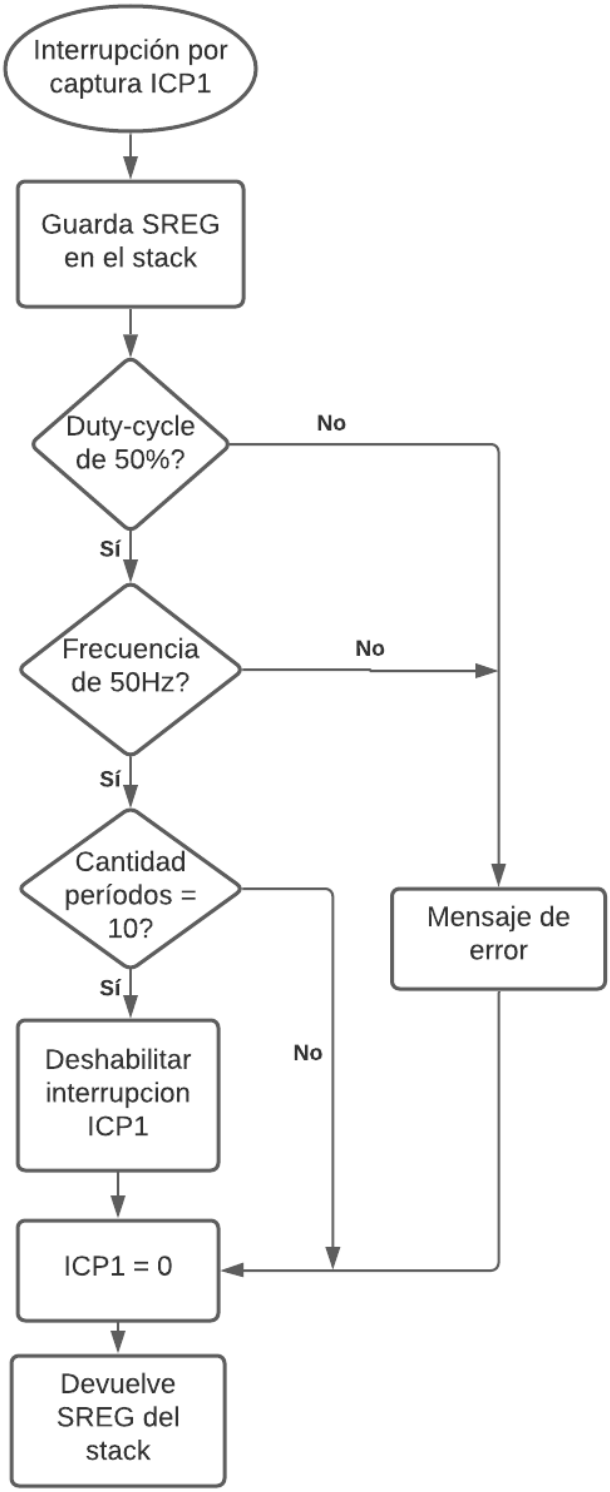

Figura 4: Diagrama de flujo de la interrupcion por captura

Figura 5: Diagrama de flujo de interrupcion externa

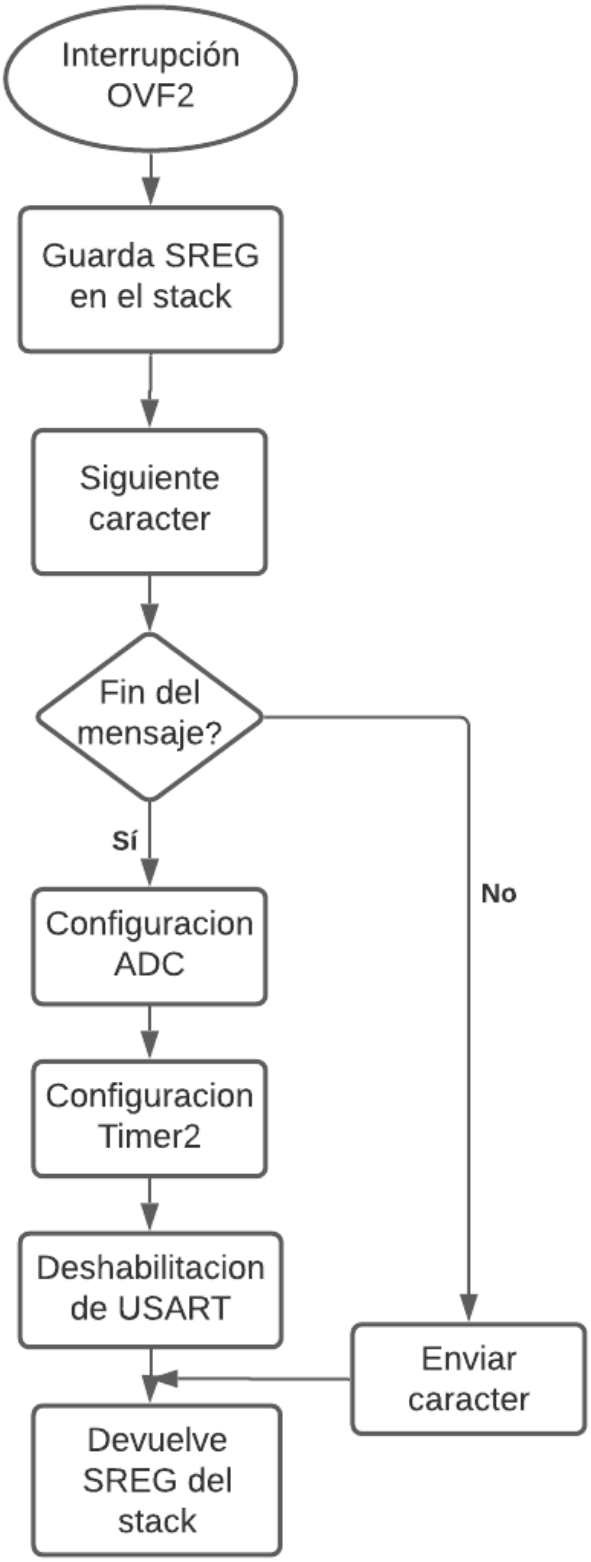

Figura 6: Diagrama de flujo por overflow

5. Resultados

Se lograron los resultados deseados, para ello se hizo uso de la herramienta Data Visualizer del programa MicroChip Studio para realizar la transmisión y recepción de datos a traves de la terminal serie. A su vez se utilizó el programa Octave para graficar los resultados los cuales se pueden observar en el video del siguiente link y en el gráfico :

```
tension = [103 97 92 86 81 77 73 76 88 97 107 115 123
]for i = 1 : length(tension)
  tensions(i) = tension(i)
end

samples = zeros(length(tension),1)

]for i = 1 : length(tension)
  samples(i) = samples(i).+i
end

time = samples
tensions = tensions.*19.53e-3
figure(2)
scatter(time,tensions, ylabel = "Tension VC")
holdon;
plot(time,tensions)
```

Figura 7: Porción de código para graficar mediante Octave

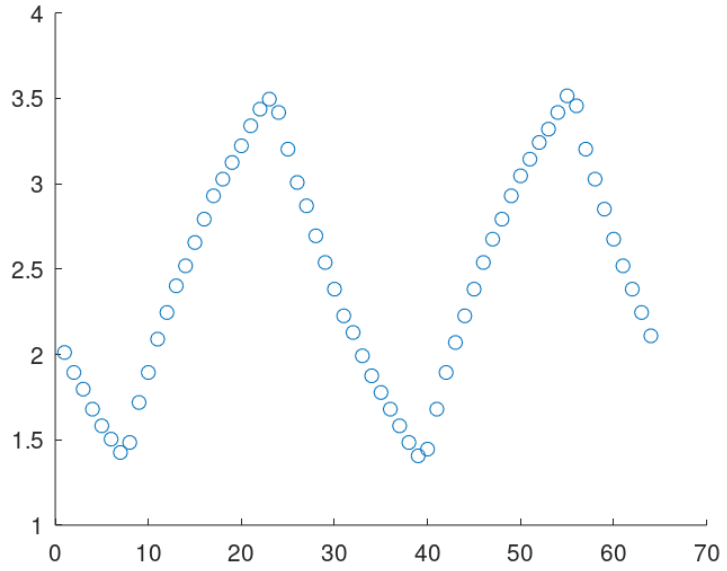

Figura 8: Grafico tensiones sobre el capacitor

Link: https://youtu.be/5lzE-aYfsPQ

6. Conclusiones

Este trabajo práctico se basó en la integración de herramientas configurables para el microprocesador ATMega328p, adquiridas a lo largo del curso. Se puede concluir que permitió replicar el funcionamiento de un osciloscopio, midiendo particularmente la carga y descarga de un capacitor; además de actuar el mismo sistema como un generador de ondas.

7. Anexo

7.1. Código fuente

```
1 ;
2 ; Laboratorio de microprocesadores (86.07)
3 ; Nahila Lutzelschwab
```

```
; Padron : 100686
; 1er cuatrimestre 2021
; Turno Martes 19hs
;

.INCLUDE "m328pdef.inc"

.CSEG       ;Segmento de codigo

.ORG 0x00
  RJMP START ; Se evitan los vectores de interrupciones

.ORG INT0addr ; Vector de interrupcion 0
  RJMP ISR_INT0

.ORG UTXCaddr
  RJMP HANDLER_USART

.ORG ICP1ADDR
  RJMP HANDLER_ICP1

.ORG OVF2ADDR
  RJMP HANDLER_OVF2

.ORG INT_VECTORS_SIZE



; Redefine registros con nombres significativo
.EQU OCR0A_VALUE = 127
.EQU DUTY_CYCLE = 128
.EQU PERIOD = 256
.EQU AMOUNT_CHECK = 21
.EQU AMOUNT_SAMPLES = 64
.EQU BAUD_RATE = 76800
.EQU FREQ = 16000000
.EQU BPS = ((FREQ/16/BAUD_RATE) - 1)   ; Baud rate prescale

.DEF TEMP = R16
.DEF TEMP2 = R17
.DEF PRE_TEMP3 = R18
.DEF TEMP3 = R19
.DEF PRE_TEMP4 = R20
.DEF TEMP4 = R21
.DEF TEMP5 = R22
.DEF TEMP6 = R23
.DEF FREQ_REG = R24
.DEF PERIOD_REG = R25
.DEF SAMPLES_REG = R26




START:
  RCALL STACK_INITIALIZE
  RCALL PORTS_CONF
  RCALL INT0_CONF
  RCALL USART_CONF
  RCALL TIMER0_CONF
  RCALL TIMER1_CONF

  CLR FREQ_REG
  CLR PERIOD_REG
  CLR TEMP5
  CLT
  CBI TIFR1, ICF1
  SEI

LOOP:
```

```
73    RJMP LOOP
74
75
76
77
78  ; Inicia el stack en la parte alta de la memoria
79  STACK_INITIALIZE:
80    LDI TEMP, LOW(RAMEND)
81    OUT SPL, TEMP
82    LDI TEMP, HIGH(RAMEND)
83    OUT SPH, TEMP
84    RET
85
86  ; Configura los puertos
87  PORTS_CONF:
88
89  /*
90    Entradas -> PC0 (AD0)
91    Salidas -> PB0 y PD6
92  */
93
94  ; Setea como entrada PC0 del ADC
95    CLR TEMP
96    OUT DDRC, TEMP
97
98  ; Setea como salida ICP1
99    LDI TEMP, (1<<DDB0)
100   OUT DDRB, TEMP
101
102 ; Setea un nivel bajo en el IPC1(PB0)
103   CBI PORTB, 0
104
105 ; Setea como salida el PWM del Timer0
106   LDI TEMP, (1<<DDD6)
107   OUT DDRD, TEMP
108   RET
109
110
111 ; Configura la interrupcion INT0
112 INT0_CONF:
113
114
115 ; Cualquier cambio en INT0 generara una interrupcion
116   LDS TEMP, EICRA
117   ORI TEMP, (1<<ISC00)
118   ANDI TEMP, ~((1<<ISC01))
119   STS EICRA, TEMP
120
121 ; Habilita la interrupcion INT0
122   IN  TEMP, EIMSK
123   ORI TEMP, (1<<INT0)
124   OUT EIMSK, TEMP
125   RET
126
127
128 ; Configura el USART
129 USART_CONF:
130
131 ; Carga el baud rate USART
132   LDI TEMP, HIGH(BPS)
133   STS UBRR0H, TEMP
134   LDI TEMP, LOW(BPS)
135   STS UBRR0L, TEMP
136
137 ; Configura 8 bits de datos, 1 bit de inicio y otro de fin, sin paridad
138   LDI TEMP, ((0<<UMSEL00)|(0<<UPM00)|(0<<USBS0)|(3<<UCSZ00))
139   STS UCSR0C, TEMP
140
141 ; Habilita la transmision de 8 bits
```

```
142   LDI TEMP, ((0<<RXEN0)|(1<<TXEN0)|(0<<UCSZ02))
143   STS UCSR0B,TEMP
144
145   LDS TEMP, UCSR0B
146   ORI TEMP, (1<<TXCIE0)
147   STS UCSR0B, TEMP
148   RET
149
150
151 ; Configura el Timer0
152 TIMER0_CONF:
153
154 /*
155   Maximo del modo Fast PWM de 8 bits es 0x00FF
156
157   WGM02 WGM01 WGM00
158    0      1      1
159
160 */
161
162 ; Configura modo Fast PWM y control de generador de se al
163   LDS TEMP, TCCR0A
164   ORI TEMP, ((1<<WGM01)|(1<<WGM00)|(1<<COM0A1))
165   ANDI TEMP, ~(1<<COM0A0)
166   OUT TCCR0A, TEMP
167
168 ; Setea el prescaler en 1024
169   LDS TEMP, TCCR0B
170   ORI TEMP, ((1<<CS00)|(1<<CS02))
171   ANDI TEMP, ~((1<<CS01)|(1<<WGM02))
172   OUT TCCR0B, TEMP
173
174 ; Reinicia el contador Timer0
175   CLR TEMP
176   OUT TCNT0, TEMP
177
178 ; Para que tenga 50Hz y un duty cicle de 50% el OCRA_VALUE
179   LDI TEMP, OCR0A_VALUE
180   OUT OCR0A, TEMP
181   RET
182
183
184 ; Configura el Timer1
185 TIMER1_CONF:
186
187 ; Configura el Timer1 en modo captura con disparo de flanco ascendente y prescaler 1024 sin
      cancelador de ruido
188 /*
189   Modo normal    WGM10 WGM11 WGM12 WGM13
190              0       0       0       0
191   maximo: 0XFFF
192
193
194   COM1A1   COM1A0   COM1B1   COM1B0   -    -    WGM11 WGM10 ->TCCR1A
195     0   0      1      1      -    -      0       0
196
197   ICNC1  ICES1    - WGM13 WGM12  CS12   CS11   CS10   ->TCCR1B
198    0       1       -   0       0     1    0    1
199
200 */
201
202   LDS TEMP, TCCR1A
203   ANDI TEMP, ~((1<<WGM11)|(1<<WGM10)|(1<<COM1A1)|(1<<COM1A0))
204   STS TCCR1A, TEMP
205
206  ;Configura la captura con flanco ascendente y un prescaler de 1024
207   LDS TEMP, TCCR1B
208   ORI TEMP, ((1<<CS10)|(1<<CS12)|(1<<ICES1))
209   ANDI TEMP, ~((1<<WGM13)|(1<<WGM12)|(1<<ICNC1)|(1<<CS11))
```

```
210   STS TCCR1B, TEMP
211
212 ; Habilita la interrupcion de captura
213   LDS TEMP, TIMSK1
214   ORI TEMP, (1<<ICIE1)
215   STS TIMSK1, TEMP
216
217 ; Resetea el Timer1
218   CLR TEMP
219   STS TCNT1H, TEMP
220   STS TCNT1L, TEMP
221   RET
222
223
224 ; Configura el Timer2
225 TIMER2_CONF:
226
227 /*
228   Modo normal WGM20 WGM21
229           0       0
230
231   Prescaler de 32   CS22 CS21   CS20
232             0      1       1
233 VER
234 */
235
236   LDS  TEMP, TCCR2A
237   ANDI TEMP, ~((1<<WGM21)|(1<<WGM20))
238   STS TCCR2A, TEMP
239 ; Configura un prescaler de 32 para obtener un delay de 0.5ms
240   LDS TEMP, TCCR2B
241   ORI TEMP, (1<<CS20)|(1<<CS21)
242   ANDI TEMP, ~((1<<WGM22)|(1<<CS22))
243   STS TCCR2B, TEMP
244
245 ; Habilita la interrupcion por overflow del Timer2
246   LDS TEMP, TIMSK2
247   ORI TEMP, (1<<TOIE2)
248   STS TIMSK2, TEMP
249
250 ; Reinicia el Timer2
251   CLR TEMP
252   STS TCNT2, TEMP
253   RET
254
255
256 ; Configura el conversor analogico digital
257 ADC_CONF:
258 /*
259   ADEN   ADSC   ADATE ADIF   ADIE   ADPS2 ADPS1 ADPS0 -> ADCSRA
260    1      1      1      0      0      1      1      1
261
262           ACME                          ADTS2 ADTS1 ADTS0 -> ADCSRB
263    -          -     -     -     0      0      0
264
265    REFS1   REFS0 ADLAR      MUX3   MUX2   MUX1   MUX0   -> ADMUX
266     0    1   1         0      0      0      0
267
268 */
269
270 ;  Habilita el ADC, la conversion, el trigger y configura un prescaler de 128 para una
      frecuencia de 125kHz
271   LDS TEMP, ADCSRA
272   ORI TEMP, ((1<< ADPS0)|(1<< ADPS1)|(1<< ADPS2)|(1<<ADEN)|(1<<ADATE)|(1<<ADSC))
273   STS ADCSRA, TEMP
274
275 ; Configura el modo de ejecucion libre
276   LDS TEMP, ADCSRB
277   ANDI TEMP, ~((1 << ADTS0)|(1 << ADTS1)|(1 << ADTS2))
```

```
278   STS ADCSRB, TEMP
279
280 ; Configura el registro ADC multiplexor: AVCC con capacitor externo, ajusta el resultado hacia
      la izquierda y unica entrada ADC0
281   LDS TEMP, ADMUX
282   ORI TEMP, (1<<ADLAR)|(1<<REFS0)
283   ANDI TEMP, ~((1<<MUX0)|(1<<MUX1)|(1<<MUX2)|(1<<MUX3)|(1<<REFS1))
284   STS ADMUX, TEMP
285   CLR SAMPLES_REG
286   RET
287
288 SEND_MESSAGE:
289   LPM TEMP, Z+       ; Carga un caracter
290   CPI TEMP, 0x00       ; Se fija si es el fin de cadena
291   BREQ END_SENT_MESSAGE
292
293 LOOP_MESSAGE:
294   LDS TEMP3, UCSR0A
295   SBRS TEMP3, UDRE0 ; Espera que el buffer de transmision este vacio
296   RJMP LOOP_MESSAGE
297
298   STS UDR0, TEMP     ; Transmite el caracter
299   RJMP SEND_MESSAGE
300
301 END_SENT_MESSAGE:
302   RET
303
304 SEND_FIRST_CH:
305   LPM TEMP, Z+
306   STS UDR0, TEMP
307   RET
308
309 ; Configura la rutina de interrupcion externa
310 ISR_INT0:
311   PUSH TEMP2
312   IN TEMP2, SREG
313   PUSH TEMP2
314
315   SBI PORTB, 0  ; Setea el ICP1 (PB0) para activar la interrupcion de captura por ser por
      flanco ascendente
316
317   POP TEMP2
318   OUT SREG, TEMP2
319   POP TEMP2
320   RETI
321
322 ; Maneja la interrupcion por captura
323 HANDLER_ICP1:
324   PUSH TEMP2
325   IN  TEMP2, SREG
326   PUSH TEMP2
327
328 ; Guarda en dos registros la parte baja y alta de registro de la captura (valor del timer)
      para contar los flancos detectados
329   LDS TEMP3, ICR1L
330   LDS TEMP4, ICR1H
331
332   INC FREQ_REG
333   INC PERIOD_REG
334
335   CPI FREQ_REG, 2
336   BREQ SECOND_FLANK
337
338   CPI PERIOD_REG, 3
339   BREQ THIRD_FLANK
340
341 FIRST_FLANK:
342   MOV PRE_TEMP3, TEMP3
343   MOV PRE_TEMP4, TEMP4
```

```
344   RJMP END_HANDLER_ICP1
345
346 SECOND_FLANK:
347   SUB TEMP3, PRE_TEMP3
348   SUB TEMP4, PRE_TEMP4
349   CPI TEMP3, DUTY_CYCLE ; Compara los dos primeros flancos para calcular el duty cycle
350   BRNE FOUND_ERROR
351   RJMP END_HANDLER_ICP1
352
353 THIRD_FLANK:
354   CLR PERIOD_REG      ; Cada tres flancos resetea el contador
355   SUB TEMP3, PRE_TEMP3
356   SBC TEMP4, PRE_TEMP4
357
358   CPI TEMP3, HIGH(PERIOD)
359   BRNE END_HANDLER_ICP1
360   CPI TEMP3, LOW(PERIOD)
361   BRNE END_HANDLER_ICP1
362
363   CPI FREQ_REG, AMOUNT_CHECK  ; Para transmitir los datos chequea que sean correctos, lo hace
        cada 10 periodos
364   BRNE END_HANDLER_ICP1
365
366   CLR FREQ_REG      ; Resetea el contador de los 10 periodos
367   STS TCNT1H, FREQ_REG
368   STS TCNT1L, FREQ_REG
369
370   LDI ZL, LOW(MESSAGE<<1)
371   LDI ZH, HIGH(MESSAGE<<1)
372   RCALL SEND_FIRST_CH
373   RJMP SEND_DATA_ADC  ; Envia los datos del conversor
374
375
376 FOUND_ERROR:
377   LDI ZL, LOW(ERROR_MESSAGE<<1)
378   LDI ZH, HIGH(ERROR_MESSAGE<<1)
379   RCALL SEND_FIRST_CH
380
381   CLR FREQ_REG
382   CLR PERIOD_REG
383   STS TCNT1H, FREQ_REG
384   STS TCNT1L, FREQ_REG
385
386   RJMP END_HANDLER_ICP1
387
388 SEND_DATA_ADC:
389   LDI TEMP6, 0xFF
390
391 ; Deshabilita el modo ICP1
392   LDI TEMP, ~(1<<ICIE1)
393   STS TIMSK1, TEMP
394
395 END_HANDLER_ICP1:
396   CBI PORTB, 0  ; Setea el ICP1 a cero
397   POP TEMP2
398   OUT SREG, TEMP2
399   POP TEMP2
400   RETI
401
402
403 ; Maneja interrupcion por desborde del Timer2
404 HANDLER_OVF2:
405   PUSH TEMP2
406   IN  TEMP2, SREG
407   PUSH TEMP2
408
409   LDS R23, ADCL
410   LDS R24, ADCH
411 /*
```

```
LOOP_OVF2:
   LDS TEMP2, UCSR0A
   SBRS TEMP2, UDRE0      ; Espera que el buffer este vacio
   RJMP LOOP_OVF2
*/
   STS UDR0, R24      ; Transmite el byte.

   INC SAMPLES_REG
   CPI SAMPLES_REG, AMOUNT_SAMPLES
   BRNE END_HANDLER_OVF2

; Apaga el Timer2
   LDS TEMP, TCCR2B
      ANDI TEMP, ~((1 << CS20)|(1 << CS21)|(1 << CS22))
      STS TCCR2B, TEMP


END_HANDLER_OVF2:
   POP TEMP2
   OUT SREG, TEMP2
   POP TEMP2
   RETI


HANDLER_USART:
   PUSH TEMP2
   IN    TEMP2, SREG
   PUSH TEMP2

   LPM TEMP, Z+
   CPI TEMP, 0x00
   BREQ COMPLETE_MESSAGE
   STS UDR0, TEMP
   RJMP   END_TRANSMITION

COMPLETE_MESSAGE:
   CPI r21, 0xFF
   BRNE END_TRANSMITION
   RCALL ADC_CONF
   RCALL TIMER2_CONF

   LDS TEMP, UCSR0B
   ANDI TEMP,   ~(1<<TXCIE0)
   STS UCSR0B, TEMP

END_TRANSMITION:
   POP TEMP2
   OUT SREG, TEMP2
   POP TEMP2
   RETI


MESSAGE: .DB " Frecuencia = 50Hz, DutyCycle=50% " ,0x0D, 0x0A, 0, 0
ERROR_MESSAGE: .DB " Error ", 0x0D, 0x0A, 0
```

8. Bibliografía

- Mazidi, M. A., Naimi, S., Naimi, S. (2010). AVR Microcontroller and Embedded Systems: Using Assembly and C (Pearson Custom Electronics Technology). New Jersey, United States of America: Pearson
- ATMEGA328P Datasheet (PDF) - ATMEL.http://ww1.microchip.com/downloads/ Corporation. en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P*Datasheet.pdf*