Universidad de Buenos Aires
Facultad De Ingeniera
2016 - 1er Cuatrimestre

86.07 - Laboratorio de microprocesadores

PROYECTO ESPECIAL
Diseño de dispositivo para control de luces
22 de junio de 2016

INTEGRANTES:

Bruno, Nicolas	95191
<nicoo.24@hotmail.com>	
Funes, Nicolas Pablo	94894
<nico.univ46@gmail.com>	

Índice

1. Introducción

Se diseñó un dispositivo capaz de controlar la intensidad de luz dependiendo de la cantidad de luz y la deteccion de movimiento en el ambiente. Este puede ser utilizado por ejemplo en el control de las luces de un estadio, habitaciones, oficinas,etc. Su principal beneficio es disminuir el consumo de energia, dado que las luces solo consumiran potencia en el momento que sea requerido.

1.1. Diagrama de bloques

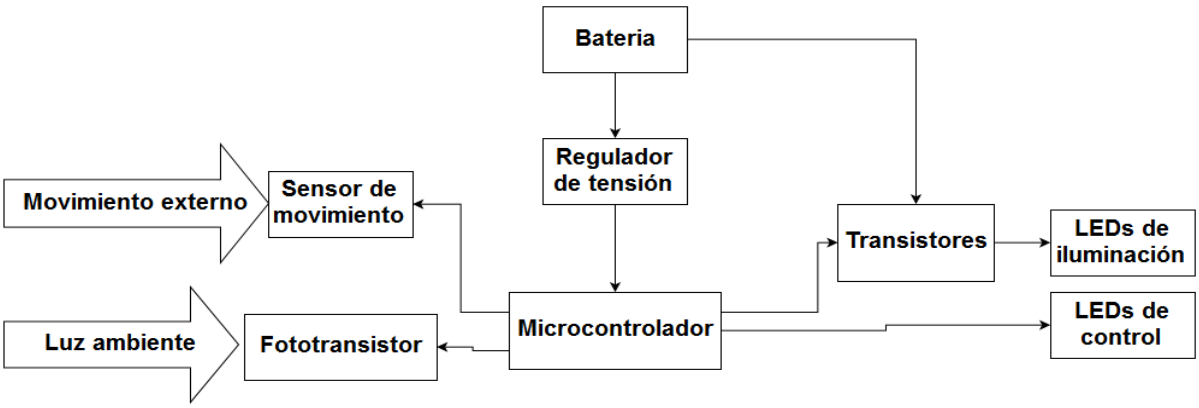

Figura 1: Diagrama de bloques del dispositivo

1.2. Diagrama de flujo

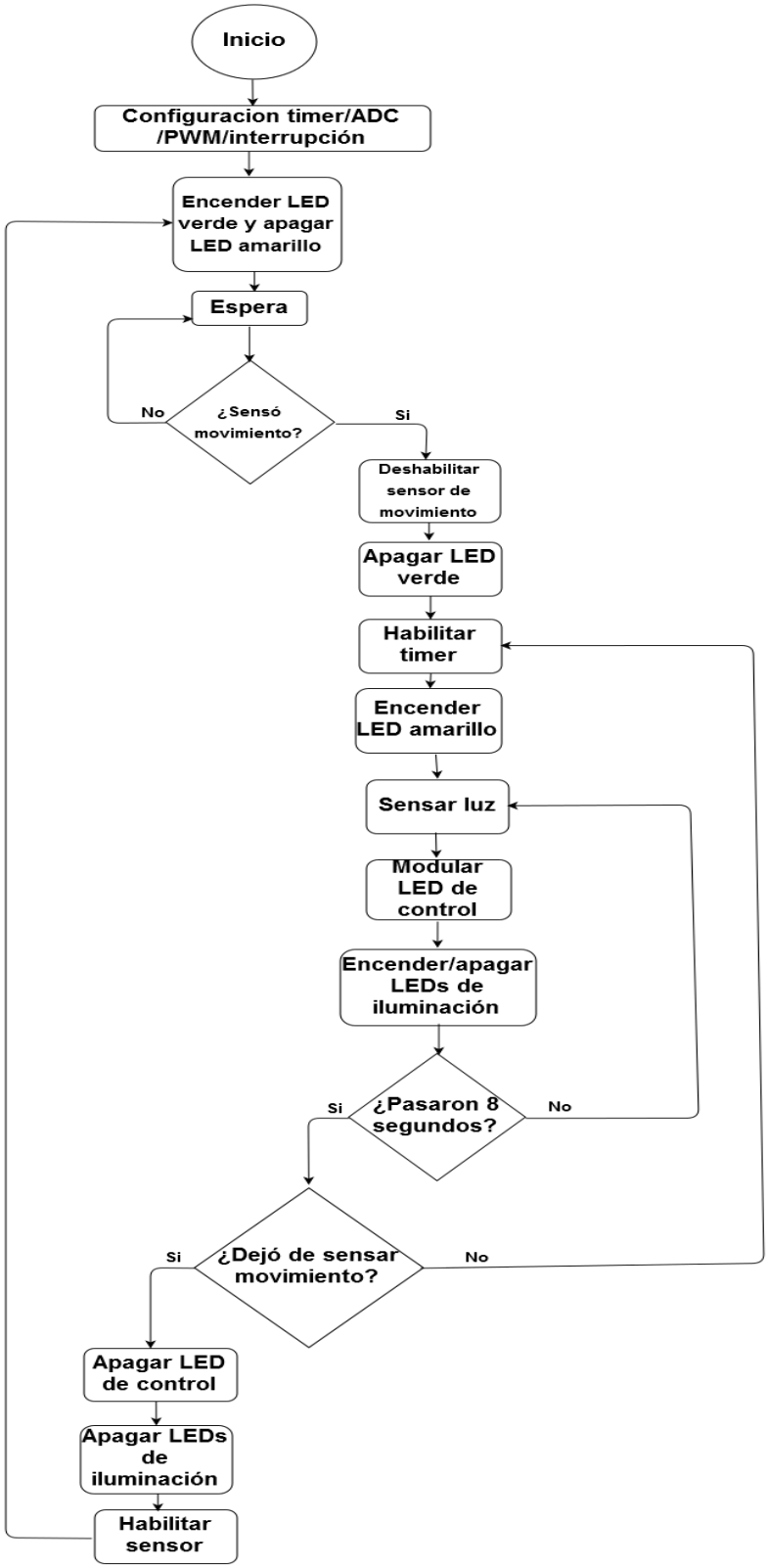

Figura 2: Diagrama de flujo del funcionamiento del dispositivo

1.3. Especificaciones

El dispositivo cuenta con dos mecanismos de control. Para el control de la intensidad de luz utiliza un fototransistor, a traves de este, sensa la cantidad de luz ambiente y en base a esto decide que cantidad de luces encender o apagar. El segundo mecanismo de control es un sensor de movimiento, que en base

a detectar o no movimiento durante cierto tiempo encendiende o apaga el dispositivo respectivamente. Ademas posee un LED de control que modula su brillo indicando la intensidad de luz que esta entregando el dispositivo. Tambien posee dos LEDS para indicar su correcto funcionamiento, uno verde que indica que el dispositivo se encuentra encendido esperando sensar movimiento, y uno amarillo que indica que el dispositivo esta trabajando correctamente.

2. Diseño

2.1. Esquematico

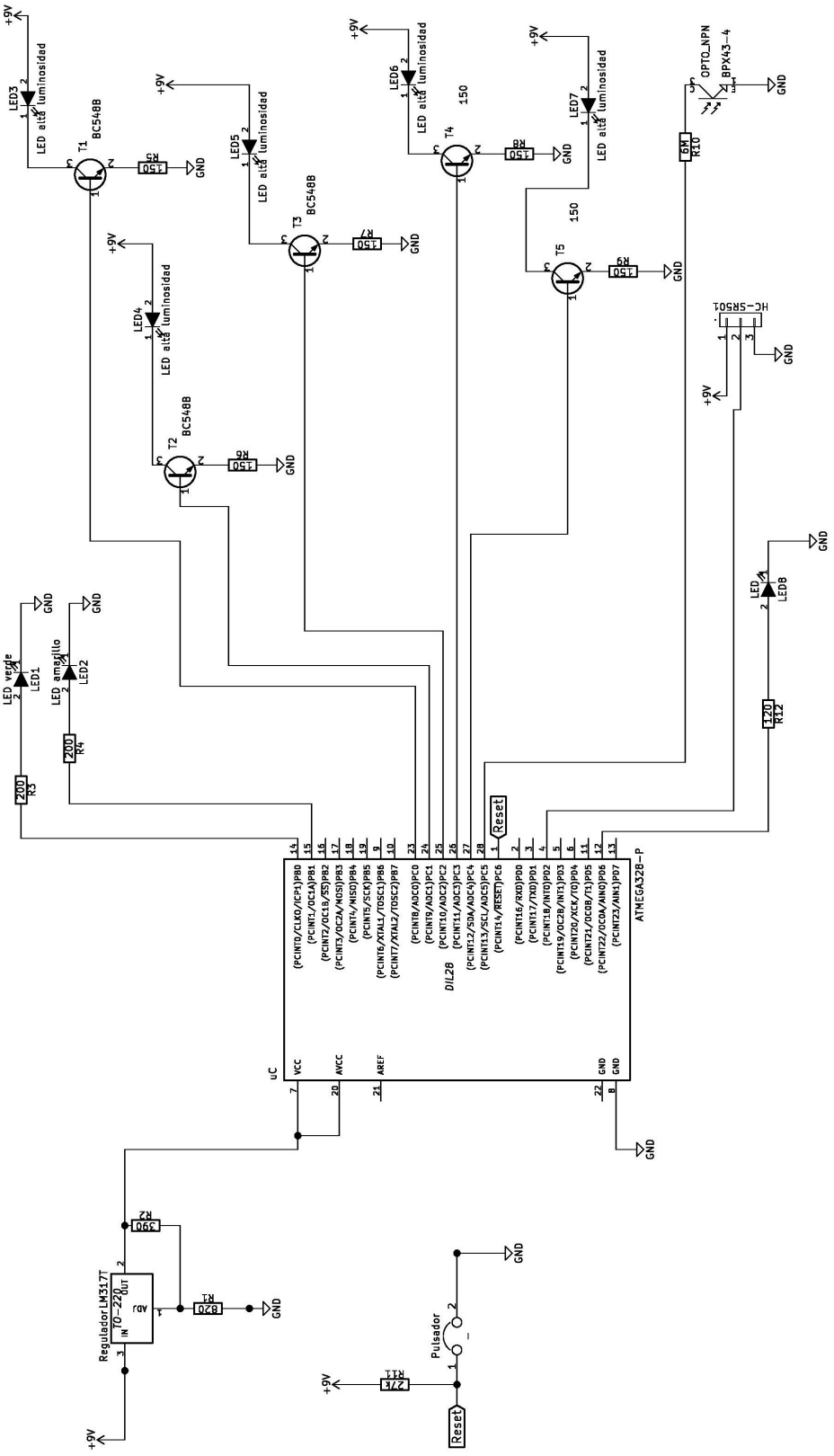

Figura 3: Esquematico del dispositivo

2.2. Hardware

2.2.1. Alimentación

- 1 bateria de 9 V
- 1 regulador de tensión *LM317T* [1]
- 1 resistencia de 820 Ω
- 1 resistencia de 390 Ω

De la hoja de datos del LM317T se obtiene que la tensión de salida será:

$$V_{CC} = 1{,}25. \left(1 + \frac{R_1}{R_2}\right) \tag{1}$$

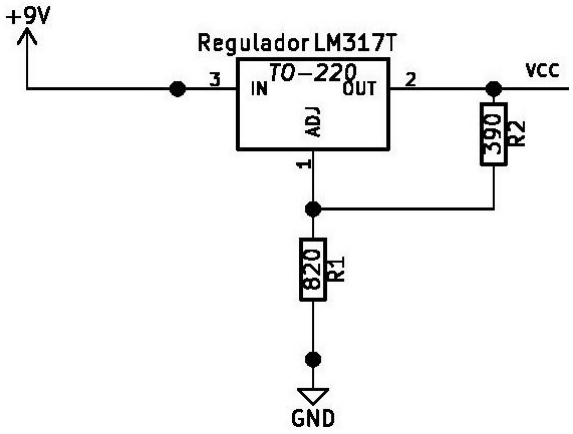

Figura 4: Circuito para el regulador de tensión y su conexión

Por lo tanto se obtiene una tensión de aproximadamente 3,9 V. De la medición del circuito se obtuvo que la tensión entregada por el circuito regulador es de 3,6 V. Se decidió alimentar al microcontrolador con una tensión cercana a 3,3 V debido a que el sensor de movimiento devuelve 3,3 V y se necesita que esta tensión sea interpretada por el microcontrolador como un estado logico alto (se utiliza para habilitar una interrupción por flanco ascendente).

2.2.2. Sensor de movimiento

- *Modulo HC-SR501 PIR infrarrojo de Arduino* [2]

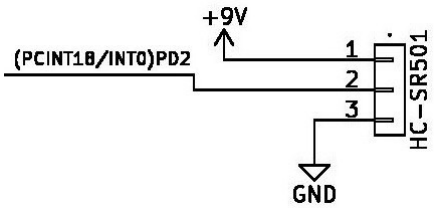

Figura 5: Conexión del sensor de movimiento

Como sensor de movimiento se utilizó el módulo HC-SR501 PIR infrarrojo de Arduino. El mismo se alimenta con 9 V y al sensar movimiento devuelve 3,3 V. Este modulo tarda 1 minuto aproximadamente en estabilizarse, tiempo durante el que devuele tensiones entre 0 y 3,3 V aleatoriamente. Su consumo de corriente es de 50 uA.

2.2.3. Sensor de luz

- 1 *fototransistor BPX43-4* [3]
- 1 resistencia de 6 MΩ

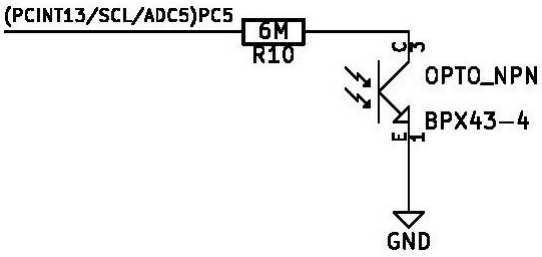

Figura 6: Conexión del sensor de luz

Para sensar la luz ambiente se decidió utilizar un fototransistor BPX43-4. Este posee una corriente de oscuridad del orden de los nanoamperes por lo que es necesaria una intensidad de luz considerable para lograr corrientes tal que su caida en la resistencia genere una tensión suficiente para el ADC (que utiliza como referencia de tension V_{CC} para las aproximaciones sucesivas). Debido a esto al estar frente a luz ambiente, el transistor maneja corrientes del orden de los microampere, por lo tanto se utilizó una resistencia del orden de los megaohms para lograr una mayor sensibilidad frente a condiciones de luz/oscuridad natural.

2.2.4. LEDs de control

- 1 LED verde
- 1 LED amarillo
- 1 LED rojo
- 2 resistencias de 200 Ω
- 1 resistencias de 120 Ω

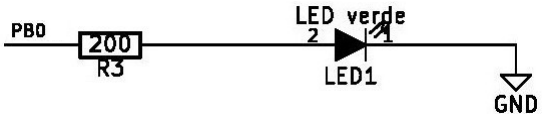

Figura 7: Conexión del LED verde

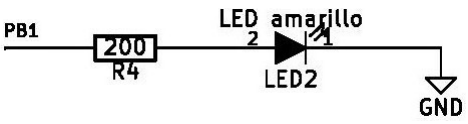

Figura 8: Conexión del LED amarillo

Al tratarse ambos de LEDS standars, se tomó que su caida de tensión es de 1, 5 V y su consumo de corriente de 10 mA (valores típicos), ademas, cada PIN devuelve una tensión de 3, 6 V por lo tanto las resistencias a conectar se calculan como:

$$R_{LED} = \frac{V_{PIN} - V_{LED}}{I_{LED}} \tag{2}$$

Utilizando la ecuación 2 se obtiene que la resistecia normalizada mas próxima es de 200 Ω.

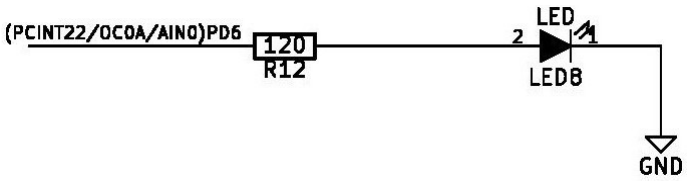

Figura 9: Conexión del LED al pwm

Este LED esta conectado a la salida del PWM, por lo que de acuerdo al ancho de pulso que entregue el micronctrolador, variará su brillo. Debido a que este LED se trata de uno de alto brillo, su caida de tensión es de 2,1 V y su corriente de 20 mA. Utilizando nuevamente la ecuación 2 se obtiene una resistencia de 75 Ω.

2.2.5. LEDs de iluminación

- 5 *LEDs de alta luminosidad* [4]
- 5 transistores BC548B
- 5 resistencias de 50 Ω

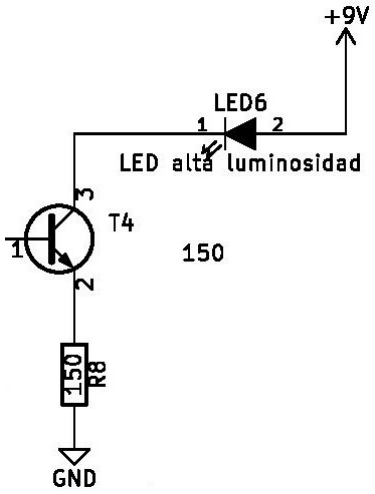

Figura 10: Conexión y circuito de los LEDs de iluminación

La base de los transistores utilizados es conectada a un pin del microcontrolador (ver figura 10). Cada LED de iluminación consume 20 mA, por lo que al tener 5, al estar los 5 encendidos (en condición de penumbra), el micro deberia entregar 100 mA. Para no forzar a que el microcontrolador entregue tanta corriente se decidió utilizar transistores para que cada PIN tenga que entregar una corriente menor. Para calcular la resistencia de emisor, se utilizó la siguiente ecuación:

$$R_E = \frac{V_{PIN} - V_{BE}}{I_E} \tag{3}$$

Considerando despreciable la corriente de base frente a la corriente de colector, se cumple que I_C=I_E, por lo que la corriente que se tenga en el emisor, será la misma que se tenga en el colector. Debido a que es necesario que el transistor se encuentre en modo activo directo, la tension base-emisor será de $0,7\,\text{V}$, por lo que utilizando la ecuación 4, se obtiene una resistencia de emisor de $150\,\Omega$. Considerando que en el LED caen entre $3,2\,\text{V}$ y $3,6\,unitV$ recorriendo desde la fuente de tensión a masa se obtiene la siguiente ecuación:

$$V_{CE} = 9 - I_E.R_E - V_{LED} \tag{4}$$

De la misma se obtiene una tensión colector-emisor entre $2,4\,\text{V}$ y $2,8\,\text{V}$, por lo que el transistor no satura.

2.3. Software

Para este proyecto se utilizó un microcontrolador *ATmega328p* [5] a 8 MHz. A continuación se detalla como se modifican los registros para lograr la configuración deseada.

2.3.1. Interrupción

Para que la regulación de la luz comience cuando el sensor detecta movimiento, se implementó una interrupcion por flanco ascendente en el pin INT0, para esto se modificaron los siguientes registros:

7	6	5	4	3	2	1	0
-	-	-	-	ISC11	ISC10	ISC01	ISC00
	0	0	0	0	0	1	1

Tabla 1: Registro EICRA

A traves de esto se configura el pin que posee INT0 como interrupcion por flanco ascendente.

7	6	5	4	3	2	1	0
-	-	-	-	-	-	INT1	INT0
	0	0	0	0	0	0	1

Tabla 2: Registro EIMSK

A traves de esto se habilita la interrupción externa.

2.3.2. Timer/counter

Luego de arrancar el proceso de sensado, es necesario que durante cierta cantidad de tiempo el programa se encuentre sensando y regulando la luz que entrega, para esto, se configuro el timer/counter 1 por overflow.

7	6	5	4	3	2	1	0
ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10
0	0	0	0	0	1	0	1

Tabla 3: Registro TCCR1B

7	6	5	4	3	2	1	0
COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10
0	0	0	0	0	0	0	0

Tabla 4: Registro TCCR1A

7	6	5	4	3	2	1	0
-	-	ICIE1	-	-	OCIE1B	OCIE1A	TOEI1
0	0	0	0	0	0	0	1

Tabla 5: Registro TIMSK1

Si bien al inicio los ultimos 3 bits del registro TCCR1B estan en 0 para que no este en funcionamiento, luego se setean de la forma indicada en la tabla 3 para que su frecuencia sea la del clkIO dividido 1024, y se habilita a traves del bit TOIE1 (que inicialmente tambien se encuentra en 0) del registro TIMSK1. Del registro TCCR1A no se modifica ningun bit, ya que inicialmente estan todos en 0, y esto indica modo de funcionamiento normal (es decir cuenta hasta 0xFFFF). Se utilizó este modo ya que se desea que el timer dure lo máximo posible.

2.3.3. PWM

Para configurar el PWM se modifican los siguientes registros:

7	6	5	4	3	2	1	0
COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00
1	0	0	0	0	0	1	1

Tabla 6: Registro TCCR0A

7	6	5	4	3	2	1	0
FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00
0	0	0	0	0	0	1	1

Tabla 7: Registro TCCR0B

Debido a como se setean estos registros, se obtiene fast PWM no inversor. Este PWM se caracteriza por ser de alta frecuenca. Ademas, el pulso que genera vale 1 hasta que el contador llega al valor 0CR0A y 0 hasta que el contador llega al final. Debido a como se setearon los bits del registro TCCR0B, la frecuencia será de clkIO/64.

2.3.4. ADC

Para configurar el conversor analógico digitales se modificcan los siguientes registros:

7	6	5	4	3	2	1	0
REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
0	1	0	0	0	1	0	1

Tabla 8: Registro ADMUX

7	6	5	4	3	2	1	0
ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
1	0	0	0	0	0	1	1

Tabla 9: Registro ADCSRA

Debido a como se setearon estos registros, se utiliza el ADC5 con el valor A_{VCC} como referencia (En este caso este valor es igual a V_{CC}) a una frecuencia igual a la del clock interno dividido 8. Se eligió esta frecuencia debido a que la frecuencia del ADC debe ser bastante menor a la frecuencia del clock interno.

2.3.5. Sleep mode

7	6	5	4	3	2	1	0
-	-	-	-	SM2	SM1	SM0	SE
0	0	0	0	0	1	0	1

Tabla 10: Registro SMCR

Se setean los bits de esta forma para tener sleep mode power down. En este, se duermen todos los clks, y son despertados a traves de la interrupción externa del sensor. Esto se realiza para que, mientras no detecta movimiento, el microcontrolador ahorre energia.

3. Resultados

A continuación se muestran dos mediciones realizadas con el oscilocopio. En rojo puede observarse el pulso que entrega el sensor de movimiento, y en azul la señal que se obtiene sobre el LED amarillo

(es decir, cuanto tiempo se encuentra sensando y modulando la luz). De las mismas puede ver que el dispositivo regula durante multiplos de 8 segundos, lo que era esperado de acuerdo al codigo.

Figura 11: Señal de salida del sensor de movimiento (rojo) y señal sobre el LED de control amarillo (azul)

Figura 12: Señal de salida del sensor de movimiento (rojo) y señal sobre el LED de control amarillo (azul)

4. Conclusión

Se logro la implementación de la idea original del trabajo práctico, utilizando los conceptos de programación de los microcontroladores avr. Se utilizaron elementos primordiales como el pwm, conversor a/d, timer, interrupciones para realizar dicho objetivo. Durante la realización del trabajo práctico se encontraron varios inconvenientes, entre estos se destacan la incapacidad de distinguir el espectro luminoso de la manera deseada, esto se debe a que el fototransistor a luz ambiente permite una circulación de corriente relativamente baja, variando levemente ante una variacion de luz moderada. Otro de los problemas encontrados, fue la forma de verificar el correcto funcionamiento del programa, para ello se emplearon los leds de control de manera de verificar el correcto funcionamiento, sin embargo con los leds solo se pudo verificar el estado del programa, por lo que los valores de los registros en cada momento eran desconocidos desde fuera, para solucionar esto podria haberse recurrido a utilizar alguna conexion del tipo puerto serie.

5. Código

```
.include "M328PDEF.INC"
.listmac

.def auxiliar =r22
.def auxiliar_2 =r23
.def auxiliar_adc_1 =r16
.def auxiliar_adc_2 =r17
.def auxiliar_sreg=r18
.def estado_sensor=r19
.def auxiliar_vector=r20
.def auxiliar_comparacion=r21
.cseg
JMP config_init
;Direcciones de los vectores de interrupcion
.ORG 0x001A
JMP TIMER1_OVF
.ORG 0x0002; Dirección del vector INT0
JMP ext_int0;

.ORG 0x0039
config_init:
;Inicializacion del stack
LDI auxiliar,LOW(RAMEND)
OUT SPL,auxiliar
LDI auxiliar,HIGH(RAMEND)
OUT SPH,auxiliar

;*****************************************************
;Configuracion interrupcion del sensor de movimiento
;Se configura INT0 por flanco ascendente
;*****************************************************
LDS auxiliar,EICRA
ORI auxiliar,((1<<ISC01)|(1<<ISC00))
STS EICRA, auxiliar; Registro EICRA
LDS auxiliar,EIMSK
ORI auxiliar,(1<<INT0)
OUT EIMSK,auxiliar

;*****************************************************************************************************
;Configuracion del timer de la interrupcion
;El timer se configura para overflow de manera de habilitar la interrupcion del sensor de movimiento
;*****************************************************************************************************
config_timer:
LDS auxiliar,TCCR1B
ANDI auxiliar,~((0<<ICNC1)|(0<<ICES1)|(0<<WGM13)|(0<<WGM12)|(1<<CS12)|(1<<CS11)|(1<<CS10))
STS TCCR1B,auxiliar
LDI auxiliar_2,0
STS TCNT1H,auxiliar_2
STS TCNT1L,auxiliar_2

;************************************************************
;Configuracion del pwm en fast mode 8 bits
;************************************************************
CONFIGURACION_PWM:
LDI auxiliar,0
OUT OCR0A,auxiliar
LDS auxiliar,TCCR0A
ORI auxiliar,((1<<WGM01)|(1<<WGM00)|(1<<COM0A1)|(0<<COM0A0))
```

```
OUT TCCR0A,auxiliar
LDS auxiliar,TCCR0B
ORI auxiliar,((1<<CS00)|(1<<CS01)|(0<<CS02))
OUT TCCR0B,auxiliar

;*************************************************************
;Configuracion del ADC
;*************************************************************
CONFIGURACION_ADC:
LDS auxiliar,ADMUX;
ORI auxiliar,((1<<REFS0)|(1<<MUX2)|(1<<MUX0));
STS ADMUX, auxiliar; Se configura AVcc como la referencia
LDS auxiliar,ADCSRA;
ORI auxiliar,((1<<ADEN)|(1<<ADPS1)|(1<<ADPS0)); Seteo el valor de division del clk 8M/8
STS ADCSRA, auxiliar; Habilito el adc

;*************************************************************
;Configuracion de los pines del ADC y PWM
;*************************************************************
CONFIGURACION_PINES:
SBI DDRD,6 ;El pin del pwm es (PCINT22/OC0A/AIN0) PD6
CBI DDRC,5;Configuro el pin PC5 (ADC5/SCL/PCINT13) como entrada
SBI DDRB,1;Configuro el led verde para debugear
SBI DDRB,2;Configuro el led amarillo para debugear
SBI DDRC,4;Configuro el pin c4 como salida
SBI DDRC,3;**********c3**********
SBI DDRC,2;**********c2**********
SBI DDRC,1;**********c1**********
SBI DDRC,0;**********c0**********
;**********************
;Programa principal
;**********************
LDI estado_sensor,0;Inicializo el estado del sensor
SBI PORTB,1
RCALL RETARDO_INICIAL
CBI PORTB,1
RCALL RETARDO_INICIAL
SEI;Habilito las interrupciones

HERE:
SBI PORTB,1
CBI PORTB,2
RCALL RETARDO
CBI PORTD,1
RCALL RETARDO
CBI PORTB,1
LDS auxiliar_2,SMCR
ORI auxiliar_2,((0<<SM1)|(1<<SE))
OUT SMCR,auxiliar_2
SLEEP
HERE_IT:
SBRS estado_sensor,0
RJMP HERE
RCALL DESACTIVAR_SENSOR
CBI PORTB,1;
HERE_TIMER:
RCALL HABILITAR_TIMER
ldi estado_sensor,1

SENSAR:
```

```
SBI PORTB,2
SENSAR_IT:
  RCALL SENSAR_LUZ
  RCALL MODULACION_LED
  RCALL CONTROL_LUZ
  SBRC estado_sensor,0
  RJMP SENSAR_IT
  RCALL DESACTIVAR_TIMER
  SBIC PIND,2
  RJMP HERE_TIMER
RCALL APAGAR_LED_CONTROL
RCALL HABILITAR_SENSOR
RCALL APAGAR_LUZ
RJMP HERE

;*********************************************************************

SENSAR_LUZ:
LDS auxiliar_adc_1,ADCSRA;
ORI auxiliar_adc_1,(1<<ADSC)
STS ADCSRA,auxiliar_adc_1;
CONVERSION:
LDS auxiliar_adc_1,ADCSRA ;
SBRS auxiliar_adc_1,ADIF;
RJMP CONVERSION;
RET

MODULACION_LED:
LDS auxiliar_adc_1,ADCSRA;
ORI auxiliar_adc_1, (1<<ADIF)
STS ADCSRA,auxiliar_adc_1
LDS auxiliar_adc_1,ADCL;
LDS auxiliar_adc_2,ADCH;
LSR auxiliar_adc_2;%Por relacion llevo el espacios de 1023ptos a 255ptos dividiendo en 4
ROR auxiliar_adc_1;
LSR auxiliar_adc_2
ROR auxiliar_adc_1;
OUT OCR0A,auxiliar_adc_1;
RET

CONTROL_LUZ:
push r30
push r31
push auxiliar_vector
mov auxiliar_comparacion,auxiliar_adc_1
LDI ZL,LOW(VECTOR_LUZ<<1)
LDI ZH,HIGH(VECTOR_LUZ<<1)
LPM auxiliar_vector,Z+
CP auxiliar_comparacion,auxiliar_vector;%Comparo con la maxima intensidad para la que esta todo apaga
BRLO ESTADO1;
LPM auxiliar_vector,Z+
CP auxiliar_comparacion,auxiliar_vector
BRLO ESTADO2;
LPM auxiliar_vector,Z+
CP auxiliar_comparacion,auxiliar_vector
BRLO ESTADO3;
LPM auxiliar_vector,Z+
CP auxiliar_comparacion,auxiliar_vector
BRLO ESTADO4;
LPM auxiliar_vector,Z+
```

```
CP auxiliar_comparacion,auxiliar_vector
BRLO ESTADO5;
LPM auxiliar_vector,Z+
CP auxiliar_comparacion,auxiliar_vector
BRLO ESTADO6;


ESTADO1:
CBI PORTC,0
CBI PORTC,1
CBI PORTC,2
CBI PORTC,3
CBI PORTC,4
RJMP FIN
ESTADO2:
SBI PORTC,0
CBI PORTC,1
CBI PORTC,2
CBI PORTC,3
CBI PORTC,4
RJMP FIN
ESTADO3:
SBI PORTC,0
SBI PORTC,1
CBI PORTC,2
CBI PORTC,3
CBI PORTC,4
RJMP FIN
ESTADO4:
SBI PORTC,0
SBI PORTC,1
SBI PORTC,2
CBI PORTC,3
CBI PORTC,4
RJMP FIN
ESTADO5:
SBI PORTC,0
SBI PORTC,1
SBI PORTC,2
SBI PORTC,3
CBI PORTC,4
RJMP FIN
ESTADO6:
SBI PORTC,0
SBI PORTC,1
SBI PORTC,2
SBI PORTC,3
SBI PORTC,4
RJMP FIN
FIN:
pop auxiliar_vector
pop r31
pop r30
RET

APAGAR_LUZ:
CBI PORTC,0
CBI PORTC,1
CBI PORTC,2
CBI PORTC,3
```

```
CBI PORTC,4
RET

APAGAR_LED_CONTROL:
LDI auxiliar_adc_1,0
OUT OCR0A,auxiliar_adc_1;
RET

DESACTIVAR_SENSOR:
CLI
LDS auxiliar_2,EIMSK
ANDI auxiliar_2,~(1<<INT0)
OUT EIMSK,auxiliar_2
SEI
RET

HABILITAR_SENSOR:
CLI
IN auxiliar_2,EIMSK
ORI auxiliar_2,(1<<INT0)
OUT EIMSK,auxiliar_2
SEI
RET

REACTIVAR_CONTADOR:
LDS auxiliar_2,TCCR1B
ORI auxiliar_2,((1<<CS12)|(0<<CS11)|(1<<CS10))
ANDI auxiliar_2,~((0<<CS12)|(1<<CS11)|(0<<CS10))
RET

DETENER_CONTADOR:
LDS auxiliar_2,TCCR1B
ANDI auxiliar_2,~((0<<ICNC1)|(0<<ICES1)|(0<<WGM13)|(0<<WGM12)|(1<<CS12)|(1<<CS11)|(1<<CS10))
RET

DESACTIVAR_TIMER:
CLI
LDS auxiliar_2,TCCR1B
ANDI auxiliar_2,~((0<<ICNC1)|(0<<ICES1)|(0<<WGM13)|(0<<WGM12)|(1<<CS12)|(1<<CS11)|(1<<CS10))
STS TCCR1B,auxiliar_2
LDI auxiliar_2,0;Desactivo la interrupcion del timer overflow
STS TIMSK1,auxiliar_2;
SEI
RET

HABILITAR_TIMER:
LDI auxiliar_2,0
STS TCNT1H,auxiliar_2
STS TCNT1L,auxiliar_2
LDS auxiliar_2,TCCR1B
ORI auxiliar_2,((1<<CS12)|(0<<CS11)|(1<<CS10))
ANDI auxiliar_2,~((0<<ICNC1)|(0<<ICES1)|(0<<WGM13)|(0<<WGM12)|(0<<CS12)|(1<<CS11)|(0<<CS10))
STS TCCR1B,auxiliar_2
LDI auxiliar_2,1;
STS TIMSK1,auxiliar_2;
RET

dormir_micro:
push auxiliar_2
lds auxiliar_2,SMCR
```

```
ori auxiliar_2,(1<<SM1)
sts SMCR,auxiliar_2
lds auxiliar_2,SMCR
ori auxiliar_2,(1<<SE)
sts SMCR,auxiliar_2
pop auxiliar_2
ret

RETARDO:
LDI R24,33
L1:
LDI R25,200
L2:
LDI R26,250
L3:
DEC R26
BRNE L3
DEC R25
BRNE L2
DEC R24
BRNE L1
RET

RETARDO_INICIAL:
LDI R24,250
L1_INICIO:
LDI R25,250
L2_INICIO:
LDI R26,250
L3_INICIO:
DEC R26
BRNE L3_INICIO
DEC R25
BRNE L2_INICIO
DEC R24
BRNE L1_INICIO
RET

;****************************************************************
;Interrupcion del sensor de movimiento
;****************************************************************
ext_int0:
IN auxiliar_sreg,SREG
PUSH auxiliar_sreg
lds auxiliar_2,SMCR
andi auxiliar_2,~(1<<SE)
out SMCR,auxiliar_2
LDI estado_sensor,1 ;se activo el sensor
POP auxiliar_sreg
OUT SREG,auxiliar_sreg
RETI
;****************************************************************
;Interrupcion del timer overflow
;****************************************************************
TIMER1_OVF:
IN auxiliar_sreg,SREG
PUSH auxiliar_sreg
LDI estado_sensor,0;Dejo de sensar
POP auxiliar_sreg
OUT SREG,auxiliar_sreg
```

```
RETI

VECTOR_LUZ: .db 16,20,32,45,100,255
```

6. Referencias

[1] www.g0kla.com/datasheets/lm317t.pdf

[2] https://www.mpja.com/download/31227sc.pdf

[3] www.osram-os.com/Graphics/XPic3/00101768_0.pdf

[4] www.sycelectronica.com.ar/optoelectronica/LED-010-70042.pdf

[5] http://www.atmel.com/images/atmel-2486-8-bit-avr-microcontroller-atmega8_l_datasheet.pdf