INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

INTRODUCCIÓN A LOS MICROCONTROLADORES

PROFESOR:

Aguilar Sanchez Fernando

ALUMNOS:

Arruti Sánchez Alondra
Carrillo Soto Cristian Eduardo

Grupo: 3CM14

Fecha de Entrega: 23/09/22

PRÁCTICA 2

Contador Ascendente y Descendente

Introducción

En electrónica digital, un contador es un circuito secuencial construido a partir de biestables y puertas lógicas capaces de almacenar y contar los impulsos (a menudo relacionados con una señal de reloj), que recibe en la entrada destinada a tal efecto, así mismo también actúa como divisor de frecuencia. Normalmente, el cómputo se realiza en código binario, que con frecuencia será el binario natural o el BCD natural (contador de decenas). Ejemplo, un contador de módulo 4 pasa por 4 estados, y contaría del 0 al 3. Si necesitamos un contador con un módulo distinto de 2^n, lo que haremos es añadir un circuito combinacional.

Los microcontroladores AVR poseen dos temporizadores/contadores de 8 bits y uno de 16 bits. En uno y otro caso, lo importante para el programa es conocer cuando se alcanza la máxima cuenta y los desbordamientos correspondientes.

En el caso de un contador de 8 bits, la cuenta máxima que puede alcanzar es 255, en cuyo caso la siguiente cuanta puede causar un desbordamiento y llevar al contador a 0. En el caso del contador de 16 bits, lo mismo ocurre para 65,535. El evento de desbordamiento es muy importante para que el programa lea exactamente los resultados del temporizador/contador.

Algunas de las aplicaciones de los contadores digitales son: Como temporizador, son utilizados como parte de circuitos de tiempo donde es necesario llevar la secuencia temporal de ejecución de diferentes procesos, la medición de las revoluciones por minuto de un motor de automóvil, el periodo exacto de un tiempo específico, tal como el tiempo de trayectoria de una bala, producir tonos para crear música o para producir la chispa de una bujía de un sistema de ignición, o proveer el ancho especifico de un pulso u obtener frecuencias específicas para el control de la velocidad de un motor.

Entonces, un contador es un circuito digital capaz de contar sucesos electrónicos, tales como impulsos, avanzando a través de una secuencia de estados binarios, para el caso de la presente práctica será un contador de 8 bits que al llegar a su límite comenzará nuevamente el conteo.

Códigos

1. Código de la configuración de los periféricos

```
1.  // I/O Registers definitions
2.  #include <mega8535.h>
3.  #include <delay.h>
4.  // Declare your global variables here
5.  void main(void)
6.  {
7.  // Declare your local variables here
8.  // Input/Output Ports initialization
9.  // Port A initialization
10. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
11. DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1) |
    (0<<DDA0);
12. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
13. PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) |
    (0<<PORTA1) | (0<<PORTA0);
14. // Port B initialization
15. // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
16. DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<DDB1) |
    (1<<DDB0);
17. // State: Bit7=1 Bit6=1 Bit5=1 Bit4=1 Bit3=1 Bit2=1 Bit1=1 Bit0=1
18. PORTB=(1<<PORTB7) | (1<<PORTB6) | (1<<PORTB5) | (1<<PORTB4) | (1<<PORTB3) | (1<<PORTB2) |
    (1<<PORTB1) | (1<<PORTB0);
19. // Port C initialization
20. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
21. DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) |
    (0<<DDC0);
22. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
23. PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) |
    (0<<PORTC1) | (0<<PORTC0);
24. // Port D initialization
25. // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
26. DDRD=(1<<DDD7) | (1<<DDD6) | (1<<DDD5) | (1<<DDD4) | (1<<DDD3) | (1<<DDD2) | (1<<DDD1) |
    (1<<DDD0);
27. // State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
28. PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) | (0<<PORTD2) |
    (0<<PORTD1) | (0<<PORTD0);
29. // Timer/Counter 0 initialization
30. // Clock source: System Clock
31. // Clock value: Timer 0 Stopped
32. // Mode: Normal top=0xFF
```

```
33. // OC0 output: Disconnected
34. TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);
35. TCNT0=0x00;
36. OCR0=0x00;
37. // Timer/Counter 1 initialization
38. // Clock source: System Clock
39. // Clock value: Timer1 Stopped
40. // Mode: Normal top=0xFFFF
41. // OC1A output: Disconnected
42. // OC1B output: Disconnected
43. // Noise Canceler: Off
44. // Input Capture on Falling Edge
45. // Timer1 Overflow Interrupt: Off
46. // Input Capture Interrupt: Off
47. // Compare A Match Interrupt: Off
48. // Compare B Match Interrupt: Off
49. TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
50. TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);
51. TCNT1H=0x00;
52. TCNT1L=0x00;
53. ICR1H=0x00;
54. ICR1L=0x00;
55. OCR1AH=0x00;
56. OCR1AL=0x00;
57. OCR1BH=0x00;
58. OCR1BL=0x00;
59. // Timer/Counter 2 initialization
60. // Clock source: System Clock
61. // Clock value: Timer2 Stopped
62. // Mode: Normal top=0xFF
63. // OC2 output: Disconnected
64. ASSR=0<<AS2;
65. TCCR2=(0<<WGM20) | (0<<COM21) | (0<<COM20) | (0<<WGM21) | (0<<CS22) | (0<<CS21) | (0<<CS20);
66. TCNT2=0x00;
67. OCR2=0x00;
68. // Timer(s)/Counter(s) Interrupt(s) initialization
69. TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0)
    | (0<<TOIE0);
70. // External Interrupt(s) initialization
71. // INT0: Off
72. // INT1: Off
73. // INT2: Off
74. MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
75. MCUCSR=(0<<ISC2);
76. // USART initialization
```

```
77. // USART disabled
78. UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSZ2) | (0<<RXB8) |
    (0<<TXB8);
79. // Analog Comparator initialization
80. // Analog Comparator: Off
81. // The Analog Comparator's positive input is
82. // connected to the AIN0 pin
83. // The Analog Comparator's negative input is
84. // connected to the AIN1 pin
85. ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
86. SFIOR=(0<<ACME);
87. // ADC initialization
88. // ADC disabled
89. ADCSRA=(0<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (0<<ADPS1) |
    (0<<ADPS0);
90. // SPI initialization
91. // SPI disabled
92. SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);
93. // TWI initialization
94. // TWI disabled
95. TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);
```

2. Código del programa principal en C

```
while (1)
      {
      // Place your code here
      PORTB++;
      PORTD--;
      delay_ms(350);

      };
```

Circuitos

1. Circuito simulado en Proteus

En la figura 1 se puede observar el contador ascendente en el lado izquierdo y el descendente en el lado derecho; en el caso del contador ascendente este inicia en 0 (todo apagado) y el descendente en 255 (todo encendido).

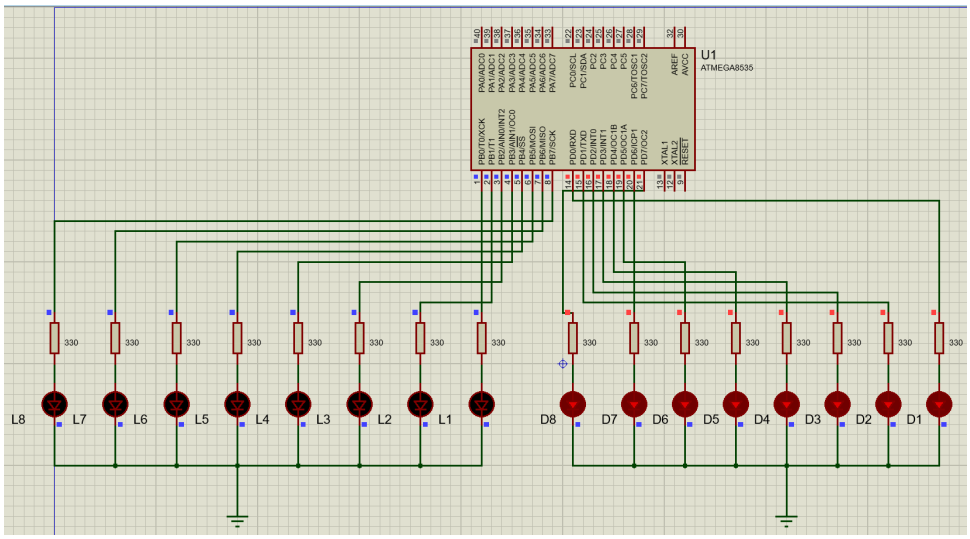

Figura. 1 Inicio de los contadores

En la figura 2 se muestran los LED's verdes como el contador ascendente y los LED's rojos como el descendente, durante el primer cambio.

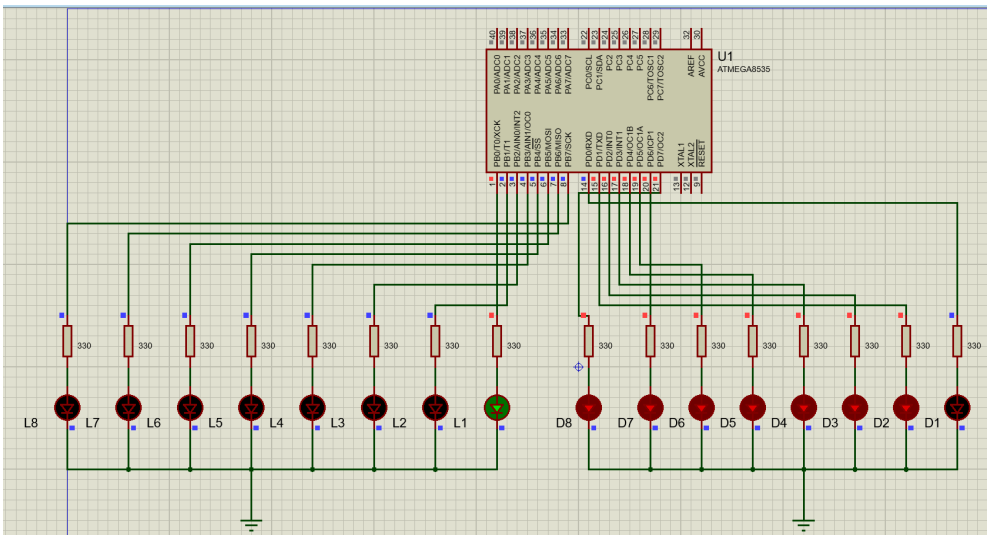

Figura. 2 Primer incremento y decremento

En la figura 3 se tiene el contador ascendente con un 213 en binario y el descendente con un 42 en binario, una forma de comprobar que efectivamente los contadores funcionan es aplicar un not en alguno de los contadores y deberá dar el valor que refleja el contador contrario.

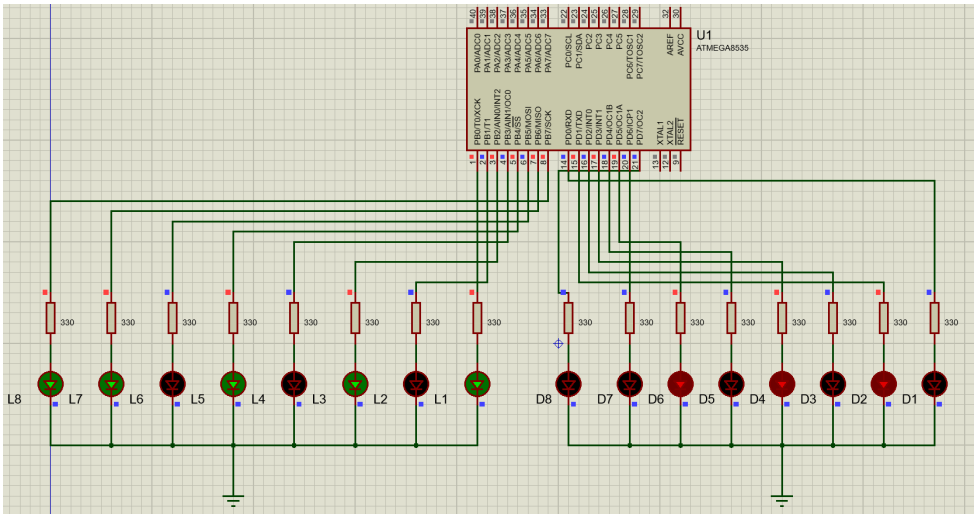

Figura. 3 Contadores funcionando

Finalmente, en la figura 4 se muestra el último cambio de los contadores antes de volver a iniciar el conteo, donde se muestra un caso contrario al presentado en la figura 1.

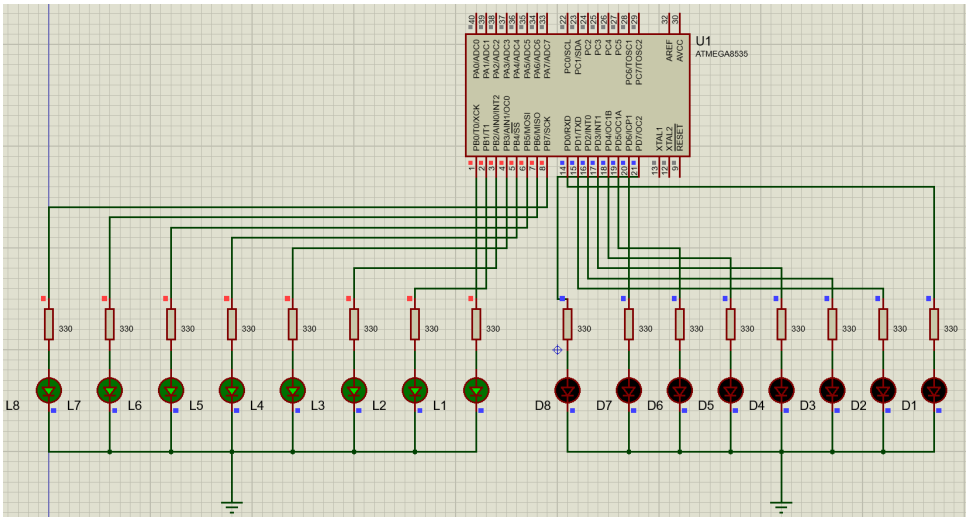

Figura. 4 último incremento y decremento

1. Circuito armado en la Proto

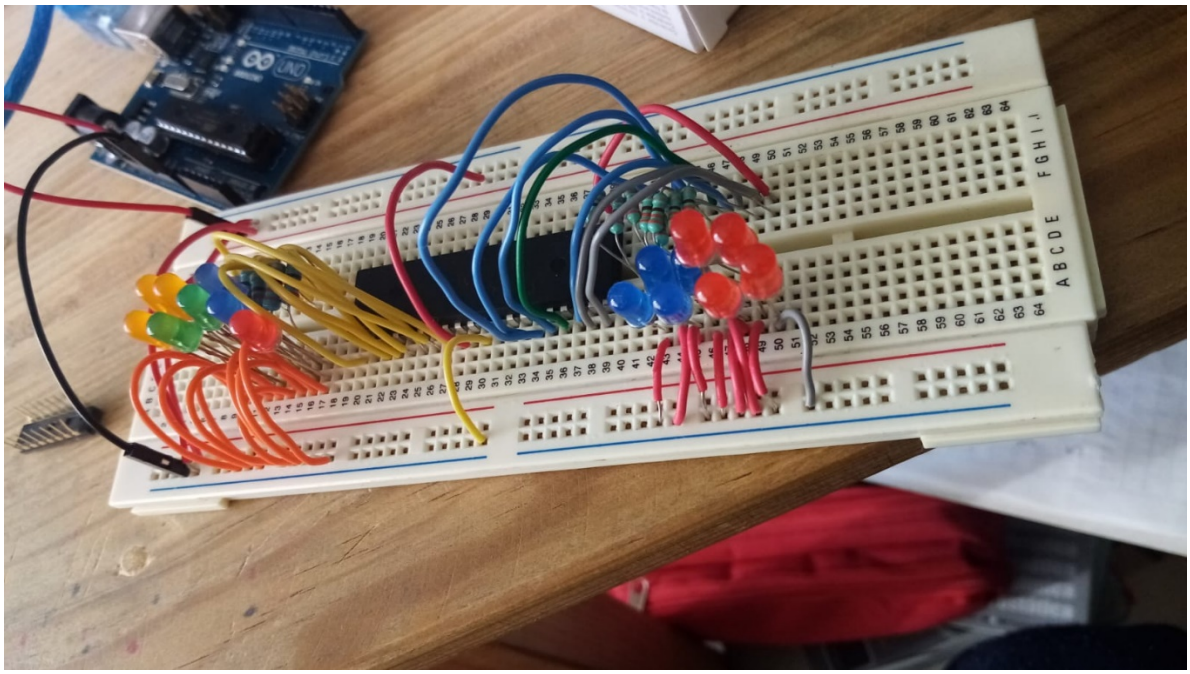

Figura. 5 Circuito Revisado

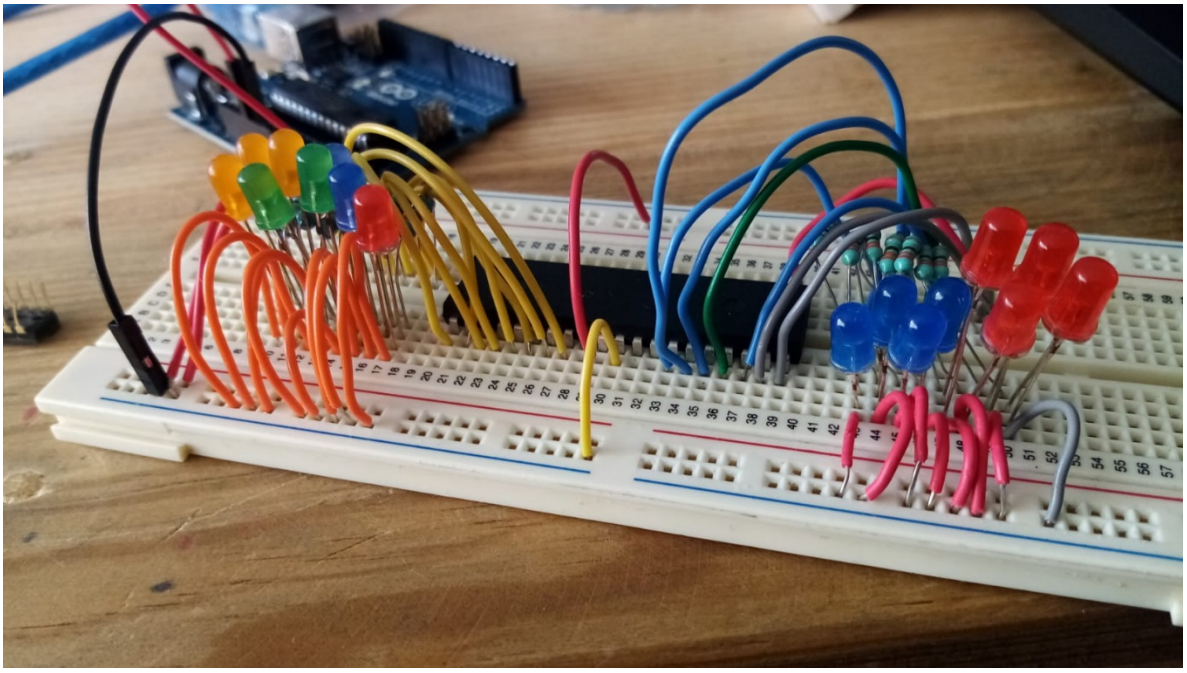

Figura. 6 Circuito Terminado

Conclusiones

Durante la realización de esta práctica se tuvieron que considerar dos detalles para el correcto funcionamiento de los contadores, como es el inicializar las salidas en 1 para el caso del ascendente y en 0 para el descendente, pues de otra forma los contadores se saltarían un paso, estos detalles son importantes y al mismo tiempo casos que no se consideraron en un principio por lo que trabajar con mayor detalle en la lógica de los programas y no tanto en la realización del código será importante para trabajos posteriores.

Por otro lado, en la presente práctica se utiliza una biblioteca llamada delay.h que permite el uso de retardos en el microcontrolador permitiendo que los valores o mejor dicho cambios sean perceptibles para el ojo, además como ya se mencionó en la anterior práctica al ir conociendo más funciones y formas en las que se comportan los puertos será posible desarrollar las siguientes actividades con menores dificultades y mayor entendimiento.

Arruti Sánchez Alondra

El contador lo había visto con anterioridad en Arquitecturas de Computadoras, sin embargo, no de la forma que se abarca ahora tan solo verlo de forma ascendente y descendente lo veía muy interesante ya que no encontraba la forma de como poder hacer sin provocar un bucle, el programa que se realizó en CodeVision fue la cable para poder evitar ese bucle ya que se agrega un “delay” que evite es bucle.

Otra parte interesante fue el armado del circuito ya que en este caso tenemos dos salidas que reciben directo el programa y lo que hará es mandar de forma directa la señal del contador, si bien en arquitectura de computadoras, lo que hacíamos era programarlo en la tarjeta y no hacer un circuito aquí me gusto que si armamos el circuito tan a detalle y que teníamos que hacer que funcionara sin el bucle.

Carrillo Soto Cristian Eduardo

Bibliografía

[1] ¿Qué es y para qué se usa un contador digital? (s. f.). Netatmo. Recuperado 21 de septiembre de 2022, de https://www.netatmo.com/es-es/guides/energy/heating/consumption/linky-meter

[2] Corazto, C. (s. f.). *Capítulo 4 temporizador contador*. Scribd. Recuperado 21 de septiembre de 2022, de https://es.scribd.com/document/208475344/Capitulo-4-temporizador-contador