Citedef, Agustín Lacomi, 22/03/22

Informe diseño y desarrollo
PLUVIÓMETRO DIGITAL

Laboratorio Técnicas Digitales
García Gerardo, Lacomi Agustín, Pastafiglia Daniel, Saluzzi Sergio, Stacul Adrián

Introducción

Las crecidas repentinas figuran entre los desastres naturales más mortíferos del mundo, pues causan miles de muertes al año y dejan importantes secuelas sociales, económicas y ambientales[1] . Un porcentaje significativo de las crecidas repentinas representan el total de las crecidas y tienen la tasa de mortalidad (entendida como el número de vidas perdidas entre el número de personas afectadas) más elevada de las distintas clases de crecidas (fluviales, costeras, etc.)[2]. Las crecidas repentinas tienen un carácter distinto del de las crecidas fluviales, a saber su escala cronológica es más corta y tienen lugar en un espacio geográfico reducido, de modo que su predicción reviste dificultades muy distintas de las que implica predecir las crecidas de un gran río. Cuando se trata de predecir crecidas repentinas, la prioridad es predecir el propio suceso, y para ello hay que centrarse en dos fenómenos causales: En primer lugar, las lluvias intensas, y en segundo lugar, la lluvia que cae sobre un suelo saturado. Las crecidas repentinas se producen en todo el mundo, y su período de desarrollo varía de región a región, desde unos minutos hasta varias horas, dependiendo de la superficie del terreno y las características geomorfológicas e hidrometeorológicas de la región. Sin embargo, la mayoría de estas zonas carecen de la capacidad y los procesos formales requeridos para emitir avisos de **crecida repentina**.

Definiciones de "crecida repentina":

1. Organización Meteorológica Mundial[3]: una inundación de corta duración que alcanza un caudal máximo relativamente alto.

2. American Meteorological Society[4]: una "...inundación que crece y baja rápidamente con poca o ninguna advertencia, usualmente como resultado de lluvia intensa sobre un área relativamente pequeña".

[1] Hydrologic Research Center. Hydrologic Research Center. [Online]. http://www.hrcwater.org/
[2] World Meteorological Organization. (2018, Febrero) Northwest South America Flash Flood Guidance System (NWSAFFGS). http://www.wmo.int/pages/prog/hwrp/flood/ffgs/nwsaffgs/nwsaffgs.php

[3] Organización Meteorológica Mundial. Organización Meteorológica Mundial. http://www.wmo.int
[4] American Meteorological Society. American Meteorological Society. https://www.ametsoc.org

3. U.S. National Weather Service[5]: una inundación rápida con caudal extremadamente alto en un área normalmente seca, o una crecida rápida del nivel del agua de un río o quebrada por encima del nivel de inundación predeterminado, que ocurre típicamente dentro de las 6 horas siguientes al evento causante (ej., lluvia intensa, ruptura de una represa o liberación del agua atrapada en el hielo). Sin embargo, el tiempo real de ocurrencia podría variar en diferentes partes del país. Una inundación continua se puede intensificar a una crecida repentina en casos donde la lluvia intensa resulta en una crecida rápida de las aguas de inundación.

Pluviómetro

El pluviómetro es un dispositivo que se emplea para calcular las precipitaciones que caen en un cierto lugar durante una determinada cantidad de tiempo. Se usa para medir justamente la cantidad de precipitaciones caídas durante un periodo de tiempo determinado.[6] Este instrumento se encuentra generalmente entre los de una estación meteorológica común y es muy importante, para obtener resultados confiables, que los pluviómetros se encuentren ubicados uniformemente en el área en estudio.

La densidad de lluvia o precipitaciones se puede definir como el cociente entre el área de recolección y el número de medidores de lluvia (pluviómetros) en dicha recolección.

Los datos que refieren a eventos de lluvia es la información requerida más importante para todo tipo de investigación, información y pronósticos hidrológicos.

Dichos datos pueden ser utilizados para:

- Análisis de tormentas
- Diseño de sistemas para mitigar inundaciones.
- Pronósticos de subida de mareas en ríos.
- Otros...

El propósito general de las mediciones de parámetros pluviométricos es de obtener valores precisos y en tiempo real que faciliten el ajuste a los sesgos de las estimaciones de precipitación de los radares y satélites, aportar datos de lluvia para los modelos

[5] https://www.weather.gov/. U.S. National Weather Service. https://www.weather.gov/
[6] https://definicion.de/pluviometro/

hidrológicos y de crecidas repentinas y apoyar los pronósticos generales del estado del tiempo y la predicción de las crecidas repentinas.

La organización mundial de meteorología (OMM) recomienda estos mínimos requisitos para obtener un dato de precipitación confiable[7].

Region	Description	Network density	
		Minimum	tolerable
1	Flat region of temperate, mediterranean and tropical zones.	1 gauge for 600 to 900km^2	1 gauge for 900 to 3000km^2
2	Mountainous area of temperate, mediterranean and trpical zones.	1 gauge for 100 to 250km^2	1 gauge fo 250 to 1000km^2
3	Arid and polar zones	1 gauge for 1500 to 10000km^2	-

Figura 1 - Datos recomendados por la OMM para la ubicación de pluviómetros

Descripción técnica

Cangilón

El sistema a utilizar será, el de doble **cubeta basculante**, este sistema también se le denomina cangilón, nos referiremos de esta forma en lo que continúa el informe. Este conjunto consta de un embudo que conduce el agua colectada a una pequeña cubeta triangular doble, de metal, con una bisagra en su punto medio. Es un sistema cuyo equilibrio varía en función de la cantidad de agua en las cubetas, así que cada vez que cae 7,9 cm^3 de lluvia la báscula oscila, vaciando la cubeta llena, mientras comienza a llenarse la otra. Cada vez que la cubeta doble se mueve, este movimiento es **registrado digitalmente** en una **memoria SD**, almacenando toda la información recolectada[8].

[7] Organización Meteorológica Mundial. (1994) Guía de prácticas hidrológicas. https://hydrologie.org/BIB/OMM/WMOSPA_v5.pdf
[8] https://es.wikipedia.org/wiki/Pluviómetro

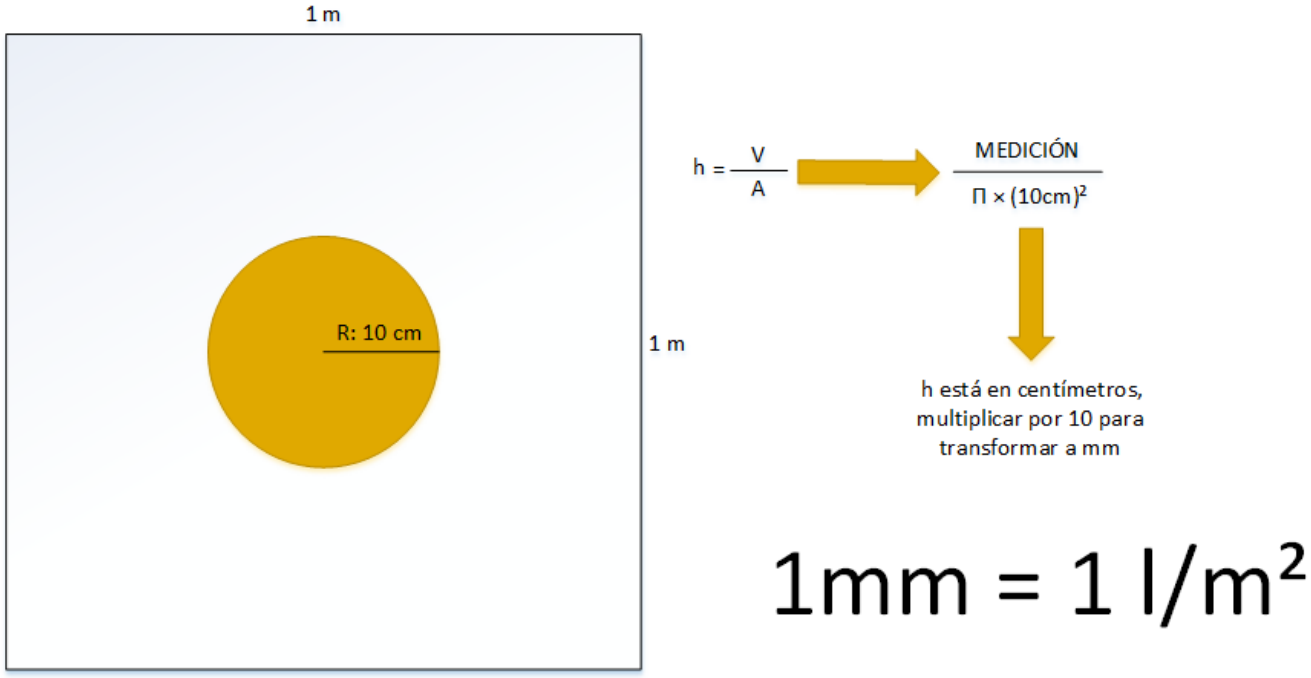
Magnitud de precipitaciones[9]

Las precipitaciones de lluvia se miden a través de altura por **milímetros (mm)** o en forma de **litros por metro cuadrado (l/m²)**

- **mm (milímetros)**: Equivale a la altura que el agua alcanzaría sobre una superficie plana e impermeable con paredes verticales.
- **l/m² (litros por metro cuadrado)**: Equivale a los litros de agua de lluvia caídos en una superficie cuadrada de una longitud de un metro por cada lado con paredes verticales.

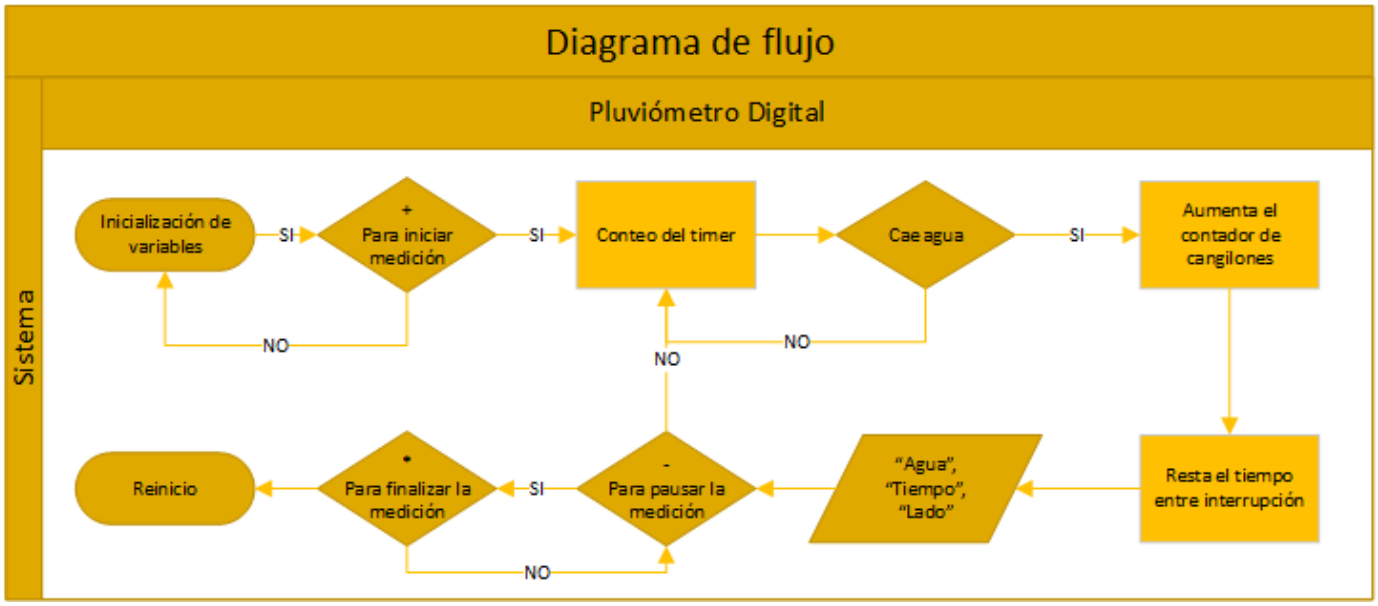
Equivalencias

Existe una correspondencia entre **1mm y 1 L/m²**, porque un litro en un cubo de un metro de ancho y un metro de largo ocupa en volumen exactamente un milímetro de altura. Sólo hay que pensar en que, como es habitual conocer, un cubo de 1 metro cúbico (un metro de alto, uno de ancho y uno de alto) tiene una capacidad de 1000 litros. Si un metro de altura son 1000 milímetros (mm), entonces 1 mm corresponderá, por tanto, a un litro.

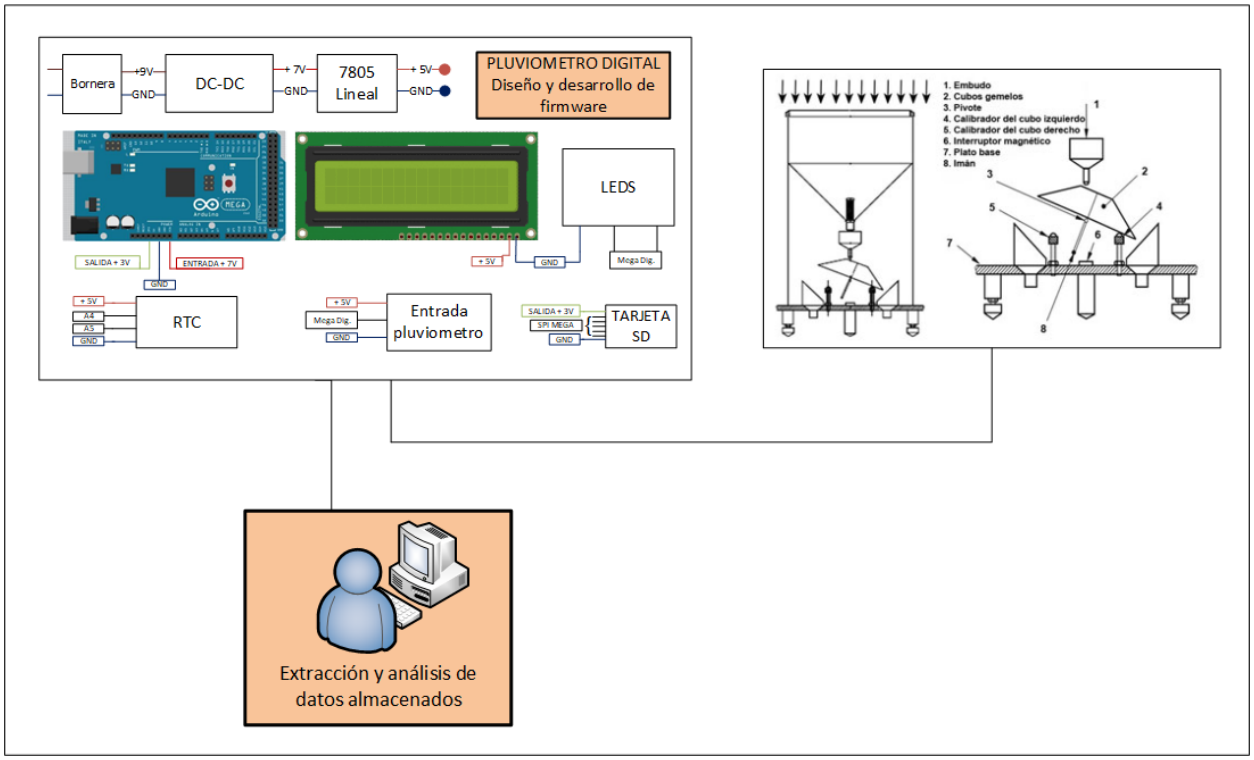


[9]Cómo se miden las precipitaciones en meteorología
https://www.aristasur.com/contenido/como-se-miden-las-precipitaciones-en-meteorologia

Diagrama de flujo

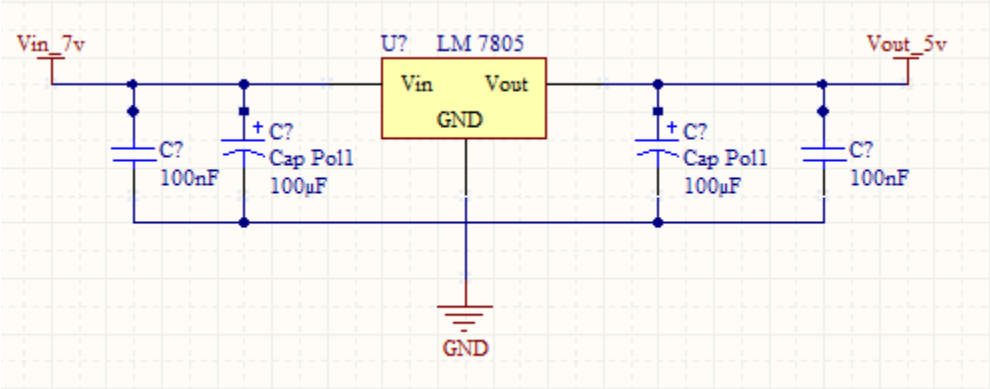


Dibujo explicativo del futuro funcionamiento del pluviómetro

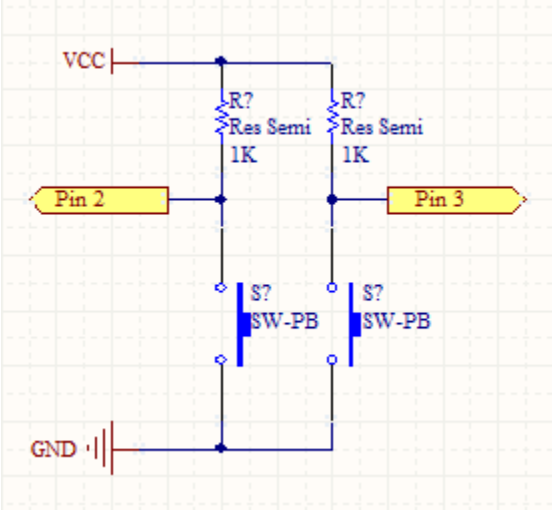


Esquemas Electrónicos

Regulador Lineal 7805



Entrada de pluviómetro-pulsadores



Desarrollo

Ensayos de testeo

Objetivo: Sin establecer una cantidad concreta de agua, se tuvo como objetivo contar la cantidad de veces que oscilaba el cangilón.

Resultado:Resultado exitoso, se logró contar correctamente la cantidad de oscilaciones del cangilón, en un futuro el conteo de las interrupciones servirá para determinar el agua que cae en el pluviómetro.

PRUEBA1_CANGILON

```
#include <LiquidCrystal.h>
#define LED 14
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

volatile int SET_RECARGA = 0;

unsigned long LIMITE = 50;
unsigned long INICIO_TIMER = 0;

  void setup()
  {
    Serial.begin(9600);

    pinMode(LED, OUTPUT);

    pinMode(18, INPUT_PULLUP);

    attachInterrupt(digitalPinToInterrupt(18), AUMENTAR, LOW);

    digitalWrite (LED, HIGH);

    lcd.begin(16,2);
    lcd.setCursor(0, 0);
    lcd.print("CONTADOR: ");
    lcd.setCursor(10, 0);
    lcd.print("0");
  }

  void loop()
  {
  }

  void AUMENTAR()
  {
    if ((millis() - INICIO_TIMER) > LIMITE)
    {
      INICIO_TIMER = millis();
```

```
39.       SET_RECARGA ++;
40.       lcd.setCursor(10, 0);
41.       lcd.print(SET_RECARGA);
42.       Serial.print("CONTADOR: ");
43.       Serial.println(SET_RECARGA);
44.     }
45.   }
```

Ensayo 1 al 5

Objetivo: Ensayando con 100 cm^3 se tuvo como objetivo contar de manera correcta basculaciones y los tiempos entre las oscilaciones del cangilón.

Resultado: El conteo de cangilones fue exitoso, pero no se logró la correcta medición del tiempo.

PRUEBA2_CANGILON

```
1. #include <LiquidCrystal.h>
2. #define LEDLCD 14
3. const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
4. LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
5.
6. volatile int SET_RECARGA = 0;
7.
8. unsigned long LIMITE = 10;
9. unsigned long INICIO_TIMER = 0;
10.
11.   int CONTADOR = 0;
12.
13.   void setup()
14.   {
15.     SET_RECARGA = 0;
16.
17.     TCCR1A = 0;
18.     TCCR1B = 0;
19.     TCCR1B |= (1<<CS10)|(1<<CS12); //para prescaler de 1024
   CS12=1 y CS10=1
20.
21.     TCNT1 = 0xC2F8;
22.
23.     TIMSK1 |= (1<<TOIE1);//timer open interrupt enable
```

```
24.
25.     Serial.begin(9600);
26.
27.     pinMode(LEDLCD, OUTPUT);
28.
29.     pinMode(18, INPUT_PULLUP);
30.
31.     attachInterrupt(digitalPinToInterrupt(18), AUMENTAR, LOW);
32.
33.     digitalWrite (LEDLCD, HIGH);
34.
35.     lcd.begin(16,2);
36.     lcd.setCursor(0, 0);
37.     lcd.print("CONTADOR: ");
38.     lcd.setCursor(10, 0);
39.     lcd.print("0");
40.   }
41.
42.   void loop()
43.   {
44.   }
45.
46.   void AUMENTAR()
47.   {
48.     if ((millis() - INICIO_TIMER) > LIMITE)
49.     {
50.       INICIO_TIMER = millis();
51.       SET_RECARGA ++;
52.       Serial.print("CONTADOR: ");
53.       Serial.println(SET_RECARGA);
54.       if (SET_RECARGA > 0)
55.       {
56.         lcd.setCursor(10, 0);
57.         lcd.print(SET_RECARGA);
58.         Serial.print("TIEMPO: ");
59.         Serial.println(CONTADOR);
60.       }
61.     }
62.   }
63.
64.   ISR(TIMER1_OVF_vect)
65.   {
66.     TCNT1=0xC2F7;
67.
68.     if (SET_RECARGA > 0)
69.     {
70.       CONTADOR ++;
71.     }
```

```
72.   }
```

Ensayo 6 al 11

Objetivo: Ensayando con 50 cm^3 y 100 cm^3 se tuvo como objetivo solucionar el problema del programa anterior que tenía como inconveniente la incorrecta visualización del tiempo. Incluir el lado en el que se encuentra el cangilón es importante para estas instancias de ensayos, ya que permite saber si ambos lados tienen el mismo almacenamiento (si hubiera una goteo constante). También en este punto del proyecto se agregó la pantalla LCD para indicar los datos medidos tanto por este como por el puerto serie.

Resultado: El conteo del sistema fue correcto, se logró modificar la visualización de los tiempos en el puerto serie, de manera que ahora muestra la diferencia entre oscilaciones de cangilón y se agregó exitosamente la posición en la que se encuentra. En la pantalla LCD se muestra hora, minutos y segundos, como también el contador de oscilaciones y el lado en el que se ubica el cangilón.

PRUEBA3_CANGILON

```
1. #include <LiquidCrystal.h>
2. #define LEDLCD 8
3. const int rs = 10, en = 9, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
4. LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
5.
6. volatile int CONTADOR = 0;
7. volatile int CONT_LCD = 0;
8.
9. unsigned long LIMITE = 10;
10.   unsigned long INICIO_TIMER = 0;
11.
12.   int SEGUNDOS_LCD = 0;
13.   int MINUTOS_LCD = 0;
14.   int HORAS_LCD = 0;
15.
16.   int TIEMPO = 0;
17.   int DIFERENCIA = 0;
18.
19.   int PAR = 0;
20.
21.   bool FLAG = 0;
22.   bool FLAG2 = 0;
```

```
23.  bool FLAG3 = 0;
24.
25.  void setup()
26.  {
27.    CONTADOR = 0;
28.    CONT_LCD = 0;
29.
30.    TCCR1A = 0;
31.    TCCR1B = 0;
32.    TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de
   1024 CS12=1 y CS10=1
33.
34.    TCNT1 = 0xC2F8;
35.
36.    TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
37.
38.    Serial.begin(9600);
39.
40.    pinMode(LEDLCD, OUTPUT);
41.
42.    pinMode(2, INPUT_PULLUP);
43.
44.    attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
45.
46.    digitalWrite (LEDLCD, HIGH);
47.
48.    bool FLAG = false;
49.    bool FLAG2 = false;
50.    bool FLAG3 = true;
51.
52.    lcd.begin(16, 2);
53.    lcd.setCursor(0, 0);
54.    lcd.print("CONTADOR: 0");
55.    lcd.setCursor(0, 1);
56.    lcd.print("00:00:00");
57.    lcd.setCursor(15, 1);
58.    lcd.print("B");
59.  }
60.
61.  void loop()
62.  {
63.    if (SEGUNDOS_LCD == 60)
64.    {
65.      MINUTOS_LCD ++;
66.      SEGUNDOS_LCD = 0;
67.      lcd.leftToRight();
68.      lcd.setCursor(6, 1);
69.      lcd.print("00");
```

```
70.     }
71.
72.     if (MINUTOS_LCD == 60)
73.     {
74.       HORAS_LCD ++;
75.       MINUTOS_LCD = 0;
76.       lcd.leftToRight();
77.       lcd.setCursor(3, 1);
78.       lcd.print("00");
79.     }
80.
81.     if (HORAS_LCD > 23)
82.     {
83.       HORAS_LCD = 0;
84.       lcd.leftToRight();
85.       lcd.setCursor(0, 1);
86.       lcd.print("00");
87.     }
88.
89.     if (FLAG2 == 1)
90.     {
91.       lcd.setCursor(2, 1);
92.       lcd.print(":");
93.       lcd.setCursor(5, 1);
94.       lcd.print(":");
95.
96.       if (HORAS_LCD < 10)
97.       {
98.         lcd.rightToLeft();
99.         lcd.setCursor(1, 1);
100.        lcd.print(HORAS_LCD);
101.      }
102.      if (HORAS_LCD >= 10)
103.      {
104.        lcd.leftToRight();
105.        lcd.setCursor(0, 1);
106.        lcd.print(HORAS_LCD);
107.      }
108.
109.      if (MINUTOS_LCD < 10)
110.      {
111.        lcd.rightToLeft();
112.        lcd.setCursor(4, 1);
113.        lcd.print(MINUTOS_LCD);
114.      }
115.      if (MINUTOS_LCD >= 10)
116.      {
117.        lcd.leftToRight();
```

```
118.         lcd.setCursor(3, 1);
119.         lcd.print(MINUTOS_LCD);
120.       }
121.
122.       if (SEGUNDOS_LCD < 10)
123.       {
124.         lcd.rightToLeft();
125.         lcd.setCursor(7, 1);
126.         lcd.print(SEGUNDOS_LCD);
127.       }
128.       if (SEGUNDOS_LCD >= 10)
129.       {
130.         lcd.leftToRight();
131.         lcd.setCursor(6, 1);
132.         lcd.print(SEGUNDOS_LCD);
133.       }
134.
135.       FLAG2 = 0;
136.
137.     }
138.
139.     if (FLAG == 1)
140.     {
141.       FLAG3 = ! FLAG3;
142.       CONTADOR ++;
143.       CONT_LCD = CONTADOR - 1;
144.       DIFERENCIA = SEGUNDOS_LCD - TIEMPO;
145.       TIEMPO = SEGUNDOS_LCD;
146.       if (FLAG3)
147.       {
148.         lcd.setCursor(15, 1);
149.         lcd.print(" ");
150.         lcd.setCursor(15, 0);
151.         lcd.print("A");
152.         Serial.println("LADO: A");
153.       }
154.       if (!FLAG3)
155.       {
156.         lcd.setCursor(15, 0);
157.         lcd.print(" ");
158.         lcd.setCursor(15, 1);
159.         lcd.print("B");
160.         Serial.println("LADO: B");
161.       }
162.
163.       lcd.setCursor(10, 0);
164.       lcd.print(CONT_LCD);
165.       Serial.print("CONTADOR: ");
```

```
166.      Serial.println(CONT_LCD);
167.      Serial.print("TIEMPO: ");
168.      Serial.println(DIFERENCIA);
169.      Serial.println(" ");
170.      FLAG = 0;
171.    }
172.  }
173.
174.  void AUMENTAR()
175.  {
176.    if ((millis() - INICIO_TIMER) > LIMITE)
177.    {
178.      INICIO_TIMER = millis();
179.      FLAG = 1;
180.    }
181.  }
182.
183.  ISR(TIMER1_OVF_vect)
184.  {
185.    TCNT1 = 0xC2F7;
186.
187.    if (CONTADOR > 0)
188.    {
189.      SEGUNDOS_LCD ++;
190.      FLAG2 = 1;
191.    }
192.  }
```

Ensayo medición mínima

Objetivo: Con 50 cm^3 y las bases del programa ya hecho se buscó mediante 8 ensayos poder calcular la parte más chica que haría ceder la oscilación del cangilón. El promedio entre todas las mediciones fue de 5,5. Mediante este valor y una regla de 3 simples se puede determinar el valor que se busca, en este caso 9,1 cm^3.

Resultado: El resultado no fue el esperado, ya que la cantidad de agua fue muy pequeña como para que signifique algo para el pluviómetro, de igual manera este programa sirvió para aprender a almacenar variables en la memoria EEPROM para poder calcular el agua multiplicando los cangilones por esa medida mínima.

PRUEBA4_CANGILON_EEFROM_COMENTADA

```
1.  //********************** Librerías ***************************************//
2.  #include <LiquidCrystal.h>
```

```
3.  #include <EEPROM.h>
4.  //********************** Librerías ************************************//
5.
6.
7.  //********************** definición de pines LCD **********************//
8.  #define LEDLCD 8
9.  const int rs = 10, en = 9, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
10. LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
11. //********************** definición de pines LCD **********************//
12.
13.
14. //********************** definición de variables ************************//
15. volatile float CONTADOR = 0;
16. volatile int CONT = 0;
17. volatile float CONT_LCD = 0;
18. float CTE;
19.
20. unsigned long LIMITE = 10;
21. unsigned long INICIO_TIMER = 0;
22.
23. int SEGUNDOS_LCD = 0;
24. int MINUTOS_LCD = 0;
25. int HORAS_LCD = 0;
26.
27. int TIEMPO = 0;
28. int DIFERENCIA = 0;
29.
30. int PAR = 0;
31.
32. bool FLAG = 0;
33. bool FLAG2 = 0;
34. bool FLAG3 = 0;
35. //********************** definición de variables ************************//
36.
37.
38. //********* ATENCION EEPROM ************//
39. //********float PROMEDIO = 5.5; ******//
40. float EEPROMEDIO;
41. //********* ATENCION EEPROM ************//
42.
43.
44. //***************************************** SETUP
    ******************************************//
45.
46. void setup()
47. {
48. // TIMER 1 //
49. // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
50. // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
51.
52.   CONTADOR = 0;
53.   CONT_LCD = 0;
54.
55.   TCCR1A = 0;
56.   TCCR1B = 0;
57.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
58.
59.   TCNT1 = 0xC2F8;
60.
61.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
62. //********************************** TIMER 1 *********************************//
63.
64.
65. //******************** Grabo en EEPROM el valor promedio ******************//
66.
```

```
                      // EEPROM.put (EEPROMEDIO, PROMEDIO);

//******************** Grabo en EEPROM el valor promedio ******************//


//****************************** Inicializacion puerto serie
  ******************************//
  Serial.begin(9600);
//****************************** Inicializacion puerto serie
  ******************************//


//****************************** SERVICIO DE INTERRUPCIÓN
  ******************************//
  pinMode(2, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);

//****************************** SERVICIO DE INTERRUPCIÓN
  ******************************//


//****************************** inicializacion de variables
  ******************************//
  bool FLAG = false;
  bool FLAG2 = false;
  bool FLAG3 = true;

  CTE = 50 / EEPROMEDIO;
//****************************** inicializacion de variables
  ******************************//


//************************ INICIALIZACION DE PINES DEL LCD ***********************//

  pinMode(LEDLCD, OUTPUT);
  digitalWrite (LEDLCD, HIGH);

//************************ INICIALIZACION DE PANTALLA LCD ***********************//
     lcd.begin(16, 2);
     lcd.setCursor(0, 0);
     lcd.print("AGUA: 0.00");
     lcd.setCursor(0, 1);
     lcd.print("00:00:00");
     lcd.setCursor(15, 1);
     lcd.print("B");
     lcd.setCursor(10,1);
     lcd.print("0");
   //************************ INICIALIZACION DE PANTALLA LCD ***********************//
   }

   //*****************************************************************************
  *************//


   //******************************************** VOID LOOP
  ***************************************//
   void loop()
   {
   //************************ SETEO DE MINUTOS ***********************************//
     if (SEGUNDOS_LCD == 60) // SI ALCANZA LOS 60 SEGUNDOS AUMENTA EN 1 EL MINUTO
     {
       MINUTOS_LCD ++; //aumento en 1 la cantidad de minutos
       SEGUNDOS_LCD = 0; // resetea la variable segundos
       lcd.leftToRight(); // setea el cursor para que escriba de izquierda a derecha
       lcd.setCursor(6, 1); // setea el cursor en la posicion 6 en la linea 1
```

```
125.        lcd.print("00"); // reseteo en display segundos
126.      }
127.    //************************ SETEO DE MINUTOS ***********************************//
128.
129.
130.    //************************ SETEO DE HORAS ***********************************//
131.      if (MINUTOS_LCD == 60) // SI ALCANZA LOS 60 MINUTOS AUMENTA EN 1 LA HORA
132.      {
133.        HORAS_LCD ++; //aumento en 1 la cantidad de horas
134.        MINUTOS_LCD = 0; // reseteo la variable minutos
135.        lcd.leftToRight(); // setea el cursor para que sea de izquierda a derecha
136.        lcd.setCursor(3, 1); // setea el cursor en la posicion 3 linea 1
137.        lcd.print("00"); // reseteo en display minutos
138.      }
139.    //************************ SETEO DE HORAS ***********************************//
140.
141.
142.    //************************ RESETEO DE HORAS ***********************************//
143.      if (HORAS_LCD > 23) // SI ALCANZA UN VALOR MAYOR A 24 RESETEA LA VARIABLE HORAS Y
    RESETEA EL DISPLAY
144.      {
145.        HORAS_LCD = 0; // reseteo la variable en horas
146.        lcd.leftToRight(); // seteo el cursor para que sea de izquierda a derecha
147.        lcd.setCursor(0, 1); // setea el cursor en la posición 0 linea 1
148.        lcd.print("00"); // reseteo en display horas
149.      }
150.    //************************ RESETEO DE HORAS ***********************************//
151.
152.
153.    //************************ FLAG 2( INTERRUPCIÓN DEL TIMER )
    *************************//
154.      if (FLAG2 == 1)
155.      {
156.        lcd.setCursor(2, 1);
157.        lcd.print(":");
158.        lcd.setCursor(5, 1);
159.        lcd.print(":");
160.
161.        if (HORAS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
    izquierda mientras las horas son menores a 10, cosa que las horas aparezcan como 01 y
    no como 1
162.        {
163.          lcd.rightToLeft(); // seteo del cursor
164.          lcd.setCursor(1, 1); // posición del cursor
165.          lcd.print(HORAS_LCD); // imprime las horas entre 0 y 9
166.        }
167.        if (HORAS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda a
    derecha mientras las horas sean mayores o iguales a 10, por eso acá se da vuelta de
    nuevo para que el 10 corresponda y no esté como 01, 11, 21
168.        {
169.          lcd.leftToRight(); // seteo del cursor
170.          lcd.setCursor(0, 1); // posicion del cursor
171.          lcd.print(HORAS_LCD); // imprimer las horas entre 10 y 23
172.        }
173.
174.        if (MINUTOS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
    izquierda mientras los minutos son menores a 10, cosa que los minutos aparezcan como 01
    y no como 1
175.        {
176.          lcd.rightToLeft(); //seteo del cursor
177.          lcd.setCursor(4, 1); // posición del cursor
178.          lcd.print(MINUTOS_LCD); // imprime los minutos entre 0 y 9
179.        }
180.        if (MINUTOS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda a
    derecha mientras los minutos son mayores o iguales a 10, por eso acá se da vuelta de
```

```
   nuevo para que el 10 corresponda y no esté como 01, 11, 21 siguiente el curso de la
   línea 176
181.        {
182.          lcd.leftToRight(); // seteo del cursor
183.          lcd.setCursor(3, 1); // posición del cursor
184.          lcd.print(MINUTOS_LCD); // imprime los minutos entre 10 y 59
185.        }
186.
187.        if (SEGUNDOS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
   izquierda mientras los segundos son menores a 10, cosa que los segundos aparezcan como
   01 y no como 1
188.        {
189.          lcd.rightToLeft(); // seteo del cursor
190.          lcd.setCursor(7, 1); // posicion del cursor
191.          lcd.print(SEGUNDOS_LCD); // imprime los segundos entre 0 y 9
192.        }
193.        if (SEGUNDOS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda
   a derecha mientras los segundos son mayores o iguales a 10, por eso acá se da vuelta de
   nuevo para que el 10 corresponda y no esté como 01, 11, 21 siguiente el curso de la
   línea 176
194.        {
195.          lcd.leftToRight(); //seteo del cursor
196.          lcd.setCursor(6, 1); //posición del cursor
197.          lcd.print(SEGUNDOS_LCD); //imprime los segundos entre 0 y 9
198.        }
199.
200.        FLAG2 = 0; // reseteo del FLAG2
201.
202.      }
203.    //*********************** FLAG 2( INTERRUPCIÓN DEL TIMER )
   *************************//
204.
205.
206.    //*********************** FLAG ( INTERRUPCIÓN POR FLANCO )
   *************************//
207.      if (FLAG == 1) // FLAG del servicio de interrupción
208.      {
209.        FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción
210.        CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS
   DEMÁS
211.        CONT = CONTADOR - 1; // ES EL CONTADOR DE CANGILOS QUE SE MUESTRA EN EL LCD
212.        CONT_LCD = CONTADOR - 1; // ES EL CONTADOR QUE SE USA PARA CALCULAR EL AGUA QUE
   CAYÓ
213.        CONT_LCD = CONT_LCD * 9.1; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1
   CANGILON 9.1CM3
214.        DIFERENCIA = SEGUNDOS_LCD - TIEMPO; // ESTE ES EL CALCULO ENTRE EL TIEMPO DE LA
   PRIMER INTERRUPCIÓN Y EL QUE LE SIGUE, PARA MOSTRAR EN PUERTO SERIE CUANTO TARDA
215.        TIEMPO = SEGUNDOS_LCD; // MAPEA LOS SEGUNDOS EN LA VARIABLE TIEMPO PARA
   REMPLAZAR Y PODER HACER EL CALCULO DE DIFERENCIA EN LA PRÓXIMA INTERRUPCIÓN
216.
217.        if (FLAG3) // INTERRUPCIÓN DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
   INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO A
218.        {
219.          lcd.setCursor(15, 1); // posición del cursor
220.          lcd.print(" "); // borro lo que estaba en la posicion de B
221.          lcd.setCursor(15, 0); // posición del cursor
222.          lcd.print("A"); // Imprime en la posicion el lado A (es en el que está)
223.          Serial.println("LADO: A"); // Por puerto serie imprime el lado tambien en el
   que se encuentra
224.        }
225.        if (!FLAG3) //INTERRUPCIÓN DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
   INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO B
226.        {
227.          lcd.setCursor(15, 0); // posición del cursor
228.          lcd.print(" "); // borro lo que estaba en la posición de B
229.          lcd.setCursor(15, 1); // posición del cursor
```

```
230.          lcd.print("B"); // Imprime en la posición el lado B (es en el que está)
231.          Serial.println("LADO: B"); // Por puerto serie imprime el lado tambien en el
    que se encuentra
232.        }
233.    //************************************* IMPRESIÓN DE VALORES EN LCD
    *************************************//
234.        lcd.leftToRight(); // seteo del cursor
235.        lcd.setCursor(6, 0); // posición del cursor
236.        lcd.print(CONT_LCD); // imprime el agua medida
237.        lcd.setCursor(10,1); // posicion del cursor
238.        lcd.print(CONT); // imprime cantidad de cangilones
239.
240.        Serial.print("AGUA: "); // imprime por puerto serie el agua medida
241.        Serial.println(CONT_LCD); // valor del agua
242.
243.        Serial.print("TIEMPO: "); // imprime por puerto serie el tiempo entre medidas
244.        Serial.println(DIFERENCIA); // valor del tiempo entre medidas
245.        Serial.println(" "); // imprime un enter para separar los valores
246.
247.    //************************************* IMPRESIÓN DE VALORES EN LCD
    *************************************//
248.        FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCIÓN
249.        }
250.    }
251.    //************************ FLAG ( INTERRUPCIÓN POR FLANCO )
    *************************//
252.
253.
254.    //************************** SERVICIO DE INTERRUPCIÓN  *********************//
255.    // CON ANTIRREBOTE //
256.    // CADA VES QUE SE INTERRUMPE PONE LA FLAG EN 1 //
257.
258.    void AUMENTAR()
259.    {
260.      if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
261.      {
262.        INICIO_TIMER = millis();
263.        FLAG = 1;
264.      }
265.    }
266.    //*************************************************************************//
267.
268.
269.    //******************************** TIMER 1 *******************************//
270.    // POR DESBORDAMIENTO //
271.    // TIMER MADRE DE 1s  //
272.    // EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
273.    // LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
274.    ISR(TIMER1_OVF_vect)
275.    {
276.      TCNT1 = 0xC2F7;
277.
278.      if (CONTADOR > 0)
279.      {
280.        SEGUNDOS_LCD ++;
281.        FLAG2 = 1;
282.      }
283.    }
284.    //*************************************************************************//
```

Ensayo 12 al 16

Objetivo: Estos ensayos se realizaron con el objetivo de caracterizar de manera correcta el programa, ya que los ensayos para determinar la medición mínima no fueron fructíferos. Se optó por modificar el programa respetando las bases, pero utilizando otro sistema para almacenar datos. De manera que el software ahora puede realizar 10 pruebas en un solo ensayo, mostrando y calculando lo mismo que antes. Lado, Tiempo entre mediciones, la posición de la memoria en la que se encuentra el tiempo y el promedio de todos los tiempos, la cantidad de cangilones y el promedio de cangilones entre todos los ensayos de la planilla.

Ensayo 12-13

Se muestrea con 50cm³ y se realizaron 15 pruebas.

Ensayo 14-15

Se muestrea con 80cm³ y se realizaron 15 pruebas.

Ensayo 16

Se muestrea con 100cm³ y se realizaron 10 pruebas.

Resultado: Exitoso, mediante todos estos ensayos se logró caracterizar correctamente la constante volumétrica aproximada que puede registrar el cangilón en este caso 7.90cm³, de esta manera los próximos ensayos que se harán será midiendo efectivamente agua y si la medición que hace el pluviómetro coincide con el agua recolectada.

PRUEBA5_CANGILON_EEFROM_COMENTADA

```
#include <LiquidCrystal.h>
#include <EEPROM.h>

//********************** definicion de pines LCD **********************//
#define LEDLCD 8
const int rs = 10, en = 9, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
//********************** definicion de pines LCD **********************//

//********************** Definicion de estructura **********************//
typedef struct ESTRUCTURA
{
  int CANGILONES;
  int TIEMPO[30];
  float TIEMPO_PROMEDIO;
} ESTRUCTURA;
```

```
17.
18. ESTRUCTURA p[10];
19. //********************** Definicion de estructura **********************//
20.
21. //********************** Definicion de variables **********************//
22. int cont_PLANILLA;
23. int cont_tiempo;
24.
25. float PROM_CANGILONES = 0;
26. float TIEMPO_PROM_GENERAL = 0;
27. int AGUA = 100;
28.
29. bool FLAG = 0;
30. bool FLAG2 = 0;
31. bool FLAG3 = 0;
32.
33. int SEGUNDOS_LCD = 0;
34. int MINUTOS_LCD = 0;
35. int HORAS_LCD = 0;
36.
37. int TIEMPO_ANTERIOR = 0;
38. int SEGUNDOS_ARRAY = 0;
39.
40. unsigned long LIMITE = 10;
41. unsigned long INICIO_TIMER = 0;
42.
43.
44. //********************** Definicion de variables **********************//
45.
46.
47. //********************** SETUP **********************//
48. void setup()
49. {
50. // TIMER 1 //
51. // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
52. // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
53.
54.   TCCR1A = 0;
55.   TCCR1B = 0;
56.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
57.
58.   TCNT1 = 0xC2F8;
59.
60.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
61. //********************************** TIMER 1 **********************************//
62.
63.
64. //********************** SERVICIO DE INTERRUPCIÓN **********************//
65.   pinMode(2, INPUT_PULLUP);
66.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
67. //********************** SERVICIO DE INTERRUPCIÓN **********************//
68.
69.
70. //********************** INICIALIZACION DE TYPEDEF STRUCT **********************//
71.   for (int i = 0; i < 10; i++)
72.   {
73.     p[i].CANGILONES = 0;
74.     p[i].TIEMPO_PROMEDIO = 0;
75.
76.     for (int e = 0; e < 30; e++)
77.     {
78.       p[i].TIEMPO[e] = 0;
79.     }
80.   }
81.
82.   cont_tiempo = cont_tiempo - 1;
```

```
83. //********************** INICIALIZACION DE TYPEDEF STRUCT **********************//
84.
85.
86. //********************** INICIALIZACION PUERTO SERIE **********************//
87.   Serial.begin(9600);
88. //********************** INICIALIZACION PUERTO SERIE **********************//
89.
90.
91. //********************** INICIALIZACION DE PINES **********************//
92.   pinMode(LEDLCD, OUTPUT);
93.   digitalWrite (LEDLCD, HIGH);
94. //********************** INICIALIZACION DE PINES **********************//
95.
96.
97. //********************** INICIO DE PANTALLA **********************//
98.   Serial.println();
99.   Serial.print("AGUA EN MEDICIÓN: ");
100.      Serial.println (AGUA);
101.      Serial.println ();
102.      Serial.print("PLANILLA: ");
103.      Serial.println (cont_PLANILLA);
104.
105.    //********************** INICIO DE PANTALLA **********************//
106.    }
107.    //********************** SETUP **********************//
108.
109.
110.    //********************** LOOP **********************//
111.    void loop()
112.    {
113.      SERIAL_EVENT (); // LECTURA DE TECLADO
114.
115.      if (FLAG == 1) // FLAG DEL SERVICIO DE INTERRUPCIÓN
116.      {
117.        cont_tiempo ++; //muestra la posición del array en la que se encuentra el
    tiempo de interrupción
118.
119.        FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción para mostrar la
    posicion en la q esta si en A o B
120.
121.        p[cont_PLANILLA].CANGILONES ++; //aumenta en 1 la cantidad de cangilones
122.
123.        p[cont_PLANILLA].TIEMPO[cont_tiempo] = SEGUNDOS_ARRAY - TIEMPO_ANTERIOR; //
    mapea el tiempo de interrupción y le resta el tiempo anterior
124.        TIEMPO_ANTERIOR = SEGUNDOS_ARRAY; // mapea el tiempo de interrupción en la
    variable tiempo_anterior para restar en la siguiente interrupción
125.
126.        Serial.print("CANGILONES: "); // imprime "CANGILONES: "
127.        Serial.println(p[cont_PLANILLA].CANGILONES); // imprime la cantidad de
    cangilones hasta el momento
128.        Serial.print("SEGUNDOS: "); //  imprime "SEGUNDOS: "
129.        Serial.println(p[cont_PLANILLA].TIEMPO[cont_tiempo]); // imprime los segundos
    entre esta y la interrupción anterior
130.        Serial.print("POSICION DEL TIEMPO: "); // imprime "POSICION DEL TIEMPO: "
131.        Serial.println(cont_tiempo); // imprime la posicion en la que mapea, el tiempo
    entre interrupciones, del array en el typedef struct
132.
133.        FLAG = 0; // reinicio de flag
134.
135.        if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO A
136.        {
137.          lcd.setCursor(15, 1); // posicion del cursor
138.          lcd.print(" "); // borro lo que estaba en la posicion de B
139.          lcd.setCursor(15, 0); // posicion del cursor
140.          lcd.print("A"); // Imprime en la posicion el lado A (es en el que está)
```

```
141.          Serial.println("LADO: A"); // Por puerto serie imprime el lado tambien en el
    que se encuentra
142.          Serial.println();
143.        }
144.        if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO B
145.        {
146.          lcd.setCursor(15, 0); // posicion del cursor
147.          lcd.print(" "); // borro lo que estaba en la posicion de B
148.          lcd.setCursor(15, 1); // posicion del cursor
149.          lcd.print("B"); // Imprime en la posicion el lado B (es en el que está)
150.          Serial.println("LADO: B"); // Por puerto serie imprime el lado tambien en el
    que se encuentra
151.          Serial.println();
152.        }
153.
154.      }
155.    }
156.    //********************** LOOP **********************//
157.
158.
159.    void SERIAL_EVENT()
160.    {
161.      while (Serial.available()) // lectura del teclado
162.      {
163.        int ESTADO = '0'; // 0 ASCII
164.
165.        ESTADO = Serial.read(); // La variable estado pasara a valer lo que detecte del
    teclado
166.
167.        if (ESTADO == '+') // cuando presione + pasará de planilla
168.        {
169.          cont_tiempo = 0; // reinicio del contador del array de
    p[].TIEMPO[cont_tiempo]
170.
171.          p[cont_PLANILLA].CANGILONES = p[cont_PLANILLA].CANGILONES - 1; // resta 1 de
    los cangilones ya que el primer cangilon es de inicio, y de esa manera se puede sacar
    el tiempo promedio de todas las mediciones
172.
173.          for (int i = 0; i < 30; i++) // suma todos los tiempos de cada cangilon
174.          {
175.           p[cont_PLANILLA].TIEMPO_PROMEDIO =  p[cont_PLANILLA].TIEMPO[i] +
    p[cont_PLANILLA].TIEMPO_PROMEDIO ; // suma el valor actual de la variable
    tiempo_promedio con el valor siguiente del tiempo
176.          }
177.
178.          p[cont_PLANILLA].TIEMPO_PROMEDIO = p[cont_PLANILLA].TIEMPO_PROMEDIO /
    p[cont_PLANILLA].CANGILONES; // divide el tiempo total entre la cantidad de cangilones
179.
180.          Serial.print("TIEMPO_PROMEDIO: "); // imprime "TIEMPO_PROMEDIO: "
181.          Serial.println (p[cont_PLANILLA].TIEMPO_PROMEDIO); // Imprime el valor del
    tiempo promedio
182.          Serial.println();
183.
184.          cont_tiempo = cont_tiempo - 1; // le resta uno para q a la hora de mapear no
    comience en la posicion 1 del array
185.
186.          cont_PLANILLA ++; // aumenta en 1 el contador de planillas
187.          FLAG2 = 0; // reinicio del flag del timer por desbordamiento
188.
189.          SEGUNDOS_ARRAY = 0; // reinicio de la variable segundos (es la q aumenta en
    uno cada vez que interrumpe el timer por desbordamiento)
190.          TIEMPO_ANTERIOR = 0; // reinicio de la variable tiempo anterior, asi se
    reinicia el sistema que calcula la diferencia entre el tiempo de interrupcion
191.
192.          Serial.println();
```

```
193.          Serial.print("CAMBIO A PLANILLA: "); // imprime "CAMBIO A PLANILLA: "
194.          Serial.println (cont_PLANILLA); // imprime el contador de planillas
195.        }
196.
197.        if (ESTADO == '-') // cuando presione - finalizará la medición
198.        {
199.          p[cont_PLANILLA].CANGILONES = p[cont_PLANILLA].CANGILONES - 1; // resta 1 de
    los cangilones ya que el primer cangilon es de inicio en la planilla correspondiente en
    este caso el cont_PLANILLA valdría 9 ya que sería la décima medicion pero podría
    cambiar en caso de querer que sean mas, o menos
200.          cont_PLANILLA ++; // aumenta en 1 la posicion en la q esta el contador de
    planillas para poder sacar luego el promedio de  cangilones totales
201.
202.          for (int i = 0; i < 10; i++) // suma en la variable PROM_CANGILONES todos los
    cangilones
203.          {
204.            PROM_CANGILONES = PROM_CANGILONES + p[i].CANGILONES; // suma los cangilones
205.          }
206.
207.          PROM_CANGILONES = PROM_CANGILONES / cont_PLANILLA; // saca el promedio
    diviendolo por 10
208.
209.          Serial.println();
210.          Serial.print("PROMEDIO DE CANGILONES: "); // imprime "PROMEDIO DE CANGILONES:
    "
211.          Serial.println(PROM_CANGILONES); // imprime el valor del promedio de los
    cangilones
212.          Serial.println("FINALIZACION DE PRUEBAS"); // imprime "FINALIZACION DE
    PRUEBAS"
213.        }
214.      }
215.    }
216.    //********************** LOOP **********************//
217.
218.
219.    //********************** SERVICIO DE INTERRUPCIÓN **********************//
220.    void AUMENTAR()
221.    {
222.      if ((millis() - INICIO_TIMER) > LIMITE) // antidebounce
223.      {
224.        INICIO_TIMER = millis(); // antidebounce
225.        FLAG = 1; // inicio del flag 1 (cuando interrumpe el cangión)
226.        FLAG2 = 1; // inicio del flag 2 (inicio del timer cuando interrumpe el
    cangilón)
227.      }
228.    }
229.    //********************** SERVICIO DE INTERRUPCIÓN **********************//
230.
231.
232.    //********************** SERVICIO DE INTERRUPCIÓN POR DESBORDAMIENTO
    **********************//
233.    // TIMER //
234.    ISR(TIMER1_OVF_vect)
235.    {
236.      TCNT1 = 0xC2F7;
237.
238.      if (FLAG2) // cuando el FLAG2 == 1
239.      {
240.        SEGUNDOS_ARRAY ++; // aumenta en uno los segundos
241.      }
242.    }
243.    //********************** SERVICIO DE INTERRUPCIÓN POR DESBORDAMIENTO
    **********************//
```

Ensayo 17 al 19

Objetivo: Los ensayos que se detallan aquí se realizaron con el objetivo de probar el programa que efectivamente mide el aproximadamente el agua por cangilón. Se le agregó como función adicional controlar las mediciones mediante teclas del teclado. La tecla " + " inicia la medición (si no se inicializa el software no medirá nada), la tecla " - " permite pausar el ensayo en caso de que ocurriera algún contratiempo, por último la tecla " * " reiniciará el software. Las funciones de la pantalla LCD siguen usándose.

Ensayo 17

Se muestrea con 150cm³ la medición que realizó el software fue de 150.1cm³. Lo recolectado por el almacenamiento de medición fue de 148cm³.

Ensayo 18

Se muestrea 250cm³ la medición que realizó el software fue de 252.8cm³ . Lo recolectado por el almacenamiento de medición fue de 249.23cm³.

En segunda instancia se ensaya con 290cm³ de agua, el pluviómetro realizó como medición 284.4cm³, teniendo un remanente en el sector de recolección 285cm³

Ensayo 19

En el ensayo número 19 se ensayó con medidas de agua desconocidas, para verificar finalmente si lo que recolectaba el sector de almacenamiento correspondía relativamente con la medición que hace el pluviómetro. En la primera prueba el pluviómetro midió 237cm³ y el sistema de recolección almacenó 240cm³. En la segunda, el pluviómetro midió 402.9cm³ y se recolectaron 371 cm³.

Resultado: Las mediciones fueron las esperadas, con un error de 3cm³ y un error de 31.9cm³ que se sabía que podría estar, de esta manera se procede a ensayar directamente la lluvia natural para corroborar el funcionamiento en la cotidianidad

PRUEBA6_VOLUMEN_CANGILON

```
1.  //********************** Liberias ************************************//
2.  #include <LiquidCrystal.h>
3.  #include <EEPROM.h>
4.  //********************** Liberias ************************************//
5.
6.
7.  //********************** definicion de pines LCD **********************//
8.  #define LEDLCD 8
9.  const int rs = 10, en = 9, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
10. LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
11. //********************** definicion de pines LCD **********************//
12.
13.
14. //********************** definicion de variables ************************//
15. volatile float CONTADOR = 0;
16. volatile int CONT = 0;
17. volatile float CONT_LCD = 0;
18. float CTE;
19.
20. unsigned long LIMITE = 10;
21. unsigned long INICIO_TIMER = 0;
22.
23. int SEGUNDOS_LCD = 0;
24. int MINUTOS_LCD = 0;
25. int HORAS_LCD = 0;
26.
27. int TIEMPO = 0;
28. int DIFERENCIA = 0;
29.
30. bool FLAG = 0;
31. bool FLAG2 = 0;
32. bool FLAG3 = 0;
33. bool FLAG4 = 0;
34. //********************** definicion de variables ************************//
35.
36.
37. //********* ATENCION EEPROM ************//
38. //********float PROMEDIO = 5.5; ******//
39. float EEPROMEDIO;
40. //********** ATENCION EEPROM ************//
41.
42.
43. //****************************************** SETUP
    ********************************************//
44.
45. void setup()
46. {
47. // TIMER 1 //
48. // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
49. // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
50.
51.   CONTADOR = 0;
52.   CONT_LCD = 0;
53.
54.   TCCR1A = 0;
55.   TCCR1B = 0;
56.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
57.
58.   TCNT1 = 0xC2F8;
59.
```

```
60.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
61. //********************************* TIMER 1 *******************************//
62.
63.
64. //******************** Grabo en EEPROM el valor promedio ******************//
65.
66. // EEPROM.put (EEPROMEDIO, PROMEDIO);
67.
68. //******************** Grabo en EEPROM el valor promedio ******************//
69.
70.
71. //****************************** Inicializacion puerto serie
    ******************************//
72.   Serial.begin(9600);
73. //****************************** Inicializacion puerto serie
    ******************************//
74.
75.
76. //****************************** SERVICIO DE INTERRUPCIÓN
    ******************************//
77.
78.   pinMode(2, INPUT_PULLUP);
79.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
80.
81. //****************************** SERVICIO DE INTERRUPCIÓN
    ******************************//
82.
83.
84. //****************************** inicializacion de variables
    ******************************//
85.   FLAG = false;
86.   FLAG2 = false;
87.   FLAG3 = true;
88.   FLAG4 = false;
89.
90.   CTE = 7.90;
91. //****************************** inicializacion de variables
    ******************************//
92.
93.
94. //************************ INICIALIZACION DE PINES DEL LCD ***********************//
95.
96.   pinMode(LEDLCD, OUTPUT);
97.   digitalWrite (LEDLCD, HIGH);
98.
99. //************************ INICIALIZACION DE PANTALLA LCD ***********************//
100.      lcd.begin(16, 2);
101.      lcd.setCursor(0, 0);
102.      lcd.print("AGUA: 0.00");
103.      lcd.setCursor(0, 1);
104.      lcd.print("00:00:00");
105.      lcd.setCursor(15, 0);
106.      lcd.print("A");
107.      lcd.setCursor(10, 1);
108.      lcd.print("0");
109.    //************************ INICIALIZACION DE PANTALLA LCD ***********************//
110.
111.      Serial.print("PRESIONE + PARA INICIAR");
112.      Serial.println();
113.
114.      Serial.print("PRESIONE - PARA PAUSAR");
115.      Serial.println();
116.
117.      Serial.print("PRESIONE * PARA REINICIAR");
118.      Serial.println();
119.      Serial.println();
```

```
120.
121.    }
122.
123.    //*****************************************************************************
    *************//
124.
125.
126.    //********************************************** VOID LOOP
    ***************************************//
127.    void loop()
128.    {
129.      SERIAL_EVENT();
130.
131.    //************************ SETEO DE MINUTOS ***********************************//
132.      if (SEGUNDOS_LCD == 60) // SI ALCANZA LOS 60 SEGUNDOS AUMENTA EN 1 EL MINUTO
133.      {
134.        MINUTOS_LCD ++; //aumento en 1 la cantidad de minutos
135.        SEGUNDOS_LCD = 0; // resetea la variable segundos
136.        lcd.leftToRight(); // setea el cursor para que escriba de izquierda a derecha
137.        lcd.setCursor(6, 1); // setea el cursor en la posicion 6 en la linea 1
138.        lcd.print("00"); // reseteo en display segundos
139.      }
140.    //************************ SETEO DE MINUTOS ***********************************//
141.
142.
143.    //************************ SETEO DE HORAS *************************************//
144.      if (MINUTOS_LCD == 60) // SI ALCANZA LOS 60 MINUTOS AUMENTA EN 1 LA HORA
145.      {
146.        HORAS_LCD ++; //aumento en 1 la cantidad de horas
147.        MINUTOS_LCD = 0; // reseteo la variable minutos
148.        lcd.leftToRight(); // setea el cursor para que sea de izquierda a derecha
149.        lcd.setCursor(3, 1); // setea el cursor en la posicion 3 linea 1
150.        lcd.print("00"); // reseteo en display minutos
151.      }
152.    //************************ SETEO DE HORAS *************************************//
153.
154.
155.    //************************ RESETEO DE HORAS ***********************************//
156.      if (HORAS_LCD > 23) // SI ALCANZA UN VALOR MAYOR A 24 RESETEA LA VARIABLE HORAS Y
    RESETEA EL DISPLAY
157.      {
158.        HORAS_LCD = 0; // reseteo la variable en horas
159.        lcd.leftToRight(); // seteo el cursor para que sea de izquierda a derecha
160.        lcd.setCursor(0, 1); // setea el cursor en la posición 0 linea 1
161.        lcd.print("00"); // reseteo en display horas
162.      }
163.    //************************ RESETEO DE HORAS ***********************************//
164.
165.
166.    //************************ FLAG 2( INTERRUPCIÓN DEL TIMER )
    *************************//
167.      if (FLAG2 == 1)
168.      {
169.        lcd.setCursor(2, 1);
170.        lcd.print(":");
171.        lcd.setCursor(5, 1);
172.        lcd.print(":");
173.
174.        if (HORAS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
    izquierda mientras las horas son menores a 10, cosa que las horas aparezcan como 01 y
    no como 1
175.        {
176.          lcd.rightToLeft(); // seteo del cursor
177.          lcd.setCursor(1, 1); // posicion del cursor
178.          lcd.print(HORAS_LCD); // imprime las horas entre 0 y 9
179.        }
```

```
180.        if (HORAS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda a
    derecha mientras las horas sean mayores o iguales a 10, por eso aca se da vuelta de
    nuevo para que el 10 corresponda y no esté como 01, 11, 21
181.        {
182.          lcd.leftToRight(); // seteo del cursor
183.          lcd.setCursor(0, 1); // posicion del cursor
184.          lcd.print(HORAS_LCD); // imprimer las horas entre 10 y 23
185.        }
186.
187.        if (MINUTOS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
    izquierda mientras los minutos son menores a 10, cosa que los minutos aparezcan como 01
    y no como 1
188.        {
189.          lcd.rightToLeft(); //seteo del cursor
190.          lcd.setCursor(4, 1); // posición del cursor
191.          lcd.print(MINUTOS_LCD); // imprime los minutos entre 0 y 9
192.        }
193.        if (MINUTOS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda a
    derecha mientras los minutos son mayores o iguales a 10, por eso aca se da vuelta de
    nuevo para que el 10 corresponda y no esté como 01, 11, 21 siguiente el curso de la
    linea 176
194.        {
195.          lcd.leftToRight(); // seteo del cursor
196.          lcd.setCursor(3, 1); // posicion del cursor
197.          lcd.print(MINUTOS_LCD); // imprime los minutos entre 10 y 59
198.        }
199.
200.        if (SEGUNDOS_LCD < 10) // este if lo que hace es poner el cursor de derecha a
    izquierda mientras los segundos son menores a 10, cosa que los segundos aparezcan como
    01 y no como 1
201.        {
202.          lcd.rightToLeft(); // seteo del cursor
203.          lcd.setCursor(7, 1); // posicion del cursor
204.          lcd.print(SEGUNDOS_LCD); // imprime los segundos entre 0 y 9
205.        }
206.        if (SEGUNDOS_LCD >= 10) // este if lo que hace es poner el cursor de izquierda
    a derecha mientras los segundos son mayores o iguales a 10, por eso aca se da vuelta de
    nuevo para que el 10 corresponda y no esté como 01, 11, 21 siguiente el curso de la
    linea 176
207.        {
208.          lcd.leftToRight(); //seteo del cursor
209.          lcd.setCursor(6, 1); //posición del cursor
210.          lcd.print(SEGUNDOS_LCD); //imprime los segundos entre 0 y 9
211.        }
212.
213.        FLAG2 = 0; // reseteo del FLAG2
214.
215.      }
216.    //********************** FLAG 2( INTERRUPCIÓN DEL TIMER )
    *************************//
217.
218.
219.    //********************** FLAG ( INTERRUPCIÓN POR FLANCO )
    *************************//
220.    if(FLAG4)
221.    {
222.        if (FLAG == 1) // FLAG del servicio de interrupción
223.        {
224.          FLAG3 =! FLAG3; // cambia de estado el FLAG3 cada interrupción
225.          CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS
    DEMAS
226.          CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1
    CANGILON 7.9CM3
227.          DIFERENCIA = SEGUNDOS_LCD - TIEMPO; // ESTE ES EL CALCULO ENTRE EL TIEMPO DE
    LA PRIMER INTERRUPCION Y EL QUE LE SIGUE, PARA MOSTRAR EN PUERTO SERIE CUANTO TARDA
```

```
228.          TIEMPO = SEGUNDOS_LCD; // MAPEA LOS SEGUNDOS EN LA VARIABLE TIEMPO PARA
    REPLAZAR Y PODER HACER EL CALCULO DE DIFERENCIA EN LA PROXIMA INTERRUPCIÓN
229.
230.          if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO A
231.          {
232.            lcd.setCursor(15, 1); // posicion del cursor
233.            lcd.print(" "); // borro lo que estaba en la posicion de B
234.            lcd.setCursor(15, 0); // posicion del cursor
235.            lcd.print("A"); // Imprime en la posicion el lado A (es en el que está)
236.            Serial.println("LADO: A"); // Por puerto serie imprime el lado tambien en
    el que se encuentra
237.          }
238.          if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO B
239.          {
240.            lcd.setCursor(15, 0); // posicion del cursor
241.            lcd.print(" "); // borro lo que estaba en la posicion de B
242.            lcd.setCursor(15, 1); // posicion del cursor
243.            lcd.print("B"); // Imprime en la posicion el lado B (es en el que está)
244.            Serial.println("LADO: B"); // Por puerto serie imprime el lado tambien en
    el que se encuentra
245.          }
246.    //************************************** IMPRESION DE VALORES EN LCD
    **************************************//
247.          lcd.leftToRight(); // seteo del cursor
248.          lcd.setCursor(6, 0); // posicion del cursor
249.          lcd.print(CONT_LCD); // imprime el agua medida
250.          lcd.setCursor(10, 1); // posicion del cursor
251.          lcd.print(CONT); // imprime cantidad de cangilones
252.
253.          Serial.print("AGUA: "); // imprime por puerto serie el agua medida
254.          Serial.println(CONT_LCD); // valor del agua
255.
256.          Serial.print("TIEMPO: "); // imprime por puerto serie el tiempo entre medidas
257.          Serial.println(DIFERENCIA); // valor del tiempo entre medidas
258.          Serial.println(" "); // imprime un enter para separar los valores
259.
260.    //************************************** IMPRESION DE VALORES EN LCD
    **************************************//
261.          FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
262.        }
263.      }
264.    }
265.    //************************ FLAG ( INTERRUPCIÓN POR FLANCO )
    ***************************//
266.
267.
268.    //************************** SERVICIO DE INTERRUPCION  **********************//
269.    // CON ANTIREBOTE //
270.    // CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //
271.
272.    void AUMENTAR()
273.    {
274.      if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
275.      {
276.        INICIO_TIMER = millis();
277.        FLAG = 1;
278.      }
279.    }
280.    //*************************************************************************//
281.
282.    void SERIAL_EVENT()
283.    {
284.      while (Serial.available()) // lectura del teclado
285.      {
```

```
286.        int ESTADO = '0'; // 0 ASCII
287.
288.        ESTADO = Serial.read(); // La variable estado pasara a valer lo que detecte del
    teclado
289.
290.        if (ESTADO == '+') // cuando presione + iniciara la medición
291.        {
292.          FLAG4 = 1;
293.
294.          Serial.print("MEDICION INICIADA");
295.          Serial.println();
296.
297.          Serial.print("PRESIONE * PARA REINICIAR");
298.          Serial.println();
299.
300.          Serial.print("PRESIONE - PARA CONTINUAR");
301.          Serial.println();
302.          Serial.println();
303.        }
304.
305.        if (ESTADO == '-') // cuando presione - pausará la medición
306.        {
307.          FLAG4 = 0;
308.
309.          Serial.print("MEDICION PAUSADA");
310.          Serial.println();
311.
312.          Serial.print("PRESIONE + PARA CONTINUAR");
313.          Serial.println();
314.
315.          Serial.print("PRESIONE * PARA REINICIAR");
316.          Serial.println();
317.        }
318.
319.        if (ESTADO == '*')
320.        {
321.          Serial.print("AGUA: "); // imprime por puerto serie el agua medida
322.          Serial.println(CONT_LCD); // valor del agua
323.
324.          lcd.setCursor(5, 0);
325.          lcd.print("         ");
326.          lcd.setCursor(0, 0);
327.          lcd.print("AGUA: 0.00");
328.          lcd.setCursor(0, 1);
329.          lcd.print("00:00:00");
330.          lcd.setCursor(10, 1);
331.          lcd.print("0");
332.
333.          CONTADOR = 0;
334.          CONT = 0;
335.          CONT_LCD = 0;
336.
337.          SEGUNDOS_LCD = 0;
338.          MINUTOS_LCD = 0;
339.          HORAS_LCD = 0;
340.
341.          TIEMPO = 0;
342.          DIFERENCIA = 0;
343.
344.          FLAG = 0;
345.          FLAG2 = 0;
346.          FLAG4 = 0;
347.        }
348.      }
349.    }
350.
```

```
//****************************** TIMER 1 ******************************//
// POR DESBORDAMIENTO //
// TIMER MADRE DE 1s  //
// EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
// LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
ISR(TIMER1_OVF_vect)
{
  TCNT1 = 0xC2F7;

  if (FLAG4)
  {
    SEGUNDOS_LCD ++;
    FLAG2 = 1;
  }
}
//*****************************************************************//
```

Ensayos a la intemperie

Ensayo 20

El ensayo número 20 estuvo en la intemperie alrededor de una semana de su instalación formal, este sensó 837.40 cm^3, que realizando la conversión a agua llovida por metro cuadrado dió 27 mm de altura.

Ensayo 21

La medición que realizó el pluviómetro digital fue de 31,60 cm^3 y lo pesado en el puesto de recolección fue de 30,55 cm^3 teniendo un error de 1,05 cm^3.

PRUEBA7_CANGILON_TECHO

```
//********************** Liberias ************************************//
#include <EEPROM.h>
//********************** Liberias ************************************//


//********************** definicion de variables ***********************//
volatile float CONTADOR = 0;
volatile int CONT = 0;
volatile float CONT_LCD = 0;
float CTE;

unsigned long LIMITE = 10;
unsigned long INICIO_TIMER = 0;

int SEGUNDOS_LCD = 0;
int MINUTOS_LCD = 0;
int HORAS_LCD = 0;
int DIAS_LCD = 0;

int SEGUNDOS = 0;
int MINUTOS = 0;
```

```
int HORAS = 0;
int DIAS = 0;

long SEG = 0;
long MIN = 0;
long HRS = 0;

int TIEMPO = 0;
int TIEMPO2 = 0;
int TIEMPO3 = 0;

int DIFERENCIAS = 0;
int DIFERENCIAM = 0;
int DIFERENCIAH = 0;

bool FLAG = 0;
bool FLAG2 = 0;
bool FLAG3 = 0;
bool FLAG4 = 0;
//********************** definicion de variables ************************//


//********* ATENCION EEPROM ************//
//********float PROMEDIO = 5.5; ******//
float EEPROMEDIO;
//********* ATENCION EEPROM ************//


//************************************** SETUP
   ********************************************//

void setup()
{
// TIMER 1 //
// TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
// TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //

  CONTADOR = 0;
  CONT_LCD = 0;

  TCCR1A = 0;
  TCCR1B = 0;
  TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1

  TCNT1 = 0xC2F8; //F9E5 100 MS

  TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
//********************************** TIMER 1 *******************************//


//******************** Grabo en EEPROM el valor promedio ******************//

// EEPROM.put (EEPROMEDIO, PROMEDIO);

//******************** Grabo en EEPROM el valor promedio ******************//


//****************************** Inicializacion puerto serie
   *******************************//
  Serial.begin(9600);
```

```
80. //****************************** Inicializacion puerto serie
    *******************************//
81.
82.
83. //****************************** SERVICIO DE INTERRUPCIÓN
    *******************************//
84.
85.   pinMode(2, INPUT_PULLUP);
86.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
87.
88. //****************************** SERVICIO DE INTERRUPCIÓN
    *******************************//
89.
90.
91. //****************************** inicializacion de variables
    *******************************//
92.   FLAG = false;
93.   FLAG2 = false;
94.   FLAG3 = true;
95.   FLAG4 = false;
96.
97.   CTE = 7.90;
98. //****************************** inicializacion de variables
    *******************************//
99.
100.
101.   //************************ INICIALIZACION DE PINES DEL LCD
    ************************//
102.
103.     Serial.print("PRESIONE + PARA INICIAR");
104.     Serial.println();
105.
106.     Serial.print("PRESIONE - PARA PAUSAR");
107.     Serial.println();
108.
109.     Serial.print("PRESIONE * PARA REINICIAR");
110.     Serial.println();
111.     Serial.println();
112.
113.   }
114.
115.   //*****************************************************************************
    *********************//
116.
117.
118.   //****************************************** VOID LOOP
    ***************************************//
119.   void loop()
120.   {
121.     SERIAL_EVENT();
122.
123.   //************************ SETEO DE MINUTOS
    ************************************//
124.     if (SEGUNDOS_LCD == 60) // SI ALCANZA LOS 60 SEGUNDOS AUMENTA EN 1 EL
    MINUTO
125.     {
126.       MINUTOS_LCD ++; //aumento en 1 la cantidad de minutos
127.       SEGUNDOS_LCD = 0; // resetea la variable segundos
128.     }
129.   //************************ SETEO DE MINUTOS
    ************************************//
```

```
130.
131.
132.   //************************ SETEO DE HORAS
   ************************************//
133.     if (MINUTOS_LCD == 60) // SI ALCANZA LOS 60 MINUTOS AUMENTA EN 1 LA HORA
134.     {
135.       HORAS_LCD ++; //aumento en 1 la cantidad de horas
136.       MINUTOS_LCD = 0; // reseteo la variable minutos
137.     }
138.   //************************ SETEO DE HORAS
   ************************************//
139.
140.
141.   //************************ RESETEO DE HORAS
   ************************************//
142.     if (HORAS_LCD > 23) // SI ALCANZA UN VALOR MAYOR A 24 RESETEA LA VARIABLE
   HORAS Y RESETEA EL DISPLAY
143.     {
144.       DIAS_LCD ++;
145.       HORAS_LCD = 0; // reseteo la variable en horas
146.     }
147.   //************************ RESETEO DE HORAS
   ************************************//
148.
149.
150.   //************************ FLAG ( INTERRUPCIÓN POR FLANCO )
   ***************************//
151.   if(FLAG4)
152.   {
153.       if (FLAG == 1) // FLAG del servicio de interrupción
154.       {
155.         FLAG3 =! FLAG3; // cambia de estado el FLAG3 cada interrupción
156.         CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN
   TODOS LOS DEMAS
157.         CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA
   DE 1 CANGILON 7.9CM3
158.
159.         DIAS = DIAS_LCD;
160.         HORAS = HORAS_LCD;
161.         MINUTOS = MINUTOS_LCD;
162.         SEGUNDOS = SEGUNDOS_LCD;
163.
164.         DIFERENCIAH = HORAS_LCD - TIEMPO3;
165.         DIFERENCIAM = MINUTOS_LCD - TIEMPO2;
166.         DIFERENCIAS = SEGUNDOS_LCD - TIEMPO; // ESTE ES EL CALCULO ENTRE EL
   TIEMPO DE LA PRIMER INTERRUPCION Y EL QUE LE SIGUE, PARA MOSTRAR EN PUERTO
   SERIE CUANTO TARDA
167.
168.         TIEMPO3 = HORAS_LCD;
169.         TIEMPO2 = MINUTOS_LCD;
170.         TIEMPO = SEGUNDOS_LCD; // MAPEA LOS SEGUNDOS EN LA VARIABLE TIEMPO
   PARA REPLAZAR Y PODER HACER EL CALCULO DE DIFERENCIA EN LA PROXIMA INTERRUPCIÓN
171.
172.         if (DIFERENCIAS>=0 && DIFERENCIAM>=0 && DIFERENCIAH>=0)
173.         {
174.           Serial.print("TIEMPO GENERAL: "); // imprime por puerto serie el
   tiempo entre medidas
175.           Serial.print(DIAS);
176.           Serial.print(":");
177.           Serial.print(DIFERENCIAH); // valor del tiempo entre medidas
178.           Serial.print(":");
```

```
179.           Serial.print(DIFERENCIAM);
180.           Serial.print(":");
181.           Serial.println(DIFERENCIAS);
182.           Serial.println(" ");
183.         }
184.
185.         if(DIFERENCIAH < 0)
186.         {
187.           SEG = DIFERENCIAH * 3600;
188.           SEG = SEG + DIFERENCIAM * 60;
189.           SEG = SEG + DIFERENCIAS;
190.         }
191.
192.         if(DIFERENCIAM < 0)
193.         {
194.           SEG = DIFERENCIAS + DIFERENCIAH * 3600;
195.           SEG = SEG + DIFERENCIAM * 60;
196.         }
197.
198.         if(DIFERENCIAS < 0)
199.         {
200.           SEG = DIFERENCIAM * 60;
201.           SEG = SEG + DIFERENCIAH * 3600;
202.           SEG = SEG + DIFERENCIAS;
203.         }
204.
205.         if (SEG < 0)
206.         {
207.           SEG = SEG * -1;
208.         }
209.
210.         if (SEG > 3599)
211.         {
212.           HRS = SEG / 3600;
213.           SEG = SEG - HRS * 3600;
214.         }
215.
216.         if (SEG > 59)
217.         {
218.           MIN = SEG / 60;
219.           SEG = SEG - MIN * 60;
220.         }
221.
222.         if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ
    QUE INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO A
223.         {
224.           Serial.println("LADO: A"); // Por puerto serie imprime el lado
    tambien en el que se encuentra
225.         }
226.         if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ
    QUE INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO B
227.         {
228.           Serial.println("LADO: B"); // Por puerto serie imprime el lado
    tambien en el que se encuentra
229.         }
230.    //************************************* IMPRESION DE VALORES EN LCD
    *************************************//
231.         Serial.print("AGUA: "); // imprime por puerto serie el agua medida
232.         Serial.println(CONT_LCD); // valor del agua
233.
```

```
234.          Serial.print("TIEMPO GENERAL: "); // imprime por puerto serie el
   tiempo entre medidas
235.          Serial.print(DIAS);
236.          Serial.print(":");
237.          Serial.print(HORAS); // valor del tiempo entre medidas
238.          Serial.print(":");
239.          Serial.print(MINUTOS);
240.          Serial.print(":");
241.          Serial.println(SEGUNDOS);
242.          Serial.println(" ");
243.    //************************************* IMPRESION DE VALORES EN LCD
   ************************************//
244.          FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
245.        }
246.      }
247.    }
248.    //********************** FLAG ( INTERRUPCIÓN POR FLANCO )
   *************************//
249.
250.
251.    //************************** SERVICIO DE INTERRUPCION
   *********************//
252.    // CON ANTIREBOTE //
253.    // CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //
254.
255.    void AUMENTAR()
256.    {
257.      if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
258.      {
259.        INICIO_TIMER = millis();
260.        FLAG = 1;
261.      }
262.    }
263.    //*************************************************************************//
264.
265.    void SERIAL_EVENT()
266.    {
267.      while (Serial.available()) // lectura del teclado
268.      {
269.        int ESTADO = '0'; // 0 ASCII
270.
271.        ESTADO = Serial.read(); // La variable estado pasara a valer lo que
   detecte del teclado
272.
273.        if (ESTADO == '+') // cuando presione + iniciara la medición
274.        {
275.          FLAG4 = 1;
276.
277.          Serial.print("MEDICION INICIADA");
278.          Serial.println();
279.
280.          Serial.print("PRESIONE * PARA REINICIAR");
281.          Serial.println();
282.
283.          Serial.print("PRESIONE - PARA CONTINUAR");
284.          Serial.println();
285.          Serial.println();
286.        }
287.
288.        if (ESTADO == '-') // cuando presione - pausará la medición
289.        {
```

```
290.          FLAG4 = 0;
291.
292.          Serial.print("MEDICION PAUSADA");
293.          Serial.println();
294.
295.          Serial.print("PRESIONE + PARA CONTINUAR");
296.          Serial.println();
297.
298.          Serial.print("PRESIONE * PARA REINICIAR");
299.          Serial.println();
300.        }
301.
302.        if (ESTADO == '*')
303.        {
304.          Serial.print("AGUA: "); // imprime por puerto serie el agua medida
305.          Serial.println(CONT_LCD); // valor del agua
306.
307.          CONTADOR = 0;
308.          CONT = 0;
309.          CONT_LCD = 0;
310.
311.          SEGUNDOS_LCD = 0;
312.          MINUTOS_LCD = 0;
313.          HORAS_LCD = 0;
314.          DIAS_LCD = 0;
315.
316.          TIEMPO = 0;
317.          TIEMPO2 = 0;
318.          TIEMPO3 = 0;
319.
320.          DIFERENCIAS = 0;
321.          DIFERENCIAM = 0;
322.          DIFERENCIAH = 0;
323.
324.          FLAG = 0;
325.          FLAG2 = 0;
326.          FLAG4 = 0;
327.        }
328.      }
329.    }
330.
331.    //******************************** TIMER 1
       ******************************//
332.    // POR DESBORDAMIENTO //
333.    // TIMER MADRE DE 1s  //
334.    // EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
335.    // LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
336.    ISR(TIMER1_OVF_vect)
337.    {
338.      TCNT1 = 0xC2F7; // F9E5 100 ms
339.
340.      if (FLAG4)
341.      {
342.        SEGUNDOS_LCD ++;
343.        FLAG2 = 1;
344.      }
345.    }
346.    //*************************************************************************//
```

Ensayo con software 8

Objetivo: Ya alcanzado los objetivos establecidos en la lectura y caracterización de datos, la etapa siguiente tuvo por objetivo almacenar dichos datos en una tarjeta-SD, para su posterior extracción y análisis. Manteniendo las bases de programas anteriores se incorporó la función de almacenar la información.

Resultado: Se logró alcanzar correctamente el objetivo, pero se notó un error en la visualización de los tiempos entre interrupciones, se procederá en los próximos ensayos a corregir tal inconveniente.

Ensayo 22

El pluviómetro digital recolectó 518,22 cm^3 y la medición que realizó fue de 553 cm^3, teniendo un error 34,78 cm^3.

PRUEBA8_CANGILON_TECHO_SD

```
1.
2.  //********************** Liberias ************************************//
3.  #include <EEPROM.h>
4.  #include <SPI.h>
5.  #include <SD.h>
6.  //********************** Liberias ************************************//
7.
8.  File myFile;
9.
10. //********************** definicion de variables ************************//
11. volatile float CONTADOR = 0;
12. volatile int CONT = 0;
13. volatile float CONT_LCD = 0;
14. float CTE;
15.
16. unsigned long LIMITE = 10;
17. unsigned long INICIO_TIMER = 0;
18.
19. int SEGUNDOS_LCD = 0;
20. int MINUTOS_LCD = 0;
21. int HORAS_LCD = 0;
22. int DIAS_LCD = 0;
23.
24. int SEGUNDOS = 0;
25. int MINUTOS = 0;
26. int HORAS = 0;
27. int DIAS = 0;
28.
29. int TIEMPO = 0;
30. int TIEMPO2 = 0;
31. int TIEMPO3 = 0;
32.
33. int DIFERENCIA = 0;
34. int DIFERENCIA2 = 0;
35. int DIFERENCIA3 = 0;
36.
37. bool FLAG = 0;
38. bool FLAG2 = 0;
39. bool FLAG3 = 0;
40. bool FLAG4 = 0;
41. //********************** definicion de variables ************************//
42.
43.
44. //********* ATENCION EEPROM ************//
45. //********float PROMEDIO = 5.5; ******//
```

```
46. float EEPROMEDIO;
47. //********* ATENCION EEPROM ************//
48.
49.
50. //************************************** SETUP ********************************************//
51.
52. void setup()
53. {
54.   Serial.begin(9600);
55.
56.   // TIMER 1 //
57.   // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
58.   // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
59.
60.   CONTADOR = 0;
61.   CONT_LCD = 0;
62.
63.   TCCR1A = 0;
64.   TCCR1B = 0;
65.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
66.
67.   TCNT1 = 0xC2F8; //F9E5 100 MS
68.
69.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
70.   //********************************** TIMER 1 *******************************//
71.
72.
73.   //******************** Grabo en EEPROM el valor promedio ******************//
74.
75.   // EEPROM.put (EEPROMEDIO, PROMEDIO);
76.
77.   //******************** Grabo en EEPROM el valor promedio ******************//
78.
79.
80.   //******************************** Inicializacion puerto serie ******************************//
81.
82.   //******************************** Inicializacion puerto serie ******************************//
83.
84.
85.   //******************************** SERVICIO DE INTERRUPCIÓN *********************************//
86.
87.   pinMode(2, INPUT_PULLUP);
88.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
89.
90.   //******************************** SERVICIO DE INTERRUPCIÓN *********************************//
91.
92.   //******************************** inicializacion de variables ******************************//
93.   FLAG = false;
94.   FLAG2 = false;
95.   FLAG3 = true;
96.   FLAG4 = false;
97.
98.   CTE = 7.90;
99.   //******************************** inicializacion de variables ******************************//
100.
101.  Serial.print("INICIANDO SD ...");
102.
103.  if (!SD.begin(4)) {
104.    Serial.println("ALGO FALLO!");
105.    while (1);
106.  }
107.  Serial.println("TODO JOYA");
108.
109.  SD.remove("PRUEBA.txt");
110.
111.  pinMode(13, OUTPUT);
112.
113.  //************************ INICIALIZACION DE PINES DEL LCD ***********************//
114.
115.  myFile = SD.open("PRUEBA.txt", FILE_WRITE);
116.  if (myFile)
117.  {
118.    Serial.println("PRESIONE + PARA INICIAR");
119.    Serial.println("PRESIONE - PARA PAUSAR");
120.    Serial.println("PRESIONE * PARA REINICIAR");
121.    Serial.println();
122.
123.    myFile.println("PRESIONE + PARA INICIAR");
```

```
124.     myFile.println("PRESIONE - PARA PAUSAR");
125.     myFile.println("PRESIONE * PARA REINICIAR");
126.     myFile.println();
127.     myFile.close();
128.   }
129.   else
130.   {
131.     Serial.println("ERROR ABRIENDO PRUEBA.txt");
132.   }
133. }
134.
135. //*************************************************************************************************//
136.
137.
138. //********************************************* VOID LOOP *****************************************//
139. void loop()
140. {
141.   SERIAL_EVENT();
142.
143.   //*********************** SETEO DE MINUTOS ***********************************//
144.   if (SEGUNDOS_LCD == 60) // SI ALCANZA LOS 60 SEGUNDOS AUMENTA EN 1 EL MINUTO
145.   {
146.     MINUTOS_LCD ++; //aumento en 1 la cantidad de minutos
147.     SEGUNDOS_LCD = 0; // resetea la variable segundos
148.   }
149.   //*********************** SETEO DE MINUTOS ***********************************//
150.
151.
152.   //*********************** SETEO DE HORAS ***********************************//
153.   if (MINUTOS_LCD == 60) // SI ALCANZA LOS 60 MINUTOS AUMENTA EN 1 LA HORA
154.   {
155.     HORAS_LCD ++; //aumento en 1 la cantidad de horas
156.     MINUTOS_LCD = 0; // reseteo la variable minutos
157.   }
158.   //*********************** SETEO DE HORAS ***********************************//
159.
160.
161.   //*********************** RESETEO DE HORAS ***********************************//
162.   if (HORAS_LCD > 23) // SI ALCANZA UN VALOR MAYOR A 24 RESETEA LA VARIABLE HORAS Y RESETEA EL DISPLAY
163.   {
164.     DIAS_LCD ++;
165.     HORAS_LCD = 0; // reseteo la variable en horas
166.   }
167.   //*********************** RESETEO DE HORAS ***********************************//
168.
169.
170.   //*********************** FLAG ( INTERRUPCIÓN POR FLANCO ) ***************************//
171.   if (FLAG4)
172.   {
173.     if (FLAG == 1) // FLAG del servicio de interrupción
174.     {
175.       FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción
176.       CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS DEMAS
177.       CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1 CANGILON 7.9CM3
178.
179.       DIAS = DIAS_LCD;
180.       HORAS = HORAS_LCD;
181.       MINUTOS = MINUTOS_LCD;
182.       SEGUNDOS = SEGUNDOS_LCD;
183.
184.       DIFERENCIA3 = HORAS_LCD - TIEMPO3;
185.       DIFERENCIA2 = MINUTOS_LCD - TIEMPO2;
186.       DIFERENCIA = SEGUNDOS_LCD - TIEMPO; // ESTE ES EL CALCULO ENTRE EL TIEMPO DE LA PRIMER INTERRUPCION
    Y EL QUE LE SIGUE, PARA MOSTRAR EN PUERTO SERIE CUANTO TARDA
187.
188.       TIEMPO3 = HORAS_LCD;
189.       TIEMPO2 = MINUTOS_LCD;
190.       TIEMPO = SEGUNDOS_LCD; // MAPEA LOS SEGUNDOS EN LA VARIABLE TIEMPO PARA REPLAZAR Y PODER HACER EL
    CALCULO DE DIFERENCIA EN LA PROXIMA INTERRUPCIÓN
191.
192.       if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO A
193.       {
194.         Serial.println("LADO: A"); // Por puerto serie imprime el lado tambien en el que se encuentra
195.       }
196.       if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO B
197.       {
```

```
198.         Serial.println("LADO: B"); // Por puerto serie imprime el lado tambien en el que se encuentra
199.       }
200.       //************************************* IMPRESION DE VALORES EN LCD
    *************************************//
201.
202.       myFile = SD.open("PRUEBA.txt", FILE_WRITE);
203.       if (myFile)
204.       {
205.         Serial.print("AGUA: "); // imprime por puerto serie el agua medida
206.         Serial.println(CONT_LCD); // valor del agua
207.
208.         Serial.print("INTERRUPCIÓN: "); // imprime por puerto serie el tiempo entre medidas
209.         Serial.print(DIAS_LCD);
210.         Serial.print(":");
211.         Serial.print(DIFERENCIA3); // valor del tiempo entre medidas
212.         Serial.print(":");
213.         Serial.print(DIFERENCIA2);
214.         Serial.print(":");
215.         Serial.println(DIFERENCIA);
216.
217.         Serial.print("TIEMPO GENERAL: "); // imprime por puerto serie el tiempo entre medidas
218.         Serial.print(DIAS);
219.         Serial.print(":");
220.         Serial.print(HORAS); // valor del tiempo entre medidas
221.         Serial.print(":");
222.         Serial.print(MINUTOS);
223.         Serial.print(":");
224.         Serial.println(SEGUNDOS);
225.         Serial.println();
226.
227.         digitalWrite(13, 1);
228.
229.         myFile.print("AGUA: "); // imprime por puerto serie el agua medida
230.         myFile.println(CONT_LCD); // valor del agua
231.
232.         myFile.print("INTERRUPCIÓN: "); // imprime por puerto serie el tiempo entre medidas
233.         myFile.print(DIAS_LCD);
234.         myFile.print(":");
235.         myFile.print(DIFERENCIA3); // valor del tiempo entre medidas
236.         myFile.print(":");
237.         myFile.print(DIFERENCIA2);
238.         myFile.print(":");
239.         myFile.println(DIFERENCIA);
240.
241.         myFile.print("TIEMPO GENERAL: "); // imprime por puerto serie el tiempo entre medidas
242.         myFile.print(DIAS);
243.         myFile.print(":");
244.         myFile.print(HORAS); // valor del tiempo entre medidas
245.         myFile.print(":");
246.         myFile.print(MINUTOS);
247.         myFile.print(":");
248.         myFile.println(SEGUNDOS);
249.         myFile.println();
250.         myFile.close();
251.
252.         digitalWrite(13, 0);
253.       }
254.       //************************************* IMPRESION DE VALORES EN LCD
    *************************************//
255.       FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
256.     }
257.   }
258. }
259. //*********************** FLAG ( INTERRUPCIÓN POR FLANCO ) **************************//
260.
261.
262. //************************** SERVICIO DE INTERRUPCION  ********************//
263. // CON ANTIREBOTE //
264. // CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //
265.
266. void AUMENTAR()
267. {
268.   if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
269.   {
270.     INICIO_TIMER = millis();
271.     FLAG = 1;
272.   }
273. }
```

```
274. //*************************************************************************//
275.
276. void SERIAL_EVENT()
277. {
278.   while (Serial.available()) // lectura del teclado
279.   {
280.     int ESTADO = '0'; // 0 ASCII
281.
282.     ESTADO = Serial.read(); // La variable estado pasara a valer lo que detecte del teclado
283.
284.     if (ESTADO == '+') // cuando presione + iniciara la medición
285.     {
286.       myFile = SD.open("PRUEBA.txt", FILE_WRITE);
287.       if (myFile)
288.       {
289.         Serial.print("MEDICION INICIADA");
290.         Serial.println();
291.
292.         Serial.print("PRESIONE * PARA REINICIAR");
293.         Serial.println();
294.
295.         Serial.print("PRESIONE - PARA CONTINUAR");
296.         Serial.println();
297.         Serial.println();
298.
299.         digitalWrite(13, 1);
300.
301.         myFile.print("MEDICION INICIADA");
302.         myFile.println();
303.
304.         myFile.print("PRESIONE * PARA REINICIAR");
305.         myFile.println();
306.
307.         myFile.print("PRESIONE - PARA CONTINUAR");
308.         myFile.println();
309.         myFile.println();
310.         myFile.close();
311.
312.         digitalWrite(13, 0);
313.       }
314.       FLAG4 = 1;
315.     }
316.
317.     if (ESTADO == '-') // cuando presione - pausará la medición
318.     {
319.       myFile = SD.open("PRUEBA.txt", FILE_WRITE);
320.       if (myFile)
321.       {
322.         digitalWrite(13, 1);
323.         Serial.print("MEDICION PAUSADA");
324.         Serial.println();
325.
326.         Serial.print("PRESIONE + PARA CONTINUAR");
327.         Serial.println();
328.
329.         Serial.print("PRESIONE * PARA REINICIAR");
330.         Serial.println();
331.
332.         myFile.print("MEDICION PAUSADA");
333.         myFile.println();
334.
335.         myFile.print("PRESIONE + PARA CONTINUAR");
336.         myFile.println();
337.
338.         myFile.print("PRESIONE * PARA REINICIAR");
339.         myFile.println();
340.         myFile.close();
341.         digitalWrite(13, 0);
342.       }
343.       FLAG4 = 0;
344.     }
345.
346.     if (ESTADO == '*')
347.     {
348.       myFile = SD.open("PRUEBA.txt", FILE_WRITE);
349.       if (myFile)
350.       {
351.         Serial.print("MEDICION FINALIZADA... REINICIANDO");
```

```
352.         myFile.print("MEDICION FINALIZADA... REINICIANDO");
353.         myFile.close();
354.       }
355.
356.       CONTADOR = 0;
357.       CONT = 0;
358.       CONT_LCD = 0;
359.
360.       SEGUNDOS_LCD = 0;
361.       MINUTOS_LCD = 0;
362.       HORAS_LCD = 0;
363.       DIAS_LCD = 0;
364.
365.       TIEMPO = 0;
366.       TIEMPO2 = 0;
367.       TIEMPO3 = 0;
368.
369.       DIFERENCIA = 0;
370.       DIFERENCIA2 = 0;
371.       DIFERENCIA3 = 0;
372.
373.       FLAG = 0;
374.       FLAG2 = 0;
375.       FLAG4 = 0;
376.     }
377.   }
378. }
379.
380. //******************************** TIMER 1 ******************************//
381. // POR DESBORDAMIENTO //
382. // TIMER MADRE DE 1s  //
383. // EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
384. // LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
385. ISR(TIMER1_OVF_vect)
386. {
387.   TCNT1 = 0xC2F7; // F9E5 100 ms
388.
389.   if (FLAG4)
390.   {
391.     SEGUNDOS_LCD ++;
392.     FLAG2 = 1;
393.   }
394. }
395. //***************************************************************************//
```

Ensayo con software 9

Objetivo: Solucionar efectivamente la problemática que hay en la visualización del tiempo, ya que este aparece con símbolos negativos.

Resultado: Se logra ver de manera correcta los segundos, minutos y horas, de igual manera se espera simplificar el algoritmo del programa para facilitar la compresión.

PRUEBA9_CANGILON_TECHO_SD

```
1.  //********************** Liberias ***************************************//
2.  #include <EEPROM.h>
3.  #include <SPI.h>
4.  #include <SD.h>
5.  //********************** Liberias ***************************************//
6.
7.  File myFile;
8.
9.  //********************** definicion de variables ************************//
10. volatile float CONTADOR = 0;
11. volatile int CONT = 0;
```

```
12. volatile float CONT_LCD = 0;
13. float CTE;
14.
15. unsigned long LIMITE = 10;
16. unsigned long INICIO_TIMER = 0;
17.
18. int SEGUNDOS_LCD = 0;
19. int MINUTOS_LCD = 0;
20. int HORAS_LCD = 0;
21. int DIAS_LCD = 0;
22.
23. int SEGUNDOS = 0;
24. int MINUTOS = 0;
25. int HORAS = 0;
26. int DIAS = 0;
27.
28. int TIEMPO = 0;
29. int TIEMPO2 = 0;
30. int TIEMPO3 = 0;
31.
32. long SEG = 0;
33. long MIN = 0;
34. long HRS = 0;
35.
36. int CASE = 0;
37.
38. int DIFERENCIAS = 0;
39. int DIFERENCIAM = 0;
40. int DIFERENCIAH = 0;
41.
42. bool FLAG = 0;
43. bool FLAG2 = 0;
44. bool FLAG3 = 0;
45. bool FLAG4 = 0;
46. //********************* definicion de variables ************************//
47.
48.
49. //********* ATENCION EEPROM ************//
50.
51. float EEPROMEDIO;
52.
53. //********* ATENCION EEPROM ************//
54.
55.
56. //***************************************** SETUP ***********************************************//
57.
58. void setup()
59. {
60.   // TIMER 1 //
61.   // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
62.   // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
63.
64.   CONTADOR = 0;
65.   CONT_LCD = 0;
66.
67.   TCCR1A = 0;
68.   TCCR1B = 0;
69.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
70.
71.   TCNT1 = 0xC2F8; //F9E5 100 MS
72.
73.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
74.   //********************************* TIMER 1 *******************************//
75.
76.
77.   //******************** Grabo en EEPROM el valor promedio ******************//
78.
79.   // EEPROM.put (EEPROMEDIO, PROMEDIO);
80.
81.   //******************** Grabo en EEPROM el valor promedio ******************//
82.
83.
84.   //****************************** Inicializacion puerto serie ******************************//
85.   Serial.begin(9600);
86.   //****************************** Inicializacion puerto serie ******************************//
87.
88.
89.   //****************************** SERVICIO DE INTERRUPCIÓN ******************************//
```

```
90.
91.    pinMode(2, INPUT_PULLUP);
92.    attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
93.
94.    //******************************** SERVICIO DE INTERRUPCIÓN ********************************//
95.
96.
97.    //******************************** inicializacion de variables ******************************//
98.    FLAG = false;
99.    FLAG2 = false;
100.   FLAG3 = true;
101.   FLAG4 = false;
102.
103.
104.   CTE = 7.90;
105.
106.   //******************************** inicializacion de variables ******************************//
107.
108.   Serial.print("INICIANDO SD ...");
109.
110.   if (!SD.begin(4)) {
111.     Serial.println("ALGO FALLO!");
112.     while (1);
113.   }
114.   Serial.println("TODO JOYA");
115.
116.   SD.remove("ENSAYO.txt");
117.
118.   pinMode(13, OUTPUT);
119.
120.   //************************ INICIALIZACION DE PINES DEL LCD ************************//
121.
122.   myFile = SD.open("ENSAYO.txt", FILE_WRITE);
123.   if (myFile)
124.   {
125.     Serial.println("PRESIONE + PARA INICIAR");
126.     Serial.println("PRESIONE - PARA PAUSAR");
127.     Serial.println("PRESIONE * PARA REINICIAR");
128.     Serial.println();
129.
130.     digitalWrite(13, 1);
131.
132.     myFile.println("PRESIONE + PARA INICIAR");
133.     myFile.println("PRESIONE - PARA PAUSAR");
134.     myFile.println("PRESIONE * PARA REINICIAR");
135.     myFile.println();
136.     myFile.close();
137.     digitalWrite(13, 0);
138.   }
139.   else
140.   {
141.     Serial.println("ERROR ABRIENDO ENSAYO.txt");
142.   }
143. }
144.
145. //*************************************************************************************************//
146.
147.
148. //********************************************** VOID LOOP ************************************//
149. void loop()
150. {
151.   SERIAL_EVENT();
152.
153.   //************************ SETEO DE MINUTOS ***********************************//
154.   if (SEGUNDOS_LCD == 60) // SI ALCANZA LOS 60 SEGUNDOS AUMENTA EN 1 EL MINUTO
155.   {
156.     MINUTOS_LCD ++; //aumento en 1 la cantidad de minutos
157.     SEGUNDOS_LCD = 0; // resetea la variable segundos
158.   }
159.   //************************ SETEO DE MINUTOS ***********************************//
160.
161.   //************************ SETEO DE HORAS ***********************************//
162.   if (MINUTOS_LCD == 60) // SI ALCANZA LOS 60 MINUTOS AUMENTA EN 1 LA HORA
163.   {
164.     HORAS_LCD ++; //aumento en 1 la cantidad de horas
165.     MINUTOS_LCD = 0; // reseteo la variable minutos
166.   }
167.   //************************ SETEO DE HORAS ***********************************//
```

```
168.
169.   //************************ RESETEO DE HORAS ***********************************//
170.   if (HORAS_LCD > 23) // SI ALCANZA UN VALOR MAYOR A 24 RESETEA LA VARIABLE HORAS Y RESETEA EL DISPLAY
171.   {
172.     DIAS_LCD ++;
173.     HORAS_LCD = 0; // reseteo la variable en horas
174.   }
175.   //************************ RESETEO DE HORAS ***********************************//
176.
177.   //************************ FLAG ( INTERRUPCIÓN POR FLANCO ) *************************//
178.   if (FLAG4)
179.   {
180.     if (FLAG == 1) // FLAG del servicio de interrupción
181.     {
182.       FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción
183.       CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS DEMAS
184.       CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1 CANGILON 7.9CM3
185.
186.       DIAS = DIAS_LCD;
187.       HORAS = HORAS_LCD;
188.       MINUTOS = MINUTOS_LCD;
189.       SEGUNDOS = SEGUNDOS_LCD;
190.
191.       DIFERENCIAH = HORAS_LCD - TIEMPO3;
192.       DIFERENCIAM = MINUTOS_LCD - TIEMPO2;
193.       DIFERENCIAS = SEGUNDOS_LCD - TIEMPO; // ESTE ES EL CALCULO ENTRE EL TIEMPO DE LA PRIMER INTERRUPCION
    Y EL QUE LE SIGUE, PARA MOSTRAR EN PUERTO SERIE CUANTO TARDA
194.
195.       TIEMPO3 = HORAS_LCD;
196.       TIEMPO2 = MINUTOS_LCD;
197.       TIEMPO = SEGUNDOS_LCD; // MAPEA LOS SEGUNDOS EN LA VARIABLE TIEMPO PARA REPLAZAR Y PODER HACER EL
    CALCULO DE DIFERENCIA EN LA PROXIMA INTERRUPCIÓN
198.
199.       if (DIFERENCIAS >= 0 && DIFERENCIAM >= 0 && DIFERENCIAH >= 0)
200.       {
201.         CASE = 1;
202.       }
203.       if (DIFERENCIAS < 0 && DIFERENCIAM >= 0 && DIFERENCIAH >= 0)
204.       {
205.         CASE = 2;
206.       }
207.
208.       if (DIFERENCIAS >= 0 && DIFERENCIAM < 0 && DIFERENCIAH >= 0)
209.       {
210.         CASE = 3;
211.       }
212.
213.       if (DIFERENCIAS >= 0 && DIFERENCIAM >= 0 && DIFERENCIAH < 0)
214.       {
215.         CASE = 4;
216.       }
217.
218.       if (DIFERENCIAS < 0 && DIFERENCIAM < 0 && DIFERENCIAH >= 0)
219.       {
220.         CASE = 5;
221.       }
222.
223.       if (DIFERENCIAS < 0 && DIFERENCIAM >= 0 && DIFERENCIAH < 0)
224.       {
225.         CASE = 6;
226.       }
227.
228.       if (DIFERENCIAS >= 0 && DIFERENCIAM < 0 && DIFERENCIAH < 0)
229.       {
230.         CASE = 7;
231.       }
232.
233.       if (DIFERENCIAS < 0 && DIFERENCIAM < 0 && DIFERENCIAH < 0)
234.       {
235.         CASE = 8;
236.       }
237.
238.       if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO A
239.       {
240.         Serial.println("LADO: A"); // Por puerto serie imprime el lado tambien en el que se encuentra
241.       }
```

```
242.      if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
      MARCAR SI ESTA EN EL LADO B
243.      {
244.        Serial.println("LADO: B"); // Por puerto serie imprime el lado tambien en el que se encuentra
245.      }
246.
247.      Serial.print("AGUA: "); // imprime por puerto serie el agua medida
248.      Serial.println(CONT_LCD); // valor del agua
249.
250.      switch (CASE) {
251.        case 1:
252.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
253.          if (myFile)
254.          {
255.            Serial.print("INTERRUPCIÓN: "); // imprime por puerto serie el tiempo entre medidas
256.            Serial.print(DIFERENCIAH); // valor del tiempo entre medidas
257.            Serial.print(":");
258.            Serial.print(DIFERENCIAM);
259.            Serial.print(":");
260.            Serial.println(DIFERENCIAS);
261.
262.            digitalWrite(13, 1);
263.            myFile.print("INTERRUPCIÓN: "); // imprime por puerto serie el tiempo entre medidas
264.            myFile.print(DIFERENCIAH); // valor del tiempo entre medidas
265.            myFile.print(":");
266.            myFile.print(DIFERENCIAM);
267.            myFile.print(":");
268.            myFile.println(DIFERENCIAS);
269.            myFile.close();
270.            digitalWrite(13, 0);
271.          }
272.          break;
273.        case 2:
274.          SEG = DIFERENCIAH * 3600;
275.          SEG = SEG + DIFERENCIAM * 60;
276.          SEG = SEG + DIFERENCIAS;
277.          HRS = SEG / 3600;
278.          SEG = SEG - HRS * 3600;
279.          MIN = SEG / 60;
280.          SEG = SEG - MIN * 60;
281.
282.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
283.          if (myFile)
284.          {
285.            Serial.print("INTERRUPCIÓN: ");
286.            Serial.print(HRS);
287.            Serial.print(":");
288.            Serial.print(MIN);
289.            Serial.print(":");
290.            Serial.println(SEG);
291.
292.            digitalWrite(13, 1);
293.            myFile.print("INTERRUPCIÓN: ");
294.            myFile.print(HRS);
295.            myFile.print(":");
296.            myFile.print(MIN);
297.            myFile.print(":");
298.            myFile.println(SEG);
299.            myFile.close();
300.            digitalWrite(13, 0);
301.          }
302.
303.          TIEMPO3 = HORAS_LCD;
304.          TIEMPO2 = MINUTOS_LCD;
305.          TIEMPO = SEGUNDOS_LCD;
306.          break;
307.        case 3:
308.          SEG = DIFERENCIAH * 3600;
309.          SEG = SEG + DIFERENCIAM * 60;
310.          MIN = SEG / 60;
311.          MIN = MIN * -1;
312.          SEG = SEG + DIFERENCIAS;
313.          SEG = SEG + MIN * 60;
314.          SEG = SEG * -1;
315.
316.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
317.          if (myFile)
318.          {
```

```
            Serial.print("INTERRUPCIÓN: ");
            Serial.print(DIFERENCIAH);
            Serial.print(":");
            Serial.print(MIN);
            Serial.print(":");
            Serial.println(SEG);

            digitalWrite(13, 1);
            myFile.print("INTERRUPCIÓN: ");
            myFile.print(DIFERENCIAH);
            myFile.print(":");
            myFile.print(MIN);
            myFile.print(":");
            myFile.println(SEG);
            myFile.close();
            digitalWrite(13, 0);
          }


          TIEMPO3 = HORAS_LCD;
          TIEMPO2 = MINUTOS_LCD;
          TIEMPO = SEGUNDOS_LCD;
          break;
        case 4:
          SEG = DIFERENCIAH * 3600;
          SEG = SEG + DIFERENCIAM * 60;
          SEG = SEG + DIFERENCIAS;
          HRS = SEG / 3600;
          HRS = HRS * -1;
          SEG = SEG + HRS * 3600;
          MIN = SEG / 60;
          MIN = MIN * -1;
          SEG = SEG + MIN;
          SEG = SEG * -1;

          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
          if (myFile)
          {
            Serial.print("INTERRUPCIÓN: ");
            Serial.print(HRS);
            Serial.print(":");
            Serial.print(MIN);
            Serial.print(":");
            Serial.println(SEG);

            digitalWrite(13, 1);
            myFile.print("INTERRUPCIÓN: ");
            myFile.print(HRS);
            myFile.print(":");
            myFile.print(MIN);
            myFile.print(":");
            myFile.println(SEG);
            myFile.close();
            digitalWrite(13, 0);
          }

          TIEMPO3 = HORAS_LCD;
          TIEMPO2 = MINUTOS_LCD;
          TIEMPO = SEGUNDOS_LCD;
          break;
        case 5:
          SEG = DIFERENCIAH * 3600;
          SEG = SEG + DIFERENCIAM * 60;
          SEG = SEG + DIFERENCIAS;
          HRS = SEG / 3600;
          SEG = SEG - HRS * 3600;
          MIN = SEG / 60;
          SEG = SEG - MIN * 60;
          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
          if (myFile)
          {
            Serial.print("INTERRUPCIÓN: ");
            Serial.print(HRS);
            Serial.print(":");
            Serial.print(MIN);
            Serial.print(":");
            Serial.println(SEG);

```

```
397.            digitalWrite(13, 1);
398.            myFile.print("INTERRUPCIÓN: ");
399.            myFile.print(HRS);
400.            myFile.print(":");
401.            myFile.print(MIN);
402.            myFile.print(":");
403.            myFile.println(SEG);
404.            myFile.close();
405.            digitalWrite(13, 0);
406.          }
407.
408.          TIEMPO3 = HORAS_LCD;
409.          TIEMPO2 = MINUTOS_LCD;
410.          TIEMPO = SEGUNDOS_LCD;
411.          break;
412.        case 6:
413.          SEG = DIFERENCIAH * 3600;
414.          SEG = SEG + DIFERENCIAM * 21;
415.          SEG = SEG + DIFERENCIAS;
416.          HRS = SEG / 3600;
417.          HRS * -1;
418.          MIN = SEG + HRS * 3600;
419.          MIN = MIN * -1;
420.          SEG = SEG + MIN * 60;
421.          SEG = SEG * -1;
422.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
423.          if (myFile)
424.          {
425.            Serial.print("INTERRUPCIÓN: ");
426.            Serial.print(HRS);
427.            Serial.print(":");
428.            Serial.print(MIN);
429.            Serial.print(":");
430.            Serial.println(SEG);
431.
432.            digitalWrite(13, 1);
433.            myFile.print("INTERRUPCIÓN: ");
434.            myFile.print(HRS);
435.            myFile.print(":");
436.            myFile.print(MIN);
437.            myFile.print(":");
438.            myFile.println(SEG);
439.            myFile.close();
440.            digitalWrite(13, 0);
441.          }
442.
443.          TIEMPO3 = HORAS_LCD;
444.          TIEMPO2 = MINUTOS_LCD;
445.          TIEMPO = SEGUNDOS_LCD;
446.          break;
447.        case 7:
448.          SEG = DIFERENCIAH * 3600;
449.          SEG = SEG + DIFERENCIAM * 60;
450.          SEG = SEG + DIFERENCIAS;
451.          HRS = SEG / 3600;
452.          HRS = HRS * -1;
453.          MIN = SEG + HRS * 3600;
454.          MIN = MIN * -1;
455.          SEG = SEG + MIN;
456.          SEG = SEG * -1;
457.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
458.          if (myFile)
459.          {
460.            Serial.print("INTERRUPCIÓN: ");
461.            Serial.print(HRS);
462.            Serial.print(":");
463.            Serial.print(MIN);
464.            Serial.print(":");
465.            Serial.println(SEG);
466.
467.            digitalWrite(13, 1);
468.            myFile.print("INTERRUPCIÓN: ");
469.            myFile.print(HRS);
470.            myFile.print(":");
471.            myFile.print(MIN);
472.            myFile.print(":");
473.            myFile.println(SEG);
474.            myFile.close();
```

```
            digitalWrite(13, 0);
          }

          TIEMPO3 = HORAS_LCD;
          TIEMPO2 = MINUTOS_LCD;
          TIEMPO = SEGUNDOS_LCD;
          break;
        case 8:
          HRS = DIFERENCIAH * -1;
          MIN = DIFERENCIAM * -1;
          SEG = DIFERENCIAS * -1;
          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
          if (myFile)
          {
            Serial.print("INTERRUPCIÓN: ");
            Serial.print(HRS);
            Serial.print(":");
            Serial.print(MIN);
            Serial.print(":");
            Serial.println(SEG);

            digitalWrite(13, 1);
            myFile.print("INTERRUPCIÓN: ");
            myFile.print(HRS);
            myFile.print(":");
            myFile.print(MIN);
            myFile.print(":");
            myFile.println(SEG);
            myFile.close();
            digitalWrite(13, 0);
          }

          TIEMPO3 = HORAS_LCD;
          TIEMPO2 = MINUTOS_LCD;
          TIEMPO = SEGUNDOS_LCD;
          break;
      }
      myFile = SD.open("ENSAYO.txt", FILE_WRITE);
      if (myFile)
      {
        Serial.print("TIEMPO GENERAL: "); // imprime por puerto serie el tiempo entre medidas
        Serial.print(DIAS);
        Serial.print(":");
        Serial.print(HORAS); // valor del tiempo entre medidas
        Serial.print(":");
        Serial.print(MINUTOS);
        Serial.print(":");
        Serial.println(SEGUNDOS);
        Serial.println(" ");

        digitalWrite(13, 1);

        myFile.print("TIEMPO GENERAL: "); // imprime por puerto serie el tiempo entre medidas
        myFile.print(DIAS);
        myFile.print(":");
        myFile.print(HORAS); // valor del tiempo entre medidas
        myFile.print(":");
        myFile.print(MINUTOS);
        myFile.print(":");
        myFile.println(SEGUNDOS);
        myFile.println();
        myFile.close();
        digitalWrite(13, 0);
      }
      //************************************* IMPRESION DE VALORES EN LCD
    ***************************************//
      FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
    }
  }
}
//********************** FLAG ( INTERRUPCIÓN POR FLANCO ) *************************//


//************************* SERVICIO DE INTERRUPCION  ********************//
// CON ANTIREBOTE //
// CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //

void AUMENTAR()
```

```
{
  if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
  {
    INICIO_TIMER = millis();
    FLAG = 1;
  }
}
//*************************************************************************//

void SERIAL_EVENT()
{
  while (Serial.available()) // lectura del teclado
  {
    int ESTADO = '0'; // 0 ASCII

    ESTADO = Serial.read(); // La variable estado pasara a valer lo que detecte del teclado

    if (ESTADO == '+') // cuando presione + iniciara la medición
    {
      myFile = SD.open("ENSAYO.txt", FILE_WRITE);
      if (myFile)
      {
        Serial.print("MEDICION INICIADA");
        Serial.println();

        Serial.print("PRESIONE * PARA REINICIAR");
        Serial.println();

        Serial.print("PRESIONE - PARA CONTINUAR");
        Serial.println();
        Serial.println();

        digitalWrite(13, 1);

        myFile.print("MEDICION INICIADA");
        myFile.println();

        myFile.print("PRESIONE * PARA REINICIAR");
        myFile.println();

        myFile.print("PRESIONE - PARA CONTINUAR");
        myFile.println();
        myFile.println();
        myFile.close();

        digitalWrite(13, 0);
      }
      FLAG4 = 1;
    }

    if (ESTADO == '-') // cuando presione - pausará la medición
    {
      myFile = SD.open("ENSAYO.txt", FILE_WRITE);
      if (myFile)
      {
        digitalWrite(13, 1);
        Serial.print("MEDICION PAUSADA");
        Serial.println();

        Serial.print("PRESIONE + PARA CONTINUAR");
        Serial.println();

        Serial.print("PRESIONE * PARA REINICIAR");
        Serial.println();

        myFile.print("MEDICION PAUSADA");
        myFile.println();

        myFile.print("PRESIONE + PARA CONTINUAR");
        myFile.println();

        myFile.print("PRESIONE * PARA REINICIAR");
        myFile.println();
        myFile.close();
        digitalWrite(13, 0);
      }
      FLAG4 = 0;
    }
```

```
630.
631.     if (ESTADO == '*')
632.     {
633.       myFile = SD.open("ENSAYO.txt", FILE_WRITE);
634.       if (myFile)
635.       {
636.         Serial.print("MEDICION FINALIZADA... REINICIANDO");
637.         myFile.print("MEDICION FINALIZADA... REINICIANDO");
638.         myFile.close();
639.       }
640.
641.       CONTADOR = 0;
642.       CONT = 0;
643.       CONT_LCD = 0;
644.
645.       SEGUNDOS_LCD = 0;
646.       MINUTOS_LCD = 0;
647.       HORAS_LCD = 0;
648.       DIAS_LCD = 0;
649.
650.       TIEMPO = 0;
651.       TIEMPO2 = 0;
652.       TIEMPO3 = 0;
653.
654.       DIFERENCIAS = 0;
655.       DIFERENCIAM = 0;
656.       DIFERENCIAH = 0;
657.
658.       FLAG = 0;
659.       FLAG2 = 0;
660.       FLAG4 = 0;
661.     }
662.   }
663. }
664.
665.
666. //******************************** TIMER 1 ******************************//
667. // POR DESBORDAMIENTO //
668. // TIMER MADRE DE 1s  //
669. // EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
670. // LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
671. ISR(TIMER1_OVF_vect)
672. {
673.   TCNT1 = 0xC2F7; // F9E5 100 ms
674.
675.   if (FLAG4)
676.   {
677.     SEGUNDOS_LCD ++;
678.     FLAG2 = 1;
679.   }
680. }
681. //***************************************************************************//
```

Ensayo con software 10

Objetivo: Luego de varias muestras, como se presumía en un primer momento, la visualización del tiempo entre interrupciones causó errores, así que se optó por cambiar todo el algoritmo decimal a hexadecimal.

Resultado: Finalmente se logra visualizar bien los tiempos entre interrupciones de igual manera, se mantendrá en observación dicho algoritmo en caso de que presente errores.

```
1.  //********************** Liberias ************************************//
2.  #include <EEPROM.h>
3.  #include <SPI.h>
4.  #include <SD.h>
5.  //********************** Liberias ************************************//
6.
7.  File myFile;
8.
9.  //********************** definicion de variables ***********************//
10. volatile float CONTADOR = 0;
11. volatile int CONT = 0;
12. volatile float CONT_LCD = 0;
13. float CTE;
14.
15. unsigned long LIMITE = 10;
16. unsigned long INICIO_TIMER = 0;
17.
18. long SEGUNDOS_GRAL = 0;
19.
20. long  SEGUNDOS_SFTW = 0;
21. int  MINUTOS_SFTW = 0;
22. int  HORAS_SFTW = 0;
23. int DIAS_SFTW = 0;
24.
25. long SEGUNDOS_INT = 0;
26. int MINUTOS_INT = 0;
27. int HORAS_INT = 0;
28. int DIAS_INT = 0;
29.
30. bool FLAG = 0;
31. bool FLAG2 = 0;
32. bool FLAG3 = 0;
33. bool FLAG4 = 0;
34. //********************** definicion de variables ***********************//
35.
36.
37. //********* ATENCION EEPROM ************//
38.
39. float EEPROMEDIO;
40.
41. //********* ATENCION EEPROM ************//
42.
43.
44. //************************************** SETUP
    ****************************************//
45.
46. void setup()
47. {
48.   // TIMER 1 //
49.   // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
50.   // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
51.
52.   CONTADOR = 0;
53.   CONT_LCD = 0;
54.
55.   TCCR1A = 0;
56.   TCCR1B = 0;
57.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
58.
59.   TCNT1 = 0xC2F8; //F9E5 100 MS
```

```
60.
61.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
62.   //********************************* TIMER 1 *******************************//
63.
64.
65.   //******************** Grabo en EEPROM el valor promedio ******************//
66.
67.   // EEPROM.put (EEPROMEDIO, PROMEDIO);
68.
69.   //******************** Grabo en EEPROM el valor promedio ******************//
70.
71.
72.   //****************************** Inicializacion puerto serie
    ******************************//
73.   Serial.begin(9600);
74.   //****************************** Inicializacion puerto serie
    ******************************//
75.
76.
77.   //******************************** SERVICIO DE INTERRUPCIÓN
    ********************************//
78.
79.   pinMode(2, INPUT_PULLUP);
80.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
81.
82.   //******************************** SERVICIO DE INTERRUPCIÓN
    ********************************//
83.
84.
85.   //****************************** inicializacion de variables
    ******************************//
86.   FLAG = false;
87.   FLAG2 = false;
88.   FLAG3 = true;
89.   FLAG4 = false;
90.
91.
92.   CTE = 7.90;
93.
94.   //****************************** inicializacion de variables
    ******************************//
95.
96.   Serial.print("INICIANDO SD ...");
97.
98.   if (!SD.begin(4)) {
99.     Serial.println("ALGO FALLO!");
100.       while (1);
101.     }
102.     Serial.println("TODO JOYA");
103.
104.     SD.remove("ENSAYO.txt");
105.
106.     pinMode(13, OUTPUT);
107.
108.     //************************ INICIALIZACION DE PINES DEL LCD
    ************************//
109.
110.     myFile = SD.open("ENSAYO.txt", FILE_WRITE);
111.     if (myFile)
112.     {
113.       Serial.println("PRESIONE + PARA INICIAR");
114.       Serial.println("PRESIONE - PARA PAUSAR");
115.       Serial.println("PRESIONE * PARA REINICIAR");
116.       Serial.println();
117.
118.       digitalWrite(13, 1);
```

```
119.        myFile.println("PRESIONE + PARA INICIAR");
120.        myFile.println("PRESIONE - PARA PAUSAR");
121.        myFile.println("PRESIONE * PARA REINICIAR");
122.        myFile.println();
123.        myFile.close();
124.        digitalWrite(13, 0);
125.      }
126.      else
127.      {
128.        Serial.println("ERROR ABRIENDO ENSAYO.txt");
129.      }
130.    }
131.
132.    //**************************************************************************************
    *************//
133.
134.
135.    //******************************************** VOID LOOP
    ************************************//
136.    void loop()
137.    {
138.      SERIAL_EVENT();
139.
140.      if (FLAG4)
141.      {
142.        if (FLAG == 1) // FLAG del servicio de interrupción
143.        {
144.          FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción
145.          CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS
    DEMAS
146.          CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1
    CANGILON 7.9CM3
147.
148.          SEGUNDOS_SFTW = SEGUNDOS_GRAL ;
149.
150.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
151.          if (myFile)
152.          {
153.
154.            if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO A
155.            {
156.              myFile.print("A;");
157.              Serial.print("A;"); // Por puerto serie imprime el lado tambien en el que
    se encuentra
158.            }
159.
160.
161.            if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE
    INTERRUMPE EL FLAG PARA MARCAR SI ESTA EN EL LADO B
162.            {
163.              myFile.print("B;"); // Por puerto serie imprime el lado tambien en el que
    se encuentra
164.              Serial.print("B;"); // Por puerto serie imprime el lado tambien en el que
    se encuentra
165.            }
166.            myFile.close();
167.          }
168.
169.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
170.          if (myFile)
171.          {
172.            digitalWrite(13, 1);
173.            myFile.print(CONT_LCD);
174.            myFile.print(";");
175.            myFile.close();
```

```
176.            digitalWrite(13, 0);
177.
178.            Serial.print(CONT_LCD); // valor del agua
179.            Serial.print(";");
180.          }
181.
182.          // TIEMPO INTERRUPCIÓN
183.          if (SEGUNDOS_INT > 0)
184.          {
185.            SEGUNDOS_INT = SEGUNDOS_SFTW - SEGUNDOS_INT ;
186.
187.            DIAS_INT = SEGUNDOS_INT / 86400 ;
188.            SEGUNDOS_INT = SEGUNDOS_INT - DIAS_INT * 86400 ;
189.            HORAS_INT = SEGUNDOS_INT / 3600 ;
190.            SEGUNDOS_INT = SEGUNDOS_INT - HORAS_INT * 3600 ;
191.            MINUTOS_INT = SEGUNDOS_INT / 60 ;
192.            SEGUNDOS_INT = SEGUNDOS_INT - MINUTOS_INT * 60;
193.
194.            myFile = SD.open("ENSAYO.txt", FILE_WRITE);
195.            if (myFile)
196.            {
197.              digitalWrite(13, 1);
198.              myFile.print (DIAS_INT);
199.              myFile.print (":");
200.              myFile.print (HORAS_INT);
201.              myFile.print (":");
202.              myFile.print (MINUTOS_INT);
203.              myFile.print (":");
204.              myFile.print (SEGUNDOS_INT);
205.              myFile.print (";");
206.              myFile.close();
207.              digitalWrite(13, 0);
208.
209.              Serial.print (DIAS_INT);
210.              Serial.print (":");
211.              Serial.print (HORAS_INT);
212.              Serial.print (":");
213.              Serial.print (MINUTOS_INT);
214.              Serial.print (":");
215.              Serial.print (SEGUNDOS_INT);
216.              Serial.print (";");
217.            }
218.          }
219.
220.          SEGUNDOS_INT = SEGUNDOS_GRAL ;
221.
222.          //TIEMPO GENERAL QUE SE MUESTRA EN SOFTWARE
223.          DIAS_SFTW = SEGUNDOS_SFTW / 86400 ;
224.          SEGUNDOS_SFTW = SEGUNDOS_SFTW - DIAS_SFTW * 86400 ;
225.          HORAS_SFTW = SEGUNDOS_SFTW / 3600 ;
226.          SEGUNDOS_SFTW = SEGUNDOS_SFTW - HORAS_SFTW * 3600 ;
227.          MINUTOS_SFTW = SEGUNDOS_SFTW / 60 ;
228.          SEGUNDOS_SFTW = SEGUNDOS_SFTW - MINUTOS_SFTW * 60;
229.
230.          myFile = SD.open("ENSAYO.txt", FILE_WRITE);
231.          if (myFile)
232.          {
233.            digitalWrite(13, 1);
234.            myFile.print (DIAS_SFTW);
235.            myFile.print (":");
236.            myFile.print (HORAS_SFTW);
237.            myFile.print (":");
238.            myFile.print (MINUTOS_SFTW);
239.            myFile.print (":");
240.            myFile.println (SEGUNDOS_SFTW);
241.            myFile.close();
```

```
242.            digitalWrite(13, 0);
243.
244.            Serial.print (DIAS_SFTW);
245.            Serial.print (":");
246.            Serial.print (HORAS_SFTW);
247.            Serial.print (":");
248.            Serial.print (MINUTOS_SFTW);
249.            Serial.print (":");
250.            Serial.println (SEGUNDOS_SFTW);
251.          }
252.          //************************************** IMPRESION DE VALORES EN LCD
    ***************************************//
253.          FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
254.        }
255.      }
256.    }
257.    //************************ FLAG ( INTERRUPCIÓN POR FLANCO )
    **************************//
258.
259.
260.    //************************** SERVICIO DE INTERRUPCION  *********************//
261.    // CON ANTIREBOTE //
262.    // CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //
263.
264.    void AUMENTAR()
265.    {
266.      if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
267.      {
268.        INICIO_TIMER = millis();
269.        FLAG = 1;
270.      }
271.    }
272.    //*************************************************************************//
273.
274.    void SERIAL_EVENT()
275.    {
276.      while (Serial.available()) // lectura del teclado
277.      {
278.        int ESTADO = '0'; // 0 ASCII
279.
280.        ESTADO = Serial.read(); // La variable estado pasara a valer lo que detecte del
    teclado
281.
282.        if (ESTADO == '+') // cuando presione + iniciara la medición
283.        {
284.          FLAG4 = 1;
285.        }
286.
287.        if (ESTADO == '/') // Mensaje para el software
288.        {
289.          Serial.println("Comunicacion iniciada");
290.        }
291.
292.        if (ESTADO == '-') // cuando presione - pausará la medición
293.        {
294.          FLAG4 = 0;
295.        }
296.
297.        if (ESTADO == '*') // Reinicio de programa
298.        {
299.          CONTADOR = 0;
300.          CONT = 0;
301.          CONT_LCD = 0;
302.
303.          SEGUNDOS_GRAL = 0;
304.
```

```
305.          SEGUNDOS_SFTW = 0;
306.          MINUTOS_SFTW = 0;
307.          HORAS_SFTW = 0;
308.          DIAS_SFTW = 0;
309.
310.          SEGUNDOS_INT = 0;
311.          MINUTOS_INT = 0;
312.          HORAS_INT = 0;
313.          DIAS_INT = 0;
314.
315.          FLAG = 0;
316.          FLAG2 = 0;
317.          FLAG4 = 0;
318.        }
319.      }
320.    }
321.
322.
323.    //******************************** TIMER 1 ******************************//
324.    // POR DESBORDAMIENTO //
325.    // TIMER MADRE DE 1s  //
326.    // EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
327.    // LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
328.    ISR(TIMER1_OVF_vect)
329.    {
330.      TCNT1 = 0xC2F7; // F9E4 100 ms
331.
332.      if (FLAG4)
333.      {
334.        SEGUNDOS_GRAL ++;
335.        FLAG2 = 1;
336.      }
337.    }
338.    //***************************************************************************//
```

Ensayo con software 11

Objetivo: En lo que se tuvo foco en este software fue independizar completamente el hardware del puesto de recolección de datos, ya que para comenzar, pausar o reiniciar la medición , la conexión con el puerto serie era necesaria. Así que se optó por modificar el software para corregir dicho comportamiento.

Resultado: Este software será el final, se optó por excluir la parte de código que muestra información por la pantalla LCD, ya que el pluviómetro en su primer versión está completamente sellado y el LCD en sí, no mostraría información al usuario.

Ensayo 23

Se recolectaron 316,28 cm^3, la medición que realizó el pluviómetro fue de 347,60 cm^3, teniendo un error de 9,9% (entre 3 y 4 cangilones) teniendo en cuenta que la medición se realizó en la madrugada del sábado 27/08/22 y se obtuvo lo recolectado el martes 30/08/22.

PRUEBA12_CANGILON_TECHO_SD

```
1.  //********************** Liberias ************************************//
2.  #include <EEPROM.h>
3.  #include <SPI.h>
4.  #include <SD.h>
5.  //********************** Liberias ************************************//
6.
7.  File myFile;
8.
9.  //********************** definicion de variables ************************//
10. volatile float CONTADOR = 0;
11. volatile int CONT = 0;
12. volatile float CONT_LCD = 0;
13. float CTE;
14.
15. unsigned long LIMITE = 10;
16. unsigned long INICIO_TIMER = 0;
17.
18. long SEGUNDOS_GRAL = 0;
19.
20. long  SEGUNDOS_SFTW = 0;
21. int  MINUTOS_SFTW = 0;
22. int  HORAS_SFTW = 0;
23. int DIAS_SFTW = 0;
24.
25. long SEGUNDOS_INT = 0;
26. int MINUTOS_INT = 0;
27. int HORAS_INT = 0;
28. int DIAS_INT = 0;
29.
30. bool FLAG = 0;
31. bool FLAG2 = 0;
32. bool FLAG3 = 0;
33. bool FLAG4 = 0;
34. bool FLAG5 = 0;
35. //********************** definicion de variables ************************//
36.
37.
38. //********** ATENCION EEPROM ************//
39.
40. float EEPROMEDIO;
41.
42. //********** ATENCION EEPROM ************//
43.
44.
45. //************************************** SETUP ********************************************//
46.
47. void setup()
48. {
49.   // TIMER 1 //
50.   // TIMER POR DESBORDAMIENTO CON PRESCALER 1024 //
51.   // TIENE UN PERIODO DE 1S FRECUENCIA DE 1 HZ MARCA EL INICIO DE LA SEÑAL //
52.
53.   //delay(1000);
54.
55.   CONTADOR = 0;
56.   CONT_LCD = 0;
57.
58.   TCCR1A = 0;
59.   TCCR1B = 0;
60.   TCCR1B |= (1 << CS10) | (1 << CS12); //para prescaler de 1024 CS12=1 y CS10=1
61.
62.   TCNT1 = 0xC2F8; //F9E5 100 MS
63.
64.   TIMSK1 |= (1 << TOIE1); //timer open interrupt enable
65.   //********************************** TIMER 1 *********************************//
66.
67.
68.   //******************** Grabo en EEPROM el valor promedio ******************//
69.
70.   // EEPROM.put (EEPROMEDIO, PROMEDIO);
71.
```

```
72.   //******************** Grabo en EEPROM el valor promedio ******************//
73.
74.
75.   //****************************** Inicializacion puerto serie ****************************//
76.   Serial.begin(9600);
77.   //****************************** Inicializacion puerto serie ****************************//
78.
79.
80.   //****************************** SERVICIO DE INTERRUPCIÓN ******************************//
81.
82.   pinMode(2, INPUT_PULLUP);
83.   attachInterrupt(digitalPinToInterrupt(2), AUMENTAR, LOW);
84.
85.   pinMode(3, INPUT_PULLUP);
86.   attachInterrupt(digitalPinToInterrupt(3), CONTROL, HIGH);
87.
88.   //****************************** SERVICIO DE INTERRUPCIÓN ******************************//
89.
90.
91.   //****************************** inicializacion de variables ****************************//
92.   FLAG = false;
93.   FLAG2 = false;
94.   FLAG3 = true;
95.   FLAG4 = false;
96.   FLAG5 = false;
97.
98.
99.   CTE = 7.90;
100.
101.  //****************************** inicializacion de variables ****************************//
102.
103.  Serial.print("INICIANDO SD ...");
104.
105.  if (!SD.begin(53)) {
106.    Serial.println("ALGO FALLO!");
107.    while (1);
108.  }
109.  Serial.println("TODO JOYA");
110.
111.  //SD.remove("ENSAYO.txt");
112.
113.  pinMode(13, OUTPUT);
114.
115.  //************************ INICIALIZACION DE PINES DEL LCD ***********************//
116.
117.  digitalWrite(13, 1);
118.  Serial.println("PRESIONE PULSADOR PARA INICIAR O PAUSAR LA MEDICION");
119.  Serial.println();
120.  digitalWrite(13, 0);
121. }
122.
123. //*************************************************************************************************//
124.
125.
126. //****************************************** VOID LOOP ***********************************//
127. void loop()
128. {
129.   if (FLAG4 == 1)
130.   {
131.     if (FLAG == 1) // FLAG del servicio de interrupción
132.     {
133.       FLAG3 = ! FLAG3; // cambia de estado el FLAG3 cada interrupción
134.       CONTADOR ++; // AUMENTA EN 1 EL CONTADOR GENERAL, DE ESTE SE BASAN TODOS LOS DEMAS
135.       CONT_LCD = CONTADOR * 7.9; // SE MULTIPLICA POR LA VARIABLE DE MEDIDA DE 1 CANGILON 7.9CM3
136.
137.       SEGUNDOS_SFTW = SEGUNDOS_GRAL ;
138.
139.       if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO A
140.       {
141.         Serial.print("A;"); // Por puerto serie imprime el lado tambien en el que se encuentra
142.       }
143.       if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO B
144.       {
145.         Serial.print("B;"); // Por puerto serie imprime el lado tambien en el que se encuentra
146.       }
147.
```

```
148.      digitalWrite(13, 1);
149.      Serial.print(CONT_LCD); // valor del agua
150.      Serial.print(";");
151.      digitalWrite(13, 0);
152.
153.
154.      // TIEMPO INTERRUPCIÓN
155.      if (SEGUNDOS_INT > 0)
156.      {
157.        SEGUNDOS_INT = SEGUNDOS_SFTW - SEGUNDOS_INT ;
158.
159.        DIAS_INT = SEGUNDOS_INT / 86400 ;
160.        SEGUNDOS_INT = SEGUNDOS_INT - DIAS_INT * 86400 ;
161.        HORAS_INT = SEGUNDOS_INT / 3600 ;
162.        SEGUNDOS_INT = SEGUNDOS_INT - HORAS_INT * 3600 ;
163.        MINUTOS_INT = SEGUNDOS_INT / 60 ;
164.        SEGUNDOS_INT = SEGUNDOS_INT - MINUTOS_INT * 60;
165.
166.        digitalWrite(13, 1);
167.        Serial.print (DIAS_INT);
168.        Serial.print (":");
169.        Serial.print (HORAS_INT);
170.        Serial.print (":");
171.        Serial.print (MINUTOS_INT);
172.        Serial.print (":");
173.        Serial.print (SEGUNDOS_INT);
174.        Serial.print (";");
175.        digitalWrite(13, 0);
176.      }
177.
178.
179.      SEGUNDOS_INT = SEGUNDOS_GRAL ;
180.
181.      //TIEMPO GENERAL QUE SE MUESTRA EN SOFTWARE
182.      DIAS_SFTW = SEGUNDOS_SFTW / 86400 ;
183.      SEGUNDOS_SFTW = SEGUNDOS_SFTW - DIAS_SFTW * 86400 ;
184.      HORAS_SFTW = SEGUNDOS_SFTW / 3600 ;
185.      SEGUNDOS_SFTW = SEGUNDOS_SFTW - HORAS_SFTW * 3600 ;
186.      MINUTOS_SFTW = SEGUNDOS_SFTW / 60 ;
187.      SEGUNDOS_SFTW = SEGUNDOS_SFTW - MINUTOS_SFTW * 60;
188.
189.      digitalWrite(13, 1);
190.      Serial.print (DIAS_SFTW);
191.      Serial.print (":");
192.      Serial.print (HORAS_SFTW);
193.      Serial.print (":");
194.      Serial.print (MINUTOS_SFTW);
195.      Serial.print (":");
196.      Serial.println (SEGUNDOS_SFTW);
197.      digitalWrite(13, 0);
198.
199.      myFile = SD.open("ENSAYO.txt", FILE_WRITE);
200.      if (myFile)
201.      {
202.        digitalWrite(13, 1);
203.        if (FLAG3) // INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO A
204.        {
205.          myFile.print("A;");
206.        }
207.        if (!FLAG3) //INTERRUPCION DEL FLAG 3, QUE ES EL QUE CAMBIA CADA VEZ QUE INTERRUMPE EL FLAG PARA
    MARCAR SI ESTA EN EL LADO B
208.        {
209.          myFile.print("B;"); // Por puerto serie imprime el lado tambien en el que se encuentra
210.        }
211.
212.        myFile.print(CONT_LCD);
213.        myFile.print(";");
214.
215.        myFile.print (DIAS_INT);
216.        myFile.print (":");
217.        myFile.print (HORAS_INT);
218.        myFile.print (":");
219.        myFile.print (MINUTOS_INT);
220.        myFile.print (":");
221.        myFile.print (SEGUNDOS_INT);
222.        myFile.print (";");
223.
```

```
        myFile.print (DIAS_SFTW);
        myFile.print (":");
        myFile.print (HORAS_SFTW);
        myFile.print (":");
        myFile.print (MINUTOS_SFTW);
        myFile.print (":");
        myFile.println (SEGUNDOS_SFTW);
        myFile.close();
        digitalWrite(13, 0);
      }
    }
    FLAG = 0; // RESETEO DEL FLAG DEL SERVICIO DE INTERRUPCION
  }
}

//************************* SERVICIO DE INTERRUPCION  ********************//
// CON ANTIREBOTE //
// CADA VES QUE SE INTERRUPME PONE LA FLAG EN 1 //

void AUMENTAR()
{
  if ((millis() - INICIO_TIMER) > LIMITE) //ANTIRREBOTE
  {
    INICIO_TIMER = millis();
    FLAG = 1;
  }
}
//************************************************************************//

void CONTROL()
{
  if ((millis() - INICIO_TIMER) > LIMITE)
  {
    INICIO_TIMER = millis();
    FLAG4 = ! FLAG4;
    if(FLAG4)
    {
      Serial.println("Medicion iniciada");
      digitalWrite(13, 1);
    }
    if(!FLAG4)
    {
      Serial.println("Medicion pausada");
      digitalWrite(13, 0);
    }
  }
}


//******************************** TIMER 1 ******************************//
// POR DESBORDAMIENTO //
// TIMER MADRE DE 1s  //
// EL FLAG2 SE SETEA EN UNO CADA VEZ QUE INTERRUMPE EL TIMER //
// LA VARIABLE SEGUNDOS_LCD AUMENTA EN 1 CADA VES QUE INTERRUMPE EL TIMER //
ISR(TIMER1_OVF_vect)
{
  TCNT1 = 0xC2F7; // F9E4 100 ms

  if (FLAG4)
  {
    SEGUNDOS_GRAL ++;
    FLAG2 = 1;
  }
}
//************************************************************************//
```

CIERRE GENERAL

El pluviómetro digital está en medición efectiva a la fecha, almacenando información constante en su memoria SD, para continuar el proyecto se desarrollará un programa que procese dicha información, pudiendo graficar, mostrar los datos, controlar las mediciones, informar de problemas, entre otras funciones que se determinarán durante el desarrollo de dicha software.