



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE TELECONTROL
DE BIODIGESTOR DE ABONO CON INTERFAZ WEB MEDIANTE USO DE
HARDWARE Y SOFTWARE LIBRE”

INFORME DE PROYECTO INTEGRADOR

Previo a la obtención del TÍTULO de

INGENIERO EN TELEMÁTICA

Presentado por:

Paola Carolina Jordán Figueroa

Adrian Patricio Elizalde Orellana

Guayaquil – Ecuador

AÑO 2015

AGRADECIMIENTO

Primeramente a Dios por darnos la fuerza y sabiduría necesaria para poder cumplir con todas las metas propuestas.

A nuestros padres por todo el apoyo incondicional por habernos brindado la mejor educación y enseñarnos que con esfuerzo, trabajo y constancia todo se consigue.

A nuestro Director el Ing. Marcos Millán quien ha desempeñado el arduo trabajo de transmitirnos sus diversos conocimientos, especialmente de los temas relacionados con nuestra profesión.

A todos aquellos que han aportado con sus conocimientos y nos han proporcionado su ayuda para poder llevar a cabo la realización de este proyecto.

DEDICATORIA

Con mucho amor a mis padres Pilar y Germán quienes con esfuerzo y sacrificio me han brindado la educación y las lecciones de vida necesarias para mi formación personal, por ser ejemplo de mi vida y por haberme enseñado que ningún logro te llena por completo si no hay esfuerzos detrás de él.

A mis hermanos Vanessa, Oscar y Andrea por cuidarme y apoyarme en todo momento.

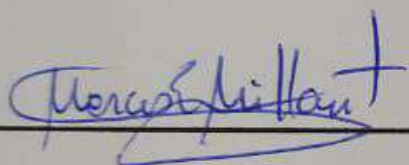
A todas las personas extraordinarias que han formado parte de mí día a día durante esta etapa y que me han ayudado a crecer tanto profesionalmente y sobre todo como persona.

Paola Carolina Jordán Figueroa

De manera especial a mis padres los cuales han sido los pilares fundamentales de mi vida tanto personal como profesional. Gracias por todos los consejos que me han sabido dar, el esfuerzo que realizan día a día por darme lo mejor y por todo el amor infinito que me han brindado.

Adrian Patricio Elizalde Orellana

TRIBUNAL DE EVALUACIÓN



M. Sc. Marcos Millán Traverso

PROFESOR EVALUADOR

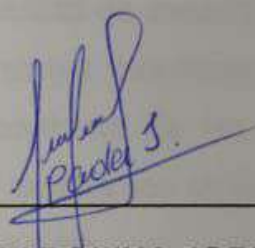


M. Sc. José Menéndez Sánchez

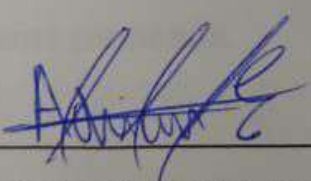
PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, nos corresponde exclusivamente; y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"



PAOLA CAROLINA JORDÁN FIGUEROA



ADRIAN PATRICIO ELIZALDE ORELLANA

RESUMEN

En el Centro de Investigaciones Biotecnológicas del Ecuador – CIBE, el área de Técnicas Agrícolas se encargan de la transferencia tecnológica de los resultados obtenidos en las investigaciones del centro hacia los agricultores del Ecuador, además de interactuar con estos para la solución de problemas de carácter productivo a nivel de campo. Un producto en específico es el Biol un tipo de abono orgánico para la fertilización de cultivos, que se lo desarrolla a partir de desechos agropecuarios y residuos sólidos urbanos, en la actualidad el centro cuenta con un biorreactor para realizar pruebas en entorno de laboratorio para la producción y mejora de Biol, este no cuenta con un mecanismo, tal como un página web que permita sistematizar y automatizar el sistema de producción del biofertilizante.

Es por ello que surge la necesidad de desarrollar un portal web con un prototipo de biodigestor de abono que incluya monitoreo del proceso a través de sensores permitiendo la adquisición constante de las variables indicadoras de calidad del producto. El prototipo también cuenta con un tablero de manejo local que cumple con la misma función que la página web.

Para la construcción de este prototipo y del portal web, lo primordial fue el uso de hardware y software en libre en todos los aspectos que conciernen con la implementación del sistema planteado.

ÍNDICE GENERAL

AGRADECIMIENTO	ii
DEDICATORIA	iii
TRIBUNAL DE EVALUACIÓN	iv
RESUMEN.....	vi
ÍNDICE GENERAL	vii
CAPÍTULO 1.....	1
1. GENERALIDADES	1
1.1 Antecedentes	1
1.2 Conceptualización del problema.....	1
1.3 Delimitación del problema	2
1.4 Justificación del proyecto	2
1.5 Objetivos	2
1.5.1 Objetivo general	2
1.5.2 Objetivos específicos.....	3
1.6 Metodología	3
CAPÍTULO 2.....	5

2.	FUNDAMENTACIÓN TEÓRICA	5
2.1	Biodigestor	5
2.1.1	Sistema de Agitación	5
2.2	Biol	6
2.3	Proceso de producción de biol	6
2.4	Descripción de los parámetros indicadores de calidad de la mezcla	7
2.4.1	Temperatura	7
2.4.2	pH	8
2.4.3	Conductividad eléctrica	8
2.4.4	Total de sólidos disueltos	9
2.4.5	Salinidad	10
	CAPÍTULO 3	11
3.	DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO	11
3.1	Estructura del prototipo	11
3.2	Sistema de control del biodigestor	11
3.2.1	Etapa de adquisición de información	12
3.2.2	Etapa de procesamiento y toma de decisiones	17

3.2.3	Etapa de salida.....	19
3.3	Descripción del software	25
3.4	Descripción del hardware	35
3.5	Interfaz humano máquina.....	36
3.6	Financiamiento.....	40
3.6.1	Análisis Comparativo Cualitativo - Económico.....	42
CAPÍTULO 4.....		43
4.	RESULTADOS	43
4.1	Prototipo en funcionamiento.....	43
CONCLUSIONES Y RECOMENDACIONES		47
BIBLIOGRAFÍA.....		49
ANEXOS		53

CAPÍTULO 1

1. GENERALIDADES

1.1 Antecedentes

En la actualidad el CIBE tiene establecidos seis líneas de investigación: Técnicas Agrícolas, Cultivo de Tejidos, Fitopatología – Microbiología, Bioproductos, Biología Molecular y Bioestadística – Bioinformática. Dentro del área de Técnicas Agrícola se estudia el desarrollo o transferencia de tecnologías novedosas y sostenibles para mejorar el rendimiento de cultivos. Uno de los tantos proyectos es la producción de un biofertilizante llamado Biol con el cual se logra una mejor fertilización del suelo y consecuentemente un alto rendimiento en el crecimiento de la planta, el cual ha sido exitosamente aplicado en varias provincias de la zona costera del país (Esmeraldas, Manabí, Los Ríos, Guayas y El Oro) con resultados alentadores, proporcionando la fórmula del compuesto a los productores y que esta se pueda distribuir a los demás sectores agrícolas por medio de sus agricultores.

Con el biol se ha conseguido una inhibición total en ambiente de laboratorio, sin embargo en el campo la eficacia del Biol se reduce al 70% u 80%. Gracias a la aplicación del compuesto orgánico en las plantaciones se ha mejorado en promedio el 50% de la producción. [1]

1.2 Conceptualización del problema

En los laboratorios del CIBE se hacen pruebas y mejoramientos continuos del producto (Biol), a través de un equipo de alto nivel, un biorreactor FoxyLogic. En la actualidad este equipo se encuentra con ciertas fallas y la medición de los parámetros se la realiza de forma manual por lo que se debe abrir el envase razón por la cual ingresa oxígeno a la mezcla retrasando de esta manera el proceso, impidiendo así las investigaciones para la mejora del compuesto, por otro lado la problemática que afecta al personal que lo maneja es que se pierde tiempo al efectuar esta toma de medidas manuales es decir, el sistema no es eficiente. Como consecuencia surge la necesidad de implementar un sistema de telecontrol que aporte al avance del desarrollo del producto.

1.3 Delimitación del problema

Este proyecto comprenderá la creación de un prototipo de un biodigestor de biofertilizante que será dirigido para el uso de pruebas y mejoramiento del producto (Biol) en los laboratorios del área de técnicas agrícolas del CIBE, se controlará el proceso tanto por un tablero de manejo local como también mediante una interfaz web que podrá ser accedida de manera remota a través de un dirección pública de Internet.

1.4 Justificación del proyecto

La decisión de implementar un sistema de control telemétrico para el biodigestor se debe a que en la actualidad el CIBE no posee el equipo que permite realizar las pruebas y mediciones de los compuestos del producto biofertilizante para llevar un control de su calidad y producción, por lo cual el servicio ofrecido es un medio viable para poder continuar realizando las pruebas necesarias para la mejora del producto.

A través de la conexión y adaptación del conjunto de software y hardware al biodigestor se logra medir las variables implicadas en el proceso de biodigestión tales como temperatura y pH entre las principales y otras medidas contenidas en el compuesto como conductividad eléctrica, total de sólidos disueltos, salinidad. Estos datos pueden ser observados a través de una interfaz web y se puede aplicar alguna medida de control en el caso de que las variables estén fuera del rango de operación, ya sea regulando su pH o temperatura.

La administración a través de acceso remoto brinda una facilidad en el manejo y operación del proceso de biodigestión mediante el cual los usuarios pueden actuar sobre el sistema desde cualquier parte sin la necesidad de tener que estar presentes en el lugar.

1.5 Objetivos

1.5.1 Objetivo general

- Diseñar e implementar un sistema automatizado para la supervisión de la actividad del proceso de biodigestión de abono orgánico.

1.5.2 Objetivos específicos

- Diseñar e implementar un prototipo de un biodigestor para la producción de abono orgánico (Biol), en el cual se lleve un control constante de las variables principales del proceso como temperatura, pH, conductividad eléctrica, total de sales disueltas, salinidad.
- Programar rutinas de software para el microcontrolador, que permita el manejo de sensores, motores, válvulas. Así como también un sistema de alarma mediante correo electrónico que notifique al usuario de cambios anormales en el proceso.
- Implementar una página web que permita el monitoreo remoto por internet de la actividad del biodigestor, permitiendo visualizar los parámetros más importantes del proceso, así como el estado de las válvulas on/off y del sistema de agitación.
- Evidenciar los resultados del sistema implementado, a través del cumplimiento de los objetivos anteriormente expuestos.

1.6 Metodología

Como etapa inicial del proyecto se recopilará información del proceso de producción del abono orgánico con la finalidad de determinar los componentes claves para la monitorización que formarán parte del prototipo final. Previamente a la etapa inicial se realizará una visita a las instalaciones del CIBE para conocer la situación actual del manejo para la producción del biofertilizante. Para la integración de todas las funcionalidades del proyecto se utilizará un microcontrolador que será el núcleo del sistema y se encargará del manejo de todas las señales obtenidas de los componentes de entrada (sensores) para habilitar componentes de salidas (actuadores) de acuerdo a las lecturas sensadas. Como parte del proyecto se implementará y diseñará una interfaz web para manejo remoto del sistema. Finalmente se construirá un prototipo para simular el proceso y verificar su funcionamiento de acuerdo a lo planteado en los objetivos.

CAPÍTULO 2

2. FUNDAMENTACIÓN TEÓRICA

2.1 Biodigestor

Un biodigestor es un recipiente herméticamente cerrado (reactor) que conserva un ambiente biológicamente activo, en el cual se deposita material orgánico a descomponerse (excrementos de animales y humanos, desechos vegetales-no se incluyen cítricos ya que acidifican-, etcétera) en determinada solución de agua para que a través de la fermentación anaerobia se origine gas metano y fertilizantes orgánicos ricos en nitrógeno, fósforo y potasio, y al mismo tiempo, se reduzca el potencial contaminante de los excrementos.

En general, un biodigestor trata de adecuar ciertas condiciones ambientales propicias (pH, temperatura, concentración de oxígeno, etcétera) al organismo o sustancia química que se cultiva.

Los fermentadores agitados se utilizan generalmente para cultivos aeróbicos y exhiben un comportamiento en la concentración de sustrato. El biorreactor de tanque agitado es de uso muy definido. La agitación se realiza mecánicamente mediante un eje provisto de turbinas accionadas por un motor. [2] [3]

2.1.1 Sistema de Agitación

La agitación es la operación que crea o acelera el contacto entre dos o varias fases. Una adecuada agitación es esencial para la fermentación ya que produce los siguientes efectos:

- Dispersión de aire en la solución de nutrientes.
- Homogeneización, para igualar la temperatura, pH, concentración de nutrientes.
- Suspensión de los microorganismos de nutrientes sólidos.
- Dispersión de los líquidos inmiscibles.

Cuanto mayor sea la agitación, mejor será el crecimiento. La agitación excesiva puede romper las células grandes e incrementar la temperatura ocasionando un descenso en la viabilidad celular.

Un sistema de agitación tiene la función de generar la potencia necesaria para producir una mezcla perfecta para el sistema de cultivo y producir un

régimen de agitación adecuado que, maximice la difusión de gases en el líquido y minimice la producción de esfuerzos cortantes y la presión hidrodinámica local y global, para optimizar los fenómenos de transferencia de momentum, calor y masa. [3]

Un sistema de agitación consta de las siguientes partes:

Motor impulsor: Estos motores deben contar con la capacidad de operar continuamente durante largas jornadas de trabajo, preferiblemente debe de ser construido de acero inoxidable.

Eje Transmisor de Potencia: Es una barra metálica cilíndrica generalmente de acero inoxidable con medidas de diámetros estándares $\frac{1}{2}$ " y $\frac{3}{4}$ ", y el largo de la varilla es dependiente de la profundidad del recipiente del biodigestor.

Agitador: Es un propulsor formado por hojas o aspas que van conectadas al eje transmisor de potencia, pueden tener una distribución de flujo axial o radial.

2.2 Biol

El biol es un abono orgánico líquido, resultado de la descomposición de los residuos animales, vegetales y minerales en ausencia de oxígeno. Contiene nutrientes que son asimilados fácilmente por las plantas haciéndolas más vigorosas y resistentes. La técnica empleada para lograr éste propósito son los biodigestores.

Es el líquido que se descarga de un digestor y es un excelente estimulante foliar para las plantas y un completo potenciador de los suelos.

Revitaliza las plantas que sufren estrés, ya sea por plagas, enfermedades o interrupción de sus procesos normales de desarrollo mediante una oportuna, sostenida y buena nutrición, ofreciendo así alimentos libres de residuos químicos [4].

2.3 Proceso de producción de biol

La producción del biol es un proceso relativamente simple y de bajo costo, ya que sus insumos de preparación son locales. El biol tiene dos componentes: una parte sólida y una líquida. La primera es conocida como biosol y se obtiene como producto de la descarga o limpieza del biodigestor donde se elabora el biol. La

parte líquida es conocida como abono foliar. El resto sólido está constituido por materia orgánica no degradada, excelente para la producción de cualquier cultivo. En el biol podemos usar cualquier tipo de estiércol.

El biol revitaliza las plantas que sufren estrés, ya sea por plagas, enfermedades o interrupción de sus procesos normales de desarrollo mediante una oportuna, sostenida y buena nutrición, ofreciendo así alimentos libres de residuos químicos. Para la elaboración del biol no es necesaria una receta, simplemente lo elaboramos con los residuos que hay en nuestro alrededor. El biol estimula y fortalece el desarrollo de las plantas, mejora la producción de frutos, los cultivos se vuelven resistentes al ataque de las enfermedades y los cambios adversos del clima [5].

El proceso de fermentación del Biol es extenso, tiene una duración de cuatro meses. Durante este proceso se deben medir constantemente varios parámetros indicadores de calidad los cuales tienen que estar dentro de un rango de operación para que garantizar un resultado óptimo. Entre las principales medidas encontramos las siguientes:

- pH entre 3.8 - 4.2
- Temperatura entre 10 – 70 °C
- Conductividad Eléctrica entre 20 – 24 mS/cm
- Total de sólidos disueltos entre 5.7 – 7 mg/lit
- Salinidad entre 9 – 11 ppm

2.4 Descripción de los parámetros indicadores de calidad de la mezcla

2.4.1 Temperatura

Una temperatura uniforme y aceptable se debe de mantener completamente en el biodigestor para prevenir cavidades localizadas de disminución en la temperatura y actividad bacteriana indeseada. Variaciones en la temperatura de incluso pocos grados afectan casi toda la actividad biológica incluyendo la inhibición de algunas bacterias anaeróbicas, especialmente las bacterias productoras de metano. La mezcolanza adecuada del contenido digestivo impide el desarrollo de zonas de variación de temperatura.

La temperatura es uno de los factores más importantes que afectan la actividad microbiana dentro de un digestor anaeróbico. Las fluctuaciones en la temperatura afectan la actividad de las bacterias productoras de metano a un rango mayor que la temperatura de trabajo.

La temperatura no solo influye no solo en las bacterias productoras de metano sino también en las bacterias productoras de ácidos.

2.4.2 pH

La suficiente alcalinidad es esencial para el adecuado control del pH. La alcalinidad sirve como un amortiguador que previene los cambios rápidos en el pH. La actividad enzimática (o el desempeño digestivo) es influenciado por el pH. Una actividad enzimática aceptable de bacterias productoras de ácidos ocurre por encima de 5.0 pH.

El pH en un biodigestor anaeróbico inicialmente decrecerá con la producción de ácidos volátiles. Sin embargo, como las bacterias productoras de metano consumen los ácidos volátiles y la alcalinidad es producida, el pH del biodigestor incrementa y después se estabiliza.

La estabilidad de los digestores es incrementado por una concentración de alcalinidad alta. Un decremento en la alcalinidad por debajo del nivel operación normal ha sido usado como indicador de falla por resolver. Un decremento en la alcalinidad puede ser causado por:

- Una acumulación de ácidos orgánicos debido a la falla de las bacterias productoras de metano al convertir los ácidos orgánicos a metano.
- Una descarga de golpe de ácidos orgánicos hacia el digestor anaeróbico.
- La presencia de desperdicios que inhiben la actividad de las bacterias productoras de metano.

Un decremento en la alcalinidad usualmente precede a un cambio rápido en el pH.

2.4.3 Conductividad eléctrica

La conductividad eléctrica consiste en la capacidad que tiene una sustancia para conducir una corriente eléctrica a través de iones disueltos,

por lo que es lo inverso a la resistencia eléctrica. Dentro de los líquidos las sales disueltas se descomponen en iones cargados positivamente y negativamente. Los iones más positivos son sodio, calcio, potasio y magnesio. Los iones más negativos son cloruro, sulfato, carbonato, bicarbonato. Fosfatos y nitratos no aportan en gran consideración a la conductividad sin embargo poseen importancia biológicamente.

La conductividad es medida por medio de una sonda electrónica que aplica voltaje entre dos electrodos. A través de la reducción del voltaje se mide la resistencia del agua la cual es traducida en conductividad. La unidad de medición comúnmente utilizada es el Siemens/cm (S/cm), en millonésimas (10^{-6}) de unidades, es decir microSiemens/cm ($\mu\text{S/cm}$), o en milésimas (10^{-3}) es decir miliSiemens/cm (mS/cm). [6]

2.4.4 Total de sólidos disueltos

El total de sólidos disueltos es una medida de la suma de los minerales, sales, metales, cationes o aniones que están disueltos en un volumen de agua. Es el conjunto total de sustancias contenidas en el agua que no sean agua pura (H_2O) y sólidos en suspensión. (Sólidos en suspensión son partículas que persisten sin disolverse en el agua.)

La calidad y pureza del agua procedente de los sistemas de purificación están intrínsecamente relacionadas con los TDS. De forma habitual, la concentración de sólidos disueltos totales es la adición de los cationes (carga positiva) y aniones (cargado negativamente) iones en el agua.

Para medir los TDS se utiliza la conductividad eléctrica del agua. El agua tiene prácticamente un valor de conductividad de cero. La conductividad es generalmente cerca de 100 veces el total de cationes o aniones expresados como equivalentes. Los medidores de TDS se calculan mediante la conversión de la CE por un factor de 0,5 a 1,0 veces la CE, dependiendo de los niveles. En soluciones acuosas la conductividad es directamente proporcional a la concentración de sólidos disueltos, por lo tanto cuanto mayor sea dicha concentración, mayor será la conductividad. El TDS se expresa en unidades de mg (miligramos) por litro (l), es decir mg/l. [7]

2.4.5 Salinidad

La salinidad es una medida de la cantidad de sales disueltas en agua. Debido a la presencia de cloruro de sodio (NaCl) en el agua el sabor es salado. El porcentaje medio que existe en los océanos es de 3,5% (35 gramos por cada litro de agua). Además esta salinidad se ve alterada de acuerdo a la intensidad de la evaporación o el aporte de agua dulce de los ríos aumente en relación a la cantidad de agua. La salinización y desalinización son procesos a través de los cuales se aumenta o disminuye la salinidad respectivamente. Las sales y otras sustancias afectan a la calidad del agua potable, además influyen en la biota acuática y cada organismo tolera una gama de valores de conductividad. La composición iónica del agua puede ser crítica. La cantidad de iones disueltos incrementan los valores tanto de la salinidad como de la conductividad por lo cual están relacionadas.

La salinidad se mide en ppm (partes por millón o partículas por millón) en agua salada. En aguas dulces se utiliza con frecuencia las cantidades totales de sólidos disueltos en vez de salinidad. Se mide filtrando una muestra, el agua filtrada se seca y los sólidos restantes se pesan. Las TDS son las materias sólidas que quedan en el agua después de haberse evaporado. TDS y ppm tienen la misma equivalencia: ppm= mg/L. [8]

CAPÍTULO 3

3. DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO

3.1 Estructura del prototipo

El diseño propuesto del prototipo del biodigestor de abono comprende dos escenarios principales: Hardware y Software. Los componentes que forman parte de la estructura fueron elegidos en base a las características que más se ajustaban a los requerimientos planteado.

Se considera para el desarrollo del proyecto la utilización de hardware y Software libre en la medida de la posible.

3.2 Sistema de control del biodigestor

Considerando los requerimientos del sistema se decidió separar el diseño en varias etapas para tratar cada sección de manera independiente. Las etapas establecidas fueron las siguientes:

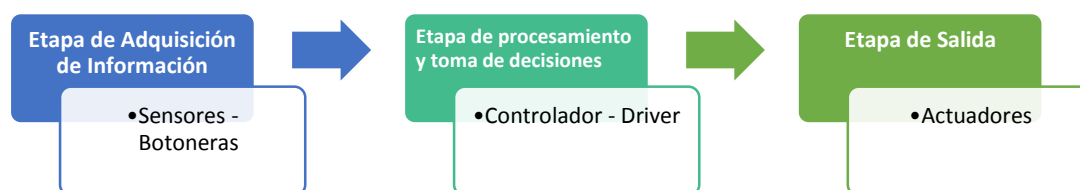


Figura 3.1: Etapas para el diseño del sistema de control

Etapa de adquisición de información, encargada de enviar la información a un control central a través de sensores.

Etapa de procesamiento y toma de decisiones, por medio de un controlador que recibe y procesa los datos obtenidos.

Etapa de salida, consta de sistemas de acople y potencia para activar los actuadores.

A continuación se explica en detalle cada una de las etapas para automatizar el proceso justificando la selección de cada componente utilizado.

3.2.1 Etapa de adquisición de información

Los parámetros indicadores de calidad de la solución dentro del biodigestor se obtienen a través de sensores que en su mayoría son analógicos los cuales envían el valor medido en forma de voltaje. Para la facilidad del manejo de los sensores analógicos se emplean circuitos controladores que convierten las señales analógicas de voltaje en señales digitales con tipo de dato String.

Sensor de temperatura

Este sensor es el encargado de medir la temperatura, variable importante dentro del proceso de biodigestión, los requerimientos mínimos para este sensor son:

- Capacidad de medir de 0 a 100 °C
- Capacidad de soportar un ambiente y algunos otros compuestos orgánicos que pudieran provocar oxidación o degradación del dispositivo.
- Tener un costo relativamente bajo.
- De fácil manejo.

En la Figura 3.1 se muestra una de las opciones ofrecidas dentro del mercado nacional, el sensor de temperatura DS18B20 el cual es a prueba de agua y ofrecido por la empresa Quiteña TecMikro.

Las principales características del sensor son:

- Alimentación: 3.0 V a 5.0 V
- Interfaz 1-Wire (Puede funcionar con un solo pin)
- Rango de Temperatura: - 55 a 125 °C
- Resolución de 9 a 12 bits (configurable)
- Precisión: ± 0.5 °C
- Identificador interno único de 64 bits
- Tiempo de captura inferior a 750 ms
- Múltiples sensores pueden compartir el mismo pin

Las características del cable:

- Largo: 91 cm
- Diámetro : 4mm
- Tubo de acero inoxidable: 6 mm diámetro por 30 mm de largo
- Cables de conexión, Rojo para alimentación positiva, Negro para GND, Amarillo para el bus de datos. La malla de cobre está internamente soldada a GND.



Figura 3.2: DS18B20 WP Sensor de Temperatura [9]

Una de las consideraciones importantes al momento de utilizar el sensor es que necesita una resistencia de 4.7 K Ohm conectada entre el pin de datos y el pin de VCC.

Sensor de pH

Este sensor tendrá una ubicación interna dentro del envase del biodigestor para la medición constante del pH por lo que la calidad de este dispositivo deberá ser tal que no se deteriore bajo las condiciones de temperatura o reacciones químicas dentro del recipiente muchas opciones no son aptos para mediciones permanentes o bajo ambientes industriales de temperatura o humedad.

En la Figura 3.2 se puede observar el dispositivo seleccionado, este elemento posee características excelentes de las cuales se destaca que no está fabricado de materiales tóxicos que pueden afectar el sistema, también da respuesta rápida en las lecturas continuas con muy pocas

fluctuaciones, cuenta con un cable de 10 pies con una chaqueta que elimina la alta interferencia y tiene una referencia Ag-Ag/Cl. [10]



Figura 3.3: Sensor de pH American Marine Pinpoint pH Probe [10]

Este sensor será operado mediante un circuito integrado embebido de la marca Atlas Scientific el EZO™ pH Circuit, este circuito integrado ofrece el más alto nivel de estabilidad y precisión, con la configuración adecuada de este circuito se obtiene como resultado el valor del pH que se obtiene del electrodo o sensor descrito anteriormente. [11]

En la Figura 3.3 se muestra el circuito embebido, a continuación se describe detalladamente algunas características importantes:

- Gama completa de lectura de pH 0.001 – 14.000
- Precisión (± 0.02)
- Lecturas independientes de la temperatura
- Protocolo de calibración flexible soporta un solo punto, dos puntos o 3 puntos.
- Lecturas individual o modo de lectura continua
- Formato de los datos es ASCII
- Protocolos de Datos: UART, I2C
- Compatible con cualquier microcontrolador que soporte UART e I2C
- Voltaje de funcionamiento de 3.3 V – 5 V



Figura 3.4: EZO TM pH Circuit de Atlas Scientific [12]

Sensor de conductividad eléctrica

Este sensor al igual que el sensor de pH debe de ir dentro del recipiente del biodigestor para tomar las medidas continuamente, debe de ser de un material duradero, soportar escenarios químicos y de altas temperaturas. En lo que respecta a su rango de medidas debe evaluar medidas de conductividad eléctrica entre 20 -24 mS/cm que son las que se necesitan para mantener un ambiente estable dentro del envase del biodigestor.

El dispositivo elegido para esta tarea es el que se muestra en la Figura 3.4, esta sonda de conductividad tiene una constante de celda de K 1.0 nos proporciona una lectura entre 5 μS – 200,000 μS , tiene dos conductores de grafito, el área del conductor es fácilmente identificado por la sección marrón de la sonda, una consideración importante a tomar es que la zona de conducción debe de ser totalmente sumergida con el fin de recibir información precisa de lectura. [13]



Figura 3.5: Sensor de Conductividad Eléctrica Conductivity Probe K 1.0 [13]

El circuito integrado encargado del manejo de esta sonda es el EZO™ Conductivity Circuit así mismo fabricado por la empresa Atlas Scientific, esto nos permite mediante la tarjeta de desarrollo elegida tener un acceso a los datos tomados por la sonda. En la Figura 3.5 se distingue este circuito, a continuación se expresan algunas características importantes [14]:

- Realiza lectura de Conductividad Eléctrica (EC), Total de Sólidos Disueltos (TDS) y Salinidad (S).
- Precisión en las lecturas de Conductividad $\pm 2 \mu\text{S}/\text{cm}$.
- Rango de medidas de EC $0.07 \mu\text{S}/\text{cm} - 500,000 \mu\text{S}/\text{cm}$.
- Lecturas independientes de la temperatura
- Protocolo de calibración flexible soporta un solo punto, 2 puntos o 3 puntos.
- Se requiere una calibración por año.
- El formato de datos es ASCII
- Se pueden usar dos tipos de protocolos de comunicación UART o I2C.
- Voltaje de funcionamiento: $3.3 \text{ V} - 5 \text{ V}$.

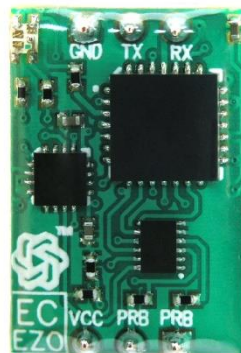


Figura 3.6: EZO™ Conductivity Circuit [14]

3.2.2 Etapa de procesamiento y toma de decisiones

El controlador constituye el elemento fundamental del sistema de control, para la elección de este componente se consideró un sistema embebido que cuenta con un sistema operativo de código abierto y libre distribución. Se han analizado las características de varios sistemas similares dentro de los cuales se optó por trabajar con la BeagleBone Black debido a su alto nivel de procesamiento y su facilidad de manejo.

La BeagleBone Black es una tarjeta de desarrollo de bajo costo que cuenta con un sistema operativo Linux con un kernel 3.8.13, distribución DEBIAN v 7.8, permitiendo realizar funciones semejantes a un ordenador. Ciertas ventajas que ofrece esta tarjeta son:

- Fácil conexión a la red, posee un puerto Ethernet para su conectividad a internet además de proveer servicios como FTP, SSH, Telnet o incluso habilitar un propio servidor web.
- Sistema de archivos, para almacenamiento, organización y respaldo de la información.
- Lenguajes de programación, se puede programar en diferentes lenguajes tales como C, C++, Python, Perl, Ruby o incluso programación Shell script.
- Multitarea, capaz de compartir el procesador entre programas y tareas que se ejecutan concurrentemente. [15]

A continuación se describe detalladamente algunas características de la Beaglebone Black:

Características	BeagleBone Black
SoC	Texas Instruments AM3358
Procesador	ARM Cortex-A8
Arquitectura	ARMv7
Velocidad	1ghz
Memory	512MB
FPU	Hardware
GPU	PowerVR SGX350
Almacenamiento Interno	2GB (rev B) o 4GB (rev C)
Almacenamiento Externo	MicroSD
Tarjeta de red	10/100Mbit Ethernet
Fuente de poder	5V desde USB mini B conector, 2.1mm Jack.
Dimensiones	3.4in x 2.1in (86.4mm x 53.3mm)
Peso	1.4oz (40g)
Pines Digitales E/S	65
Voltaje E/S Digitales	3.3V
Entrada Analógica	7 con 12-bit ADC, 0-1.8V
Salidas PWM	8
UART	4

SPI	2
I2C	2
Salida de Video	Micro HDMI
Entrada de Video	Ninguna
Salida de audio	Micro HDMI
Potencia de Salida	3.3V a 250mA, 5V a 1A.
Otros	Pines con múltiples funcionalidades tales como audio I2S, bus CAN, etc.

Tabla 1: Especificaciones Beaglebone Black [16]

3.2.3 Etapa de salida

Los actuadores son dispositivos que convierten magnitudes eléctricas en salidas que pueden provocar efectos sobre el proceso automatizado, con la orden de un controlador generan la señal para activar un elemento final de control. Existen diferentes tipos de actuadores tales como: Electrónicos, Neumáticos, Hidráulicos y Eléctricos. El tipo de actuador utilizado es el Eléctrico debido a su simplicidad, requiriendo solo de energía eléctrica como fuente de poder.

El proyecto consta de dos electroválvulas y de un sistema de agitación, los cuales pueden ser activadas dependiendo del estado en que se encuentra el sistema.

Electroválvulas

Las válvulas seleccionadas para el proceso de control de flujo son las Solenoid Valve 2P025 – 06 que trabajan bajo 12V DC. Para la implementación del prototipo se consiguió las electroválvulas proveídas por la empresa Quiteña Owan Electronics, que debido a su pequeño tamaño proporcionan un fácil manejo y adaptación al sistema. Este actuador sirve como mecanismo de regulación del pH de la solución, se

han definido dos sustancias para equilibrar el pH: el hidróxido de sodio (NaOH) que aumenta el pH de la mezcla (Base) y el Ácido clorhídrico (HCl) que realiza la acción contraria disminuyendo el pH (Ácido). En la Figura 3.6 se observa el modelo de electroválvulas escogidas las cuales cuentan con las siguientes especificaciones:

Modelo de Válvula	2P025-06
Tipo	2 vías, normalmente cerrada
Operación	Acción directa, tiempo de respuesta < 20ms
Orificio	2.5 mm
Tamaño del puerto	1/8"
Presión Operacional	0 – 0.7MPa
Máxima resistencia a la presión	1.05MPa
Rango de Temperatura Operacional	-5 a 80 °C
Voltaje Convencional	12V DC
Material del Cuerpo	Engineerd Nylon
Fluido	Agua, Aceite, Liquido, Gas

Tabla 2: Características electroválvulas 2P025-06 [17]



Figura 3.7: Electroválvula 2P025-06 [18]

El transporte de los líquidos que son enviados desde los contenedores de estas sustancias hacia el biodigestor se realiza a través de mangueras que son adaptadas a las electroválvulas por medio de un material que permite la unión de los mismos conocidos como neplo.

Para la conmutación de la electroválvula es necesario utilizar un pequeño circuito electrónico como método de conexión entre la tarjeta de desarrollo controladora y la electroválvula. Posteriormente se explica la estructura y funcionamiento del circuito.

Sistema de agitación

Este sistema está formado por un motor y una estructura metálica de una barra cilíndrica acoplada por un lado a un ventilador que será el agitador y por el otro lado ajustado al motor logrando de esta manera producir un movimiento constante homogéneo en toda la mezcla.

Motor paso a paso

Dado que el biodigestor debe operar continuamente durante el proceso de fertilización, es necesario un motor que realice constantemente este trabajo por lo cual se consideró un motor paso a paso bipolar debido a que en el mercado nacional los motores AC y DC encontrados no cumplían con las necesidades del proyecto, mientras que el motor elegido ofrece una facilidad en cuanto al control de su velocidad y giro.

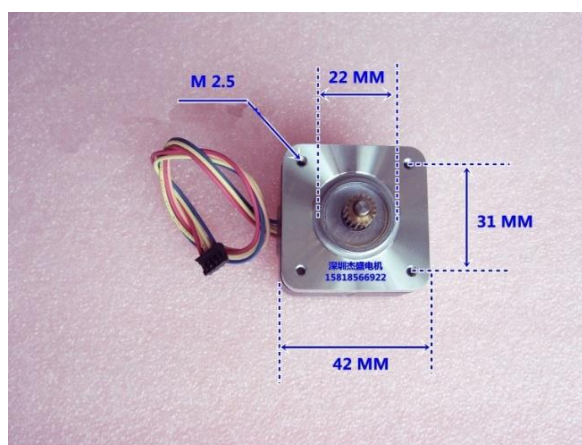


Figura 3.8: Motor paso a paso modelo 17PM-K054-G7WS [19]

En la Figura 3.7 se muestra el motor seleccionado para el cumplimiento de esta función, con algunas especificaciones detalladas a continuación:

Modelo	17PM – K054 – G7WS
Especificaciones del motor	42 x 42 mm
Altura del motor	33,3 mm
Eje de Salida	4,4 mm
Fase	2
Tipo	Híbrido
Angulo de Paso	1.8 °
Corriente	0.6 A - 1 A
Voltaje	5 V
Resistencia interna	2.5 Ohm
Numero de Hilos	4 líneas (donde 1 y 3 son la fase A, y 2 y 3 son de la fase B). Rojo y Amarillo es un conjunto de líneas, rosa y azul es otro grupo de líneas.

Tabla 3: Especificaciones del motor 17PM-K054-G7WS [20]

Para poder operar el motor a través de la BeagleBone Black se utilizó un circuito integrado, el Easy Driver adquirido por medio de Enix Robotics, este circuito permite programar directamente desde la BeagleBone una rutina con la secuencia de paso para poner en marcha el motor en conjunto con la estructura acoplada. En la Figura 3.8 se observa el Easy Driver detallando en la siguiente tabla la descripción de los pines:

Pines	Descripción
GND	Existen 3 pines GND (Tierra). Están interconectadas dentro de la tarjeta. Conectar el lado negativo de la fuente de poder a este pin.
M+	Es la entrada de poder a Easy Driver. Conectar este pin al lado positivo de la fuente de poder. Debe estar entre 6V a 30V con 2A (o más)
A y B (4 pines)	Son las conexiones del motor. Observar el diagrama de conexión del motor.
STEP (Paso)	Necesita estar entre 0V a 5V (o 0V a 3.3V si se lo configura de ese modo)
DIR (Dirección)	Necesita estar entre 0V a 5V (o 0V a 3.3V si se lo configura de ese modo). El nivel de la señal (Alto/Bajo) indica el sentido de giro.
MS1/MS2	Controlan el modo micropaso. Las posibles configuraciones (MS1/MS2): paso completo (0,0), ½

	paso (1,0), ¼ paso (0,1), 1/8 paso (1,1: predeterminado).
RST (reset)	Señal normalmente en alto, restablecerá el traductor interno y deshabilitará todas las salidas del driver cuando se pone en bajo.
SLP (sleep)	Señal normalmente en alto, minimizará la potencia de consumo deshabilitando al circuito interno y las salidas del driver cuando se pone en bajo.
ENABLE	Señal normalmente en bajo, deshabilitará todas las salidas cuando se ponga en alto.
PFD	Observar la hoja de datos para más información.
5V	Es una salida que provee tanto 5 V(predeterminadamente) como 3.3 V y suministra de una corriente de (50 mA dependiendo del voltaje de entrada)

Tabla 4: Descripción de los pines Easy Driver [21]

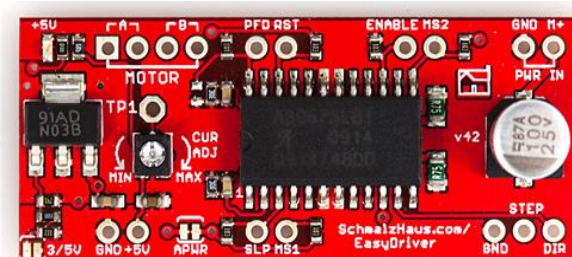


Figura 3.9: Easy Driver v 4.2 [21]

Luz piloto de neón

Es una lámpara que opera a partir de 12 V de Corriente Directa o más tiene bajo consumo de energía por lo cual ha sido considerada como un indicador de emergencia en el caso de que alguno de los medidas de la mezcla del biodigestor excede el rango normal.



Figura 3.10: Luz Piloto de Neón [22]

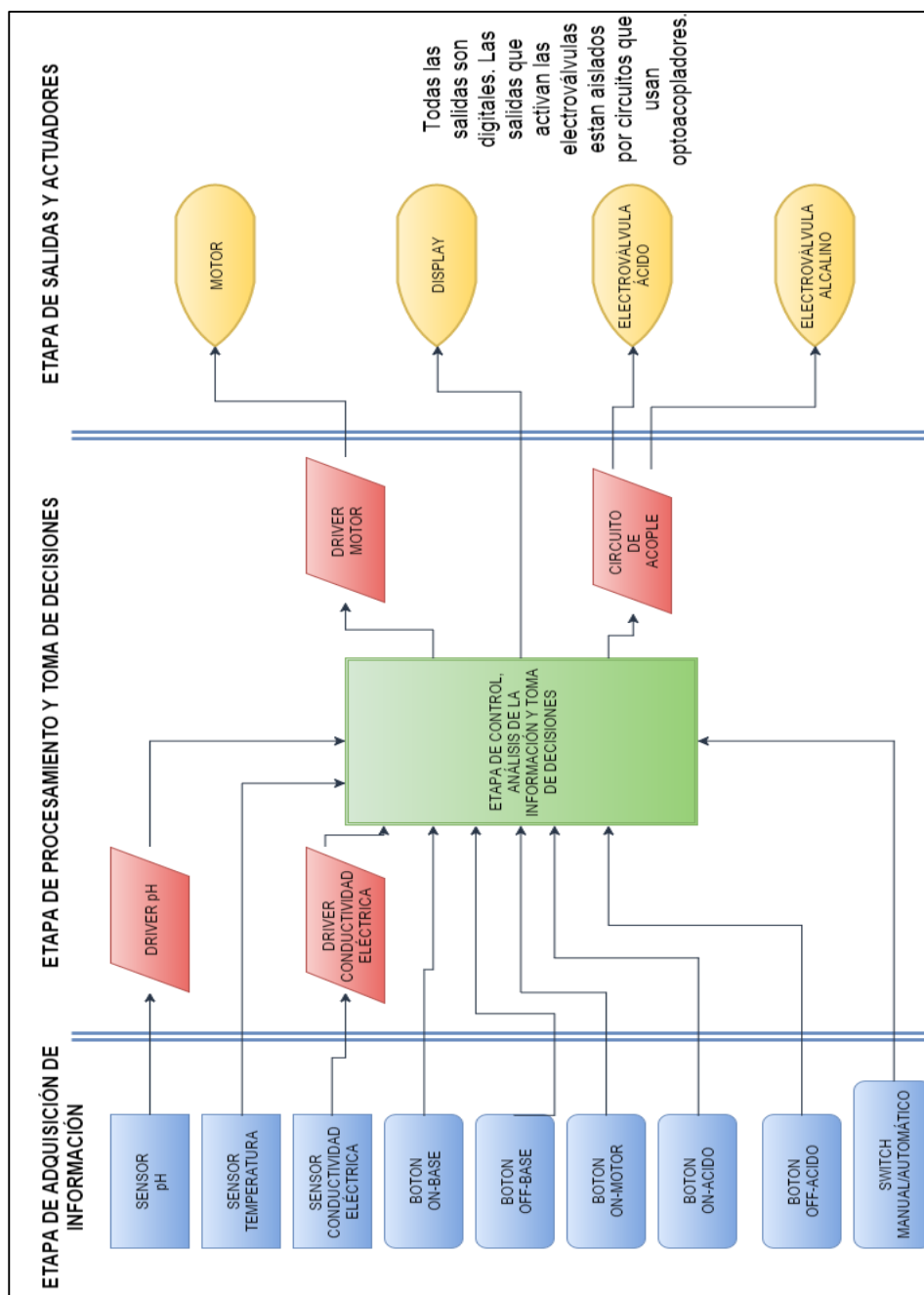


Figura 3.11: Diagrama de bloques del Sistema de Control

En la Figura 3.11 se observa un diagrama de bloques que muestra los elementos que conforman el sistema dando una visión completa del diseño propuesto.

3.3 Descripción del software

El software diseñado para el proyecto comprende dos partes, el código programado en la BeagleBone y la página web predestinada a mostrar toda la información del sistema.

Rutinas de la beaglebone black

Como lenguaje de programación principal se utiliza Python versión 2.7.3 que es un lenguaje de programación interpretado de alto nivel y orientado a objetos, de programación imperativa usa tipado dinámico y es multiplataforma, permitiendo una sencilla programación de rutinas. La BeagleBone cuenta con este lenguaje instalado de manera predeterminada en su sistema.

El código ha sido modularizado de acuerdo a las funciones específicas del proyecto por eso se ha segmentado el programa principal de la siguiente manera:

Obtención de temperatura

Para obtener el valor de la temperatura proveniente del sensor se utilizó el Pin 22 de P9 de la tarjeta para lo cual hay que compilar el árbol de dispositivo para que el mismo proporcione al kernel la dirección y el tipo del hardware, el código necesario se muestra en la Figura 3.12.

```

/dts-v1/;
/plugin/;

/ {
    compatible = "ti,beaglebone", "ti,beaglebone-black";

    part-number = "BB-W1";
    version = "00A0";

    /* state the resources this cape uses */
    exclusive-use =
        /* the pin header uses */
        "P9.22",
        /* the hardware IP uses */
        "gpio0_2";

    fragment@0 {
        target = &am33xx_pinmux;
        __overlay__ {
            dallas_w1_pins: pinmux_dallas_w1_pins {
                pinctrl-single,pins = <
                    0x150 0x37
                >;
            };
        };
    };

    fragment@1 {
        target = &ocp;
        __overlay__ {
            onewire@0 {
                compatible = "w1-gpio";
                pinctrl-names = "default";
                pinctrl-0 = <&dallas_w1_pins>;
                status = "okay";

                gpios = <&gpio1 2 0>;
            };
        };
    };
};

```

Figura 3.12: Archivo BB-W1-00A0.dts para crear dispositivo de sensor de temperatura.

El archivo BB-W1-00A0.dts se lo debe guardar en el directorio y para compilar y crear la instancia respectiva correspondiente al sensor se deben ejecutar los siguientes comandos en la Shell de Linux dentro del mismo directorio donde se guardó el archivo antes mencionado. En la Figura 3.13 se aprecian los comandos.

```

dtc -O dtb -o BB-W1-00A0.dtbo -b 0 -@ BB-W1-00A0.dts
echo 'BB-W1:00A0' > '/sys/devices/bone_capemgr.9/slots'

```

Figura 3.13: Comandos para compilar dispositivo para sensor de temperatura.

El código genera un número de serie único correspondiente al sensor con el cual se pueden hacer las lecturas de las temperaturas.

```
import time

w1="/sys/bus/w1/devices/28-000006886a4c/w1_slave"

while True:
    raw = open(w1, "r").read()
    print "Temperature is "+str(float(raw.split("t=")[-1])/1000)+" degrees"
    time.sleep(1)
```

Figura 3.14: Rutina para obtener datos del sensor de temperatura.

En la Figura 3.14 se observan las líneas de código utilizadas, la instancia w1 se usa para llamar al sensor de temperatura DS18B20 y dentro del lazo while se realiza una lectura de esta instancia y posteriormente una conversión del tipo de dato obtenido.

Obtención de pH y conductividad eléctrica

La interfaz de comunicación escogida para la transmisión y recepción de los datos de los sensores entre los driver y la BeagleBone fue el protocolo UART, aprovechando que ambos controladores disponen de este hardware. Es un protocolo serie muy simple, usa una línea de datos para transmitir y otra para recibir datos. Los parámetros para establecer la comunicación UART son:

- Velocidad de transmisión: 9600 bps
- Bits de datos: 8
- Bit de parada: 1
- Sin paridad.
- Sin control de flujo.

En la Figura 3.15 se describe el código para la adquisición de los valores del sensor de conductividad eléctrica que además incluye las mediciones de total de sólidos disueltos y salinidad.

```

import Adafruit_BBIO.UART as UART
import serial
import csv

UART.setup("UART1")

ser = serial.Serial(port = "/dev/ttyO1", baudrate=9600)
ser.write("L,1\r")
ser.write("C,1\r")
line = ""
data = ""
while True:
    data = ""
    try:
        while(data != "\r"):
            data = ser.read()
            if (data != "\r"):
                line = line + data
    except serial.SerialException:
        pass
    if(data == "\r"):
        if(line[0:1] != ""):
            reader = csv.reader(line.split('\n'), delimiter=',')
            for row in reader:
                ec = row[0]
                tds = row[1]
                s = row[2]
                print "EC: " + ec
                print "TDS: " + tds
                print "SAL: " + s
            line = ""

```

Figura 3.15: Rutina de Programación para obtención de Datos del Sensor de Conductividad Eléctrica.

El manejo del puerto serial se realiza mediante el llamado de la librería `Adafruit_BBIO.UART`, luego se consideró los pines de la BeagleBone que trabajan en modo UART y se estableció que para este caso se manipularía el UART1 correspondiente a los pines 24 y 26 de P9 para la recepción (RX) y transmisión (TX) respectivamente.

Para habilitar el puerto serial correspondiente al UART seleccionado, es necesario conocer la dirección del dispositivo para lo cual se usó la Tabla 5.

	RX	TX	Dispositivo
UART0	J1_4	J1_5	/dev/ttyO0
UART1	P9_26	P9_24	/dev/ttyO1
UART2	P9_22	P9_21	/dev/ttyO2
UART3		P9_42	/dev/ttyO3
UART4	P9_11	P9_13	/dev/ttyO4
UART5	P8_38	P8_37	/dev/ttyO5

Tabla 5: Correspondencia Puertos Seriales / UART BeagleBone Black. [23]

Debido a que el driver del sensor devuelve los datos en cadena de string separado por comas conocido como string csv, para la lectura de este tipo de datos se utilizó la librería csv de Python y con la función de split se obtuvieron los datos individuales de cada medida.

En el caso de la lectura del sensor de pH se decidió habilitar el puerto serial UART4. A diferencia del sensor de conductividad, el formato de los datos es un simple String lo que hace más fácil la adquisición del dato.

```
import Adafruit_BBIO.UART as UART
import serial
import StringIO

UART.setup("UART4")

ser = serial.Serial(port = "/dev/ttyO4", baudrate=9600)
ser.write("L,1\r")
ser.write("C,1\r")
line = ""
data = ""
while True:
    data = ""
    while(data != "\r"):
        data = ser.read()
        if (data != "\r"):
            line = line + data
    if(data == "\r"):
        if(line[0:1] != "*"):
            ph = line
            print "pH: " + ph

    line = ""
```

Figura 3.16: Rutina de programación para la obtención de datos de pH.

En la Figura 3.16 se muestra la rutina programada para la obtención del valor de pH.

Tanto los sensores de pH como de conductividad eléctrica deben ser calibrados una vez al año, para realizar esta actividad en el caso de la conductividad se hizo una calibración de doble punto ejecutando los siguientes comandos:

`ser.write("Cal,dry\r").` Ejecutado al principio de la calibración. La sonda debe estar seca.

`ser.write("Cal,low,12880\r").` Establecer el valor con la conductividad más baja de las soluciones para la calibración.

`ser.write("Cal,high,80000\r").` Establecer el valor con la conductividad más alta de las soluciones para la calibración.

Similarmente para el caso de la calibración del pH, se ejecutaron los siguientes comandos:

`ser.write("Cal,mid,7.00\r").` Establecer el valor con un pH medio, el agua.

`ser.write("Cal,low,4.00\r").` Establecer el valor con el pH más bajo de las soluciones para la calibración.

`ser.write("Cal,high,10.00\r").` Establecer el valor con el pH más alto de las soluciones para la calibración.

Los valores asignados para la calibración en ambos casos fueron considerados debido a que las soluciones adquiridas para la calibración correspondían a dichos valores.

Base de datos local

Se instaló una base de Datos usando MySQL, la funcionalidad de esta medida es utilizarla como almacenamiento de los usuarios y de los valores obtenidos de los sensores. Para mayor agilidad del sitio web se accederán a los datos actualizados en la base de datos. Cabe mencionar algunas características importantes de la base como por ejemplo que está almacenada dentro de la memoria SD de la BeagleBone, en la Figura 3.17 se detallan sus parámetros para la conexión:

```

import MySQLdb

DB_HOST = 'localhost' #Dirección de la Base de Datos
DB_USER = 'root'      #Usuario para conexión a la Base de Datos
DB_PASS = 'root'      #Contraseña para conexión a la Base de Datos
DB_NAME = 'Biodigestor' #Nombre de la Base de Datos

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()        # Crear un cursor
cursor.execute(query)         # Ejecutar una consulta

```

Figura 3.17: Conexión base de Datos Local BeagleBone Black.

Rutina para movimiento del motor paso a paso.

El motor paso a paso desempeña un rol importante dentro del biodigestor en vista de que forma parte del sistema de agitación para mantener la mezcla uniforme. La rutina programada para proceder con el movimiento del motor y por ende de la estructura del sistema de agitación se analiza en la Figura 3.18.

```

import Adafruit_BBIO.GPIO as GPIO
import time

#Dir = P9_12
#Step = P8_7
#MS1 = P8_9
#MS2 = P8_26

def motorpaso():
    GPIO.setup("P9_12", GPIO.OUT)
    GPIO.setup("P8_7", GPIO.OUT)

    GPIO.setup("P8_9", GPIO.OUT)
    GPIO.setup("P8_26", GPIO.OUT)

    GPIO.output("P8_9", GPIO.LOW)
    GPIO.output("P8_26", GPIO.LOW)

    GPIO.output("P9_12", GPIO.HIGH)

    for i in range(0,4000):
        #while True:
            GPIO.output("P8_7", GPIO.HIGH)
            time.sleep(0.00074)
            GPIO.output("P8_7", GPIO.LOW)
            time.sleep(0.00074)

        time.sleep(0.5)
        GPIO.output("P9_12", GPIO.LOW)

    for i in range(0,4000):
        GPIO.output("P8_7", GPIO.HIGH)
        time.sleep(0.00074)
        GPIO.output("P8_7", GPIO.LOW)
        time.sleep(0.00074)

```

Figura 3.18: Rutina de programación para motor paso a paso.

La velocidad de giro del motor paso a paso que se estableció fue de 100 rpm. Puesto que el motor es de 1.8 grados cada paso completo requiere de un ciclo eléctrico de 360 grados, para el caso $360/1.8 = 200$ pasos completos por revolución mecánica, es decir 800 pulsos de entrada por cada revolución del eje, dado que son 4 pulsos por ciclo eléctrico $200 \times 4 = 800$. Los 100 rpm equivalen a 1.67 rps, de acuerdo a esto se necesitan aproximadamente $800 \times 1.67 = 1336$ pulsos de entrada por segundo para obtener los 100 rpm, de manera que el tiempo establecido en el código fue de $t = 1/1336 = 0.00075$ segundos. Esto trabajando con el motor en modo "full step" (paso completo), por lo cual a MS1 y MS2 se les asignaron los valores de 0 y 0. El número 4000 que se establece en el lazo for es para que se den 20 vueltas continuas al terminar cada lazo.

Notificación de alarmas mediante correo electrónico

Las alarmas a través de correos electrónicos son muy útiles en el entorno laboral debido a que tenemos nuestros buzones de correos no solamente sincronizados en nuestros ordenadores sino también en los dispositivos móviles. Por lo cual esta función nos pareció muy aprovechada para este proyecto. En la Figura 3.19 se observa la programación necesaria para habilitar esta función con la BeagleBone posteriormente se hace un explicación del código.

```
import smtplib
from email.mime.text import MIMEText

def alertMe(subject, body):
    myAddress = "apelior@gmail.com"
    addressSend = "apelior_92@hotmail.com"
    msg = MIMEText(body)
    msg['Subject'] = subject
    msg['From'] = myAddress
    msg['Reply-to'] = myAddress
    msg['To'] = addressSend

    server = smtplib.SMTP('smtp.gmail.com',587)
    server.starttls()
    server.login(myAddress, 'passwordxxxx')
    server.sendmail(myAddress, addressSend, msg.as_string())
    server.quit()
```

Figura 3.19: Rutina para enviar notificaciones de los datos de los sensores mediante correo electrónico.

Inicialmente se llaman a las librerías respectivas de Python que nos proporcionaran las funcionalidades necesarias para lograr enviar alertas de correo cuando alguno de los parámetros indicadores se encuentren fuera de rango, posteriormente se ha creado una función llamada alertMe (Subject,body) que recibe tanto el asunto como el cuerpo que forman el correo, dentro de esta función se define el correo al que se desea lleguen las notificaciones y también una dirección de correo desde la cual se envían todas las alertas, luego se establece la configuración particular del servidor SMPT de correo del remitente , conjuntamente con la contraseña del mismo.

Archivo python “biodigestor.py”

Este archivo conecta a la base de datos para almacenar dentro de esta las lecturas de los valores de los sensores provenientes del sistema de control automático para posteriormente cargarlos en la página web y también mostrarlos a través de la LCD, además se habilitan las entradas y salidas digitales de la BeagleBone para accionar mediante las botoneras los circuitos de las electroválvulas, el motor y la luz de alerta.

Archivo python “login.py”

Este archivo tiene varias funcionalidades, entre las cuales levanta un servidor web a través de un microframework llamado Flask el cual tiene una fácil instalación y configuración en la BeagleBone. Con la función render_template() de Flask se cargan los archivos html y css de las páginas web para ser alojadas en el servidor interactuando así de una manera dinámica.

Otra de las funciones de este archivo es enlazar la página web con la base de datos y la BeagleBone a través de la web, estableciendo sesiones entre cliente y servidor.

Con la ayuda de la base de datos se presentan los datos en la página web y cada función de la infraestructura del código programado es enlazada con la página a través del comando @app.route("/") que establece una ruta solicitada al servidor web, tanto para redirigir a otras páginas como también ejecutar alguna función en específico.

Autoarranque del programa

Con la finalidad de no necesitar gestión del administrador para poner en ejecución todos los programas que inician el sistema, se convertirán todos los archivos .py necesarios en servicios que se ejecutarán al arrancar el sistema operativo incrustado en la tarjeta de desarrollo.

Estadísticas de datos

Desde la página web existe la opción de poder visualizar de manera gráfica y ordenada las estadísticas de los datos proporcionados por los sensores a través del acceso a la base de datos junto con una función facilitada por google chart permiten de esta manera obtener un gráfico más comprensivo y útil para el usuario. Además se generan reportes de los parámetros de medición obtenidos de los sensores en un archivo pdf.

En la Figura 3.20 se presentan los datos del sensor de temperatura mostrados de forma gráfica.

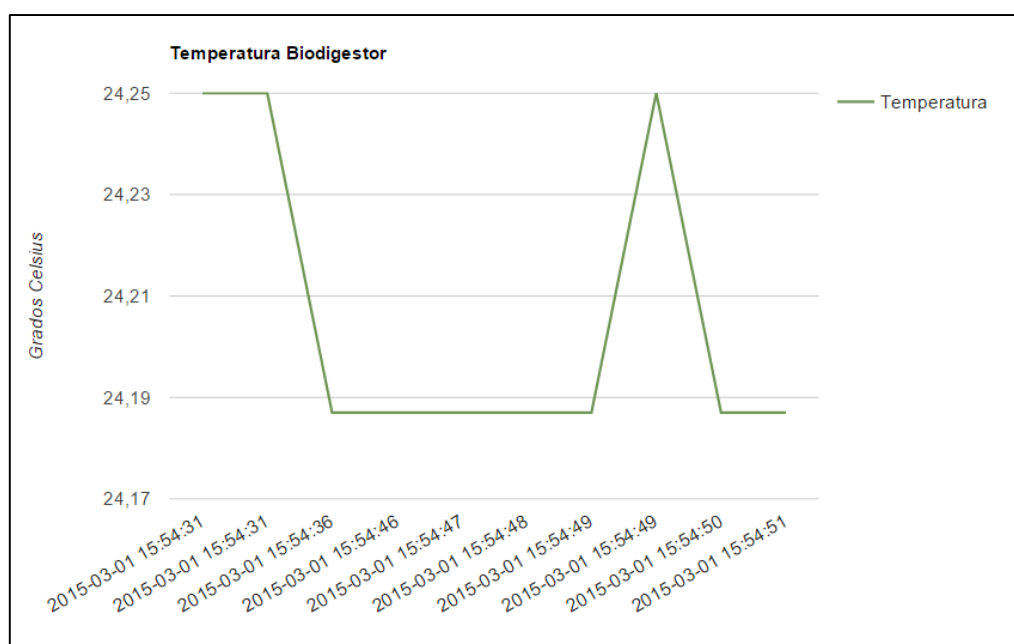


Figura 3.20: Gráfico Estadístico Temperatura Biodigestor

3.4 Descripción del hardware

En este apartado se describirá el circuito que se ha utilizado para poder accionar los actuadores.

Para el desarrollo del control tanto de las electroválvulas como de la luz indicadora de alerta en el tablero local se utilizó un módulo de 4 canales de relés de las características se describen a continuación [24]:

- Alto nivel de disparo.
- 12V 4-Canales de Relé, Interfaz de tarjeta, cada uno necesita 50-60mA para control.
- LEDs de indicación de estado de la salida del relé.
- Máxima Carga: AC 250V/10A, DC 30V/10A
- Tamaño del módulo: 7.5cm x 5.5cm x 1.7cm



Figura 3.21: 12V Relay Module 4-Channel [25]

El circuito interno del módulo se lo observa en la Figura 3.21 se observa que se tiene dos tiras de pines, una que es donde se encuentra el jumper, que tiene dos pines y se puede seleccionar alimentar todo el módulo de forma conjunta, es decir seleccionar con el jumper JD-VCC y VCC, o bien alimentar de forma independiente tanto optoacopladores como relés quitando el jumper, se ha usado la forma de energizar de manera separada, el pin de JD-VCC se lo ha polarizado con 12 V DC. La otra tira de pines aparecen en el orden descrito a continuación: GND IN1 IN2 IN3 IN4 VCC, aquí es donde unimos la alimentación con la BeagleBone Black, las entradas IN1, IN2, IN3 activan las bobinas de los relés

cuando ponemos en bajo dichas entradas, conectados a los pines que se han programado para control de las dos electroválvulas y foco.

Por otro lado se tiene las borneras a los que conectaremos los actuadores antes mencionados, siguiendo la conveniencia de un relé en el cual se tiene el contacto normalmente cerrado, normalmente abierto y común.

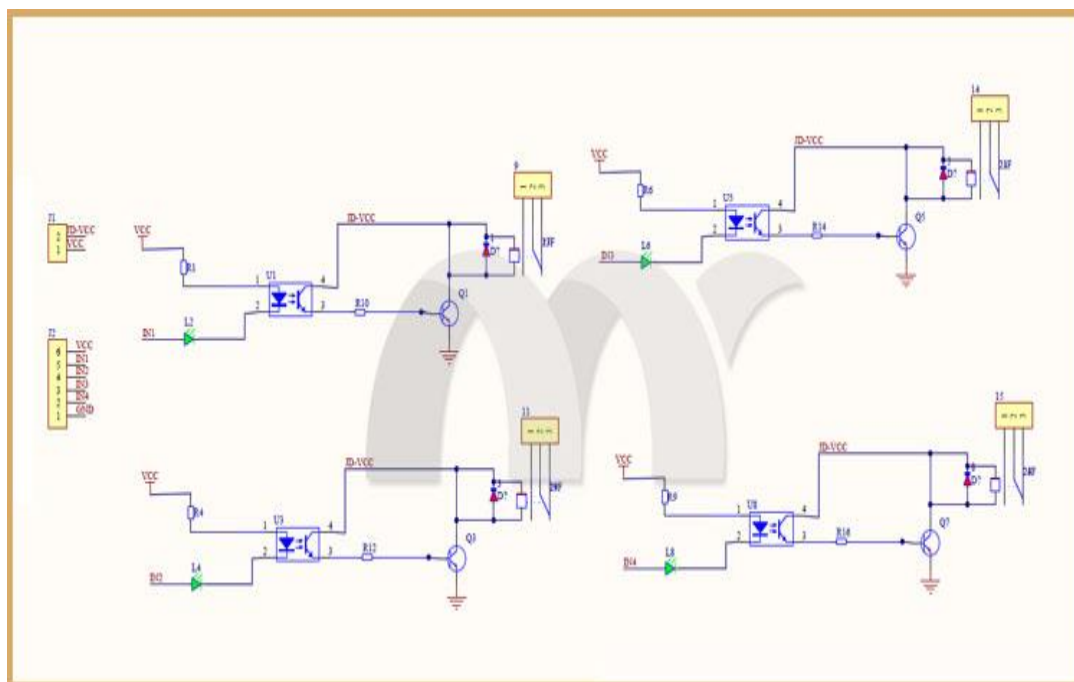


Figura 3.22: Circuito Módulo 4 - Canales de Relé [25]

Con el fin de evitar los picos de voltaje y corriente producidos por el accionamiento de los contactos del relé se ha puesto un diodo 1n4004 en paralelo a la carga es decir entre el contacto del relé y GND, este diodo es de gran ayuda debido a que suprime las corrientes parásitas que se producen por efecto del relé y que podrían afectar a circuitos integrados sensibles como por ejemplo la LCD e incluso a la tarjeta de desarrollo.

3.5 Interfaz humano máquina

La interfaz entre el sistema de control y el operador del biodigestor consta de dos escenarios, una página web y un pequeño panel eléctrico. El panel eléctrico gestiona las actividades del sistema de manera local a través de botones y presenta los datos por medio de una pantalla LCD. La página web se utiliza para

la administración remota por medio de una conexión a internet en donde se muestran los datos y el estado de funcionamiento del biodigestor, tiene tres componentes principales, la página web como tal, una base de datos donde se registran las variables y se almacena los datos de los usuarios para establecer las sesiones y un archivo programado en lenguaje Python que enlaza la página web con la BeagleBone Black y la base de datos, cada sección se expone en los siguientes apartados.

Interfaz de página web

Para el diseño de la página web se utilizó el lenguaje HTML en conjunto con la hoja de estilos CSS consiguiendo de esta manera mostrar una página dinámica e interactiva con un mayor grado de presentación estructurada, además el diseño es adaptable a ciertas pantallas de dispositivos móviles, tablets, laptops. La página web consta de varias pantallas en las cuales el usuario puede encontrar información acerca de lo que trata el proyecto o contactar por medio de correo electrónico a los desarrolladores, adicional a las paginas informativas, hay una pantalla de inicio de sesión en donde los usuarios registrados ingresan su usuario y contraseña para luego dirigirse a una página que muestra el estado de trabajo y el valor de los sensores de la planta, como se observa en la Figura 3.23 se aprecian todos los componentes de diseño y cada una de las válvulas, sensores y motor.

En el lado derecho de la imagen se muestran los datos de todos los sensores instalados en el biodigestor, se utilizó la técnica AJAX para la actualización de los datos de forma periódica en tiempo real la cual es controlada por la BeagleBone por lo que cada vez que existe información nueva la BeagleBone envía dichos datos a la base de datos y esta actualiza cada una de las variables.

En el lado izquierdo de la imagen se muestran los botones on/off para accionar las válvulas de las soluciones tanto ácida como base, así como el botón de encendido del motor.



Figura 3.23: Página Web del Proyecto

Los usuarios han sido categorizados en dos tipos, Administrador y Operario. Ambos pueden operar el sistema pero adicionalmente existe una opción de parámetros en la página en donde sólo los usuarios administradores pueden cambiar los rangos de valores máximos y mínimos de las variables.

Luego de que el servicio web se haya puesto en marcha para facilitar el acceso a través de los navegadores web se habilitó además un servidor DNS local maestro en la BeagleBone con el dominio “mibiodigestor.com”. Para un escenario de producción se debería adquirir un dominio público o también utilizar los servicios gratuitos que se ofrecen en Internet, con lo cual se podría acceder a la interfaz desde cualquier punto.

Interfaz local

Como se indicó anteriormente, esta parte consiste de un pequeño panel eléctrico para el cual se ha elegido una caja plástica electrónica, botones pulsadores para accionar las electroválvulas y el motor, un switch de balancín para habilitar el modo automático del sistema y una pantalla LCD modelo QC2004A de 20 caracteres y 4 líneas para presentar los datos de los sensores. Adicionalmente tiene una luz de neón piloto que se activará como alarma local cuando uno de los

parámetros no se encuentre dentro del rango normal. La conexión de los pines de la LCD a los pines de la BeagleBone Black se especifica en la Tabla 6:

BeagleBone Black	LCD QC2004A
GND	Pin 1 (VSS), Pin 5 (R/W) y Pin 16 (K)
VCC (5V)	Pin 2 (VDD) y Pin 15(A)
P8_8	Pin 4 (RS)
P8_10	Pin 6 (Clock/Enable)
P8_12	Pin 11 (DB4)
P8_14	Pin 12 (DB5)
P8_16	Pin 13 (DB6)
P8_18	Pin 14 (DB7)

Tabla 6: Interconexión entre pines LCD y BeagleBone Black.



Figura 3.24: LCD QC2004A [26]

En la Figura 3.24 se muestra la LCD utilizada, el uso de los pines se especifica a continuación:

- **VSS:** Pin negativo (GND)
- **VDD:** es la alimentación principal de la pantalla, funciona a 5V.
- **V0:** es el contraste de la pantalla, debe conectarse con un potenciómetro de unos 10K Ohms.

- **RS:** Es el selector de registro, para mostrar caracteres o enviar comandos de control.
- **RW:** Es el pin que controla la Lectura/Escritura cuando la pantalla está enviando o recibiendo.
- **E:** Es Enable, habilita la pantalla para recibir información.
- **DB0-DB7:** Son la línea del bus de datos por donde se envían los bits de datos al LCD.
- **A y K:** Son los pines de led de la luz de fondo de pantalla. A (5V) y K (GND)

3.6 Financiamiento

La inversión total del proyecto es la que comprende la compra de los materiales para la construcción del prototipo como tal, comienza con la adquisición de los componentes esenciales tales como sensores, actuadores y controlador. Posteriormente se adquirieron materiales para la maquetación del proyecto.

En la Tabla 7 se muestra detalladamente los costos de la adquisición de los materiales equipos obtenidos para la implementación, cabe mencionar que debido a que muchas de los componentes del proyecto no se los podían conseguir dentro del país se los mandó a traer de Estados Unidos dándole un valor agregado por gastos de envío, todos los gastos han sido asumidos por los autores del proyecto.

Cantidad	Descripción	Precio Unitario	Precio Total
1	BeagleBone Black	\$ 67,00	\$ 67,00
1	Caja plastica electronica 223x139x92mm	\$ 17,50	\$ 17,50
1	Atlas Scientific Conductivity Probe K 1.0	\$ 126,00	\$ 126,00

1	EZO Electrical Conductivity Circuit	\$ 52,00	\$ 52,00
1	American Marine PINPOINT pH Probe	\$ 40,00	\$ 40,00
1	pH CIRCUIT	\$ 34,00	\$ 34,00
1	Sensor De Temperatura Digital Ds18b20	\$ 14,00	\$ 14,00
2	Pre-Assembled Female BNC x 2	\$ 11,00	\$ 22,00
1	LCD 20x4	\$ 15,00	\$ 15,00
1	easy driver	\$ 14,00	\$ 14,00
1	motor paso a paso	\$ 12,00	\$ 12,00
1	Aspas ventilador bomba de agua	\$ 3,00	\$ 3,00
5	Botoneras Arcade	\$ 1,20	\$ 6,00
1	luz piloto	\$ 2,80	\$ 2,80
2	Electrovalvulas 1/8 12V DC	\$ 31,50	\$ 63,00
1	Manguera 1/4	\$ 0,75	\$ 0,75
4	Neplo	\$ 1,50	\$ 6,00
1	Módulo Relé 4 Canales 12 VDC	\$ 19,00	\$ 19,00
30	Cable Jumper conector macho macho	\$ 0,15	\$ 4,50
20	Cable Jumper conector macho hembra	\$ 0,15	\$ 3,00
1	Frasco de Vidrio - Contenedor	\$ 16,00	\$ 16,00
1	Gastos de Envío	\$ 118,00	\$ 118,00
3	Mini Protoboard	\$ 1,50	\$ 4,50
2	Envase plástico	\$ 1,68	\$ 3,36

3	1n4004	\$ 0,05	\$ 0,15
1	Estructura de Madera	\$ 35,00	\$ 35,00
TOTAL			\$ 698,56

Tabla 7: Costo del Proyecto.

3.6.1 Análisis Comparativo Cualitativo - Económico

El prototipo desarrollado se lo diseñó para un entorno de laboratorio por lo que se realizó un análisis comparativo con el dispositivo utilizado en el área de los laboratorios del CIBE el FoxyLogic. En la Tabla 8 se muestra una comparativa de precios y características.

Característica	FoxyLogic	Prototipo
Interfaz Web	No	Si
Interfaz de manejo local	Si	Si
Sistema de enfriamiento	Si	No
Control automático de pH	Si	Si
Sistema de Agitación	Si	Si
Sensores y adquisición de datos	Si	Si
Base de Datos para registro de datos	Si	Si

Tabla 8: Comparación cualitativa FoxyLogic y Prototipo

En el mercado equipos con características similares tienen un precio de \$ 13,000 dólares americanos por lo que comparando con nuestro prototipo en el cual se invirtió cerca de los \$ 700 dólares americanos se evidencia un ahorro significativo, lo representativo del prototipo es que tiene una interfaz web permitiendo al usuario manejo remoto del sistema. En ambos dispositivo se puede llevar un control de calidad del abono orgánico producido sin embargo lo negativo del prototipo diseñado es que no se ofrece ninguna solución respecto a la temperatura debido a que no posee un sistema de enfriamiento.

CAPÍTULO 4

4. RESULTADOS

4.1 Prototipo en funcionamiento

En el presente apartado se muestra a través de imágenes las distintas funcionalidades del proyecto y como es el resultado final después de su diseño y construcción.

En la Figura 4.1 se muestra el prototipo finalizado con la estructura de madera, el recipiente del biodigestor y el tablero de administración local.



Figura 4.1: Prototipo Finalizado

El tablero de administración local se observa en la Figura 4.2 en la cual se visualiza que en la pantalla de LCD se muestran los parámetros de medición de calidad de la mezcla, también se aprecia uno de los actuadores que es la luz piloto que nos indica si alguno de los parámetros está fuera del rango normal.



Figura 4.2: Tablero de Administración local Prototipo

Lo que respecta a los sensores y los demás actuadores se pueden apreciar en la Figura 4.3 en la cual tenemos una vista superior del envase pudiendo observar el sensor de pH, conductividad eléctrica y de temperatura, adicionalmente también se distinguen las electroválvula y el sistema de agitación que principalmente lo forma el motor.



Figura 4.3: Vista Superior de Envase de Prototipo

El sistema de agitación es visible en la Figura 4.4 en la cual se aprecia las aspas que fueron acopladas al eje del motor y que sirven para agitar la mezcla y como parte del proceso para homogeneizar la misma.



Figura 4.4: Sistema de Agitación del Prototipo

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Luego de haber culminado el presente proyecto se han determinado las siguientes conclusiones:

- La implementación del prototipo tuvo resultados satisfactorios debido a que se pudo medir los parámetros indicadores de temperatura, pH, Conductividad Eléctrica tal como se había establecido.
- Los modos de administración del sistema manual/automático facilitan el manejo para el usuario permitiendo realizar actividades tanto de pruebas como de mantenimiento.
- Los dispositivos seleccionados para la construcción del prototipo son los más adecuados de acuerdo a los requerimientos del proyecto.
- La página web ofrece una interfaz simple y amigable del sistema, permitiendo al usuario una administración más sencilla.
- El manejo del sistema puede ser tanto local mediante el panel de botones y LCD como remoto mediante la página web.

Recomendaciones

Se recomienda en la medida de la posible lo siguiente:

- Fortalecer el sistema añadiéndole más sensores tales como Oxígeno Disuelto, Grados Brix, densidad para un control más eficaz del Biol.
- Buscar los implementos necesarios para adaptar el sistema en un ambiente de campo, por ejemplo considerar el uso de sensores industriales y motores y electroválvulas de mayor potencia.
- Generar reportes diarios de las mediciones obtenidas para tener una visión más general del desempeño del sistema.
- Para futuras implementaciones utilizar un software que sea orientado a ambientes de producción por ejemplo implementar un servidor web apache o

similar. Además se recomienda adquirir un nombre de dominio para poder acceder a la página de una red externa.

- Tener precaución con la alimentación de los circuitos del hardware, cuidado con las sobretensiones, teniendo en cuenta que la mayoría de los implementos utilizados trabajan con voltaje de 3.3 V y 5 V.

BIBLIOGRAFÍA

- [1] El Universo, «El Universo,» 13 Diciembre 2014. [En línea]. Available: <http://www.eluniverso.com/noticias/2014/12/13/nota/4335416/politecnicos-aplicaran-mezcla-combatir-plaga-cacao>. [Último acceso: 20 Julio 2015].
- [2] A. C. L. Pérez, «Academia,» 28 Mayo 2015. [En línea]. Available: http://www.academia.edu/9681164/VALORIZACION_DEL_ESTI%3%89RCOL_DE_CERDO_A_TRAV%3%89S_DE_LA_PRODUCION_DE_BIOGAS. [Último acceso: 20 Julio 2015].
- [3] S. M. G. Koshkin N. I., Manual de Física Elemental, Mir. ISBN., 1995.
- [4] E. V. J. Cuesta, «Repositorio ESPE,» 15 Septiembre 2011. [En línea]. Available: <http://repositorio.espe.edu.ec/bitstream/21000/4664/1/T-ESPE-IASA%201-004573.pdf>. [Último acceso: 27 Julio 2015].
- [5] E. t. f. FASES., «GUÍA DE ELABORACIÓN DE ABONOS ORGÁNICOS FERMENTADOS,» 2007.
- [6] Hanna Instruments, «Hanna Instruments,» 01 Enero 2015. [En línea]. Available: <http://www.hannainst.es/blog/conductividad-y-solidos-disueltos/>. [Último acceso: 30 Julio 2015].
- [7] Carbotecnia, «Carbotecnia,» 16 Septiembre 2014. [En línea]. Available: <http://www.carbotecnia.info/encyclopedia/solidos-disueltos-totales-tds/>. [Último acceso: 30 Julio 2015].

- [8] Waterboards, «Waterboards,» [En línea]. Available: http://www.waterboards.ca.gov/water_issues/programs/swamp/docs/cwt/guidance/3130sp.pdf. [Último acceso: 30 Julio 2015].
- [9] TecMikro, «TecMikro,» 12 08 2015. [En línea]. Available: <http://tecmikro.com/ds18b20-wp-sensores-de-temperatura-arduino>.
- [10] American Marine Inc, «Products: American Marine Inc,» 12 08 2015. [En línea]. Available: <http://americanmarineusa.com/collections/accessories/products/pinpoint-ph-replacement-probe>.
- [11] Atlas Scientific LLC , «Atlas-Scientific.com,» [En línea]. Available: http://www.atlas-scientific.com/_files/_datasheets/_circuit/pH_EZO_datasheet.pdf? [Último acceso: 30 Julio 2015].
- [12] Atlas Scientific LLC, «Circuit: Atlas Scientific,» 12 08 2015. [En línea]. Available: http://www.atlas-scientific.com/product_pages/circuits/ezo_ph.html.
- [13] Atlas Scientific LLC, «Probes: Atlas Scientific,» [En línea]. Available: http://www.atlas-scientific.com/_files/_datasheets/_probe/ec-probe-datasheet.pdf. [Último acceso: 13 Agosto 2015].
- [14] Atlas Scientific LLC, «Atlas Scientific,» [En línea]. Available: http://www.atlas-scientific.com/_files/_datasheets/_circuit/EC_EZO_Datasheet.pdf? [Último acceso: 13 Agosto 2015].
- [15] M. Richardson, «Getting Started with BeagleBone,» *Make*, pp. 3-4, 2014.
- [16] T. Dicola, «Adafruit,» 04 05 2015. [En línea]. Available: <https://learn.adafruit.com/embedded-linux-board-comparison/overview>.

- [17] O. Electronics, «Mercado Libre,» [En línea]. Available: http://articulo.mercadolibre.com.ec/MEC-407464964-electrovalvula-solenoide-agua-control-electronica-12vdc-110v-_JM. [Último acceso: 15 Agosto 2015].
- [18] AliExpress, «Productos al por Mayor: AliExpress,» 14 Agosto 2015. [En línea]. Available: <http://es.aliexpress.com/w/wholesale-direct-acting-valve.html>.
- [19] eBay Inc, «Motors: eBay Inc,» 14 Agosto 2015. [En línea]. Available: <http://www.ebay.ca/itm/4-wire-CNC-42-Stepper-Motor-Model-17PM-K054-G7WS-Size-42-42mm-4mm-shaft-Dia-/190882395297>.
- [20] AliExpress, «Productos: AliExpress,» 14 Agosto 2015. [En línea]. Available: http://es.aliexpress.com/store/product/Japanese-Minebea-NEMA-17-stepper-motor-17PM-K054-G7WS-NEMA17-CE-ROSH-ISO-CNC-Laser-Grind/133424_1489658607.html.
- [21] B. Schmalz, «schmalzhaus,» 14 Agosto 2015. [En línea]. Available: <http://www.schmalzhaus.com/EasyDriver/>.
- [22] Calvos Electrónica Ltda., «Calvos Electrónica Ltda.,» 28 Agosto 2015. [En línea]. Available: <http://www.calvoselectronica.com/licuadoras/463-piloto-1752-codigo-220806-110v-220v.html>.
- [23] J. Cooper, «Adafruit,» 4 Mayo 2015. [En línea]. Available: <https://learn.adafruit.com/setting-up-io-python-library-on-beaglebone-black/uart>.
- [24] Ebay, «P4pm Nuevo 12v 4 Canales el módulo De Relé Con Optoacoplador Para Arduino Dsp Avr Pic Arm,» 03 Julio 2014. [En línea]. Available: http://www.ebay.com/itm/P4PM-New-12V-4-Channel-Relay-Module-With-Optocoupler-For-Arduino-DSP-AVR-PIC-ARM/301233789370?_trksid=p2047675.c100005.m1851&_trkparms=aid%3D222007%26algo%3DSIC.MBE%26ao%3D1%26asc%3D20131003132420%26

meid%3Dd169d9d07c964018a2b20c8e1. [Último acceso: 08 Septiembre 2015].

- [25] NYPlatform, «4-Channel 12V Relay Module With Optocoupler For Arduino DSP AVR PIC ARM,» [En línea]. Available: http://www.nyplatform.com/index.php?route=product/product&product_id=766. [Último acceso: 08 Septiembre 2015].

- [26] Gravitech, «LCDs: Gravitech,» 14 Agosto 2015. [En línea]. Available: <http://www.gravitech.us/20chbllcd.html>.

- [27] Atlas Scientific LLC, «Circuit: Atlas Scientific,» 13 Agosto 2015. [En línea]. Available: http://www.atlas-scientific.com/product_pages/circuits/ezo_ec.html.

ANEXOS

Asignación de Pines de la BeagleBone Black

Variable	Tipo	BeagleBone Black
RX_pH	Digital Output	P9_11
Dir_Motor	Digital Output	P9_12
TX_pH	Digital Output	P9_13
Temperatura	Digital Output	P9_22
TX_EC	Digital Output	P9_24
RX_EC	Digital Output	P9_26
Electrovalvula_Base	Digital Output	P9_27
Electrovalvula_Acida	Digital Output	P9_30
Step_Motor	Digital Output	P8_7
RS LCD	Digital Output	P8_8
MS1_Motor	Digital Output	P8_9
CLOCK/ENABLE LCD	Digital Output	P8_10
DB4 LCD	Digital Output	P8_12
DB5 LCD	Digital Output	P8_14
DB6 LCD	Digital Output	P8_16
Luz piloto	Digital Output	P8_17
DB7 LCD	Digital Output	P8_18
MS2_Motor	Digital Output	P8_26
Boton_off_base	Digital Input	P8_11
Boton_on_motor	Digital Input	P8_15
Automatico_manual	Digital Input	P9_14
Boton_on_acido	Digital Input	P9_15
Boton_on_base	Digital Input	P9_23
Boton_off_acido	Digital Input	P9_41

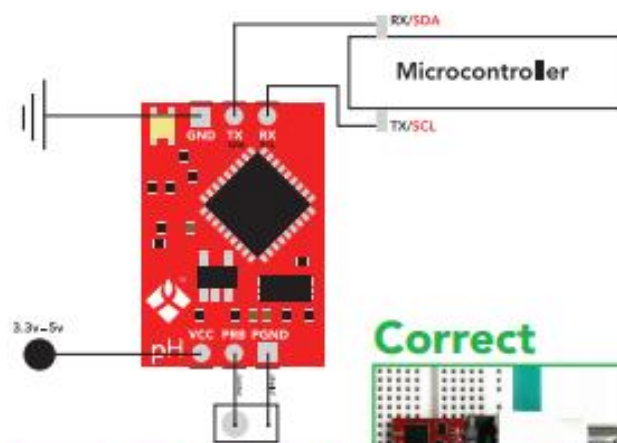
Diagrama de Conexión pH Circuit Atlas con BeagleBone Black

Fuente: [12]

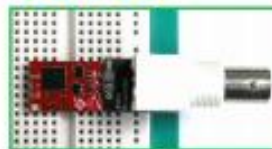


Wiring diagram

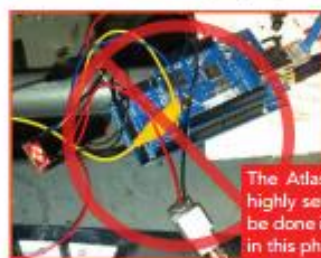
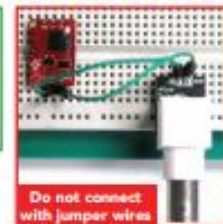
- To connect the Circuit to your microcontroller, follow the diagram below.
- Make sure your Circuit and microcontroller share a common ground.
- TX on your Circuit connects to RX on your microcontroller.
- If in PC mode connect SDA to SDA and SCL to SCL
- *4.7k pull up resistor on SDA and SCL may be required



Correct



Incorrect



The Atlas Scientific™ EZO™ class pH circuit is highly sensitive equipment. Debugging should be done in a bread board; Not like what is show in this photo.

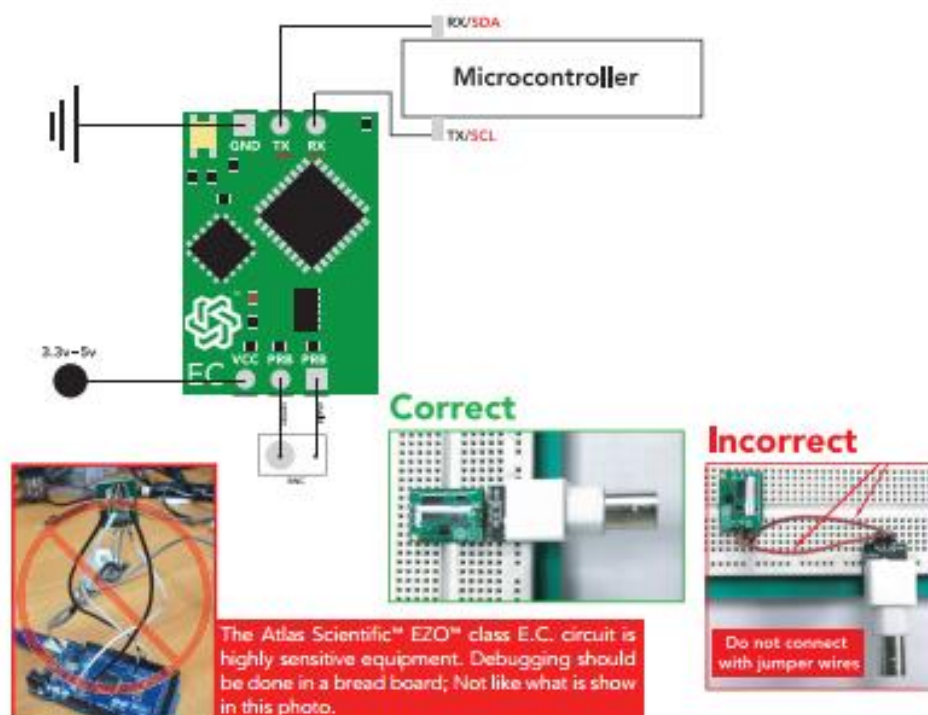
Diagrama de Conexión EC Circuit Atlas con BeagleBone Black

Fuente: [14]



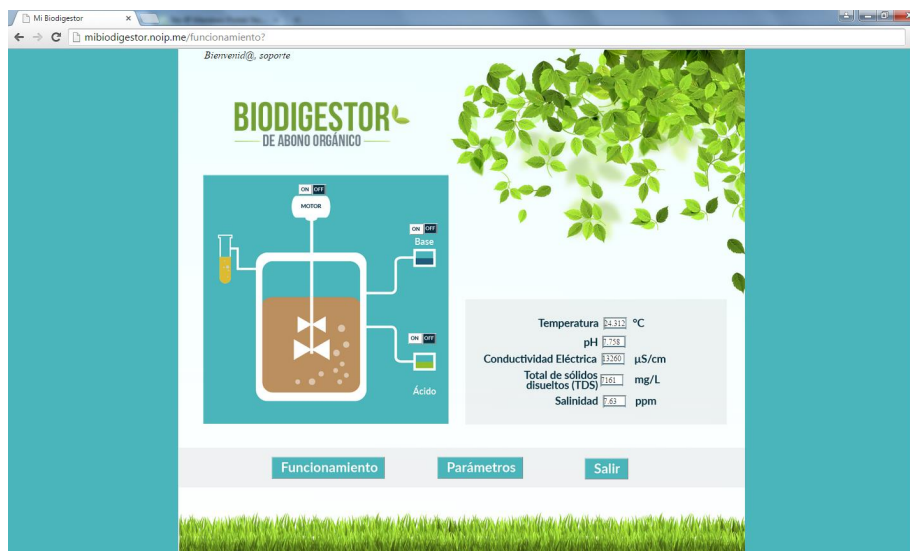
Wiring diagram

- To connect the Circuit to your microcontroller, follow the diagram below.
- Make sure your Circuit and microcontroller share a common ground.
- TX on your Circuit connects to RX on your microcontroller.
- If in I²C mode connect SDA to SDA and SCL to SCL
- *4.7k pull up resistor on SDA and SCL may be required



Página Web

Funcionamiento



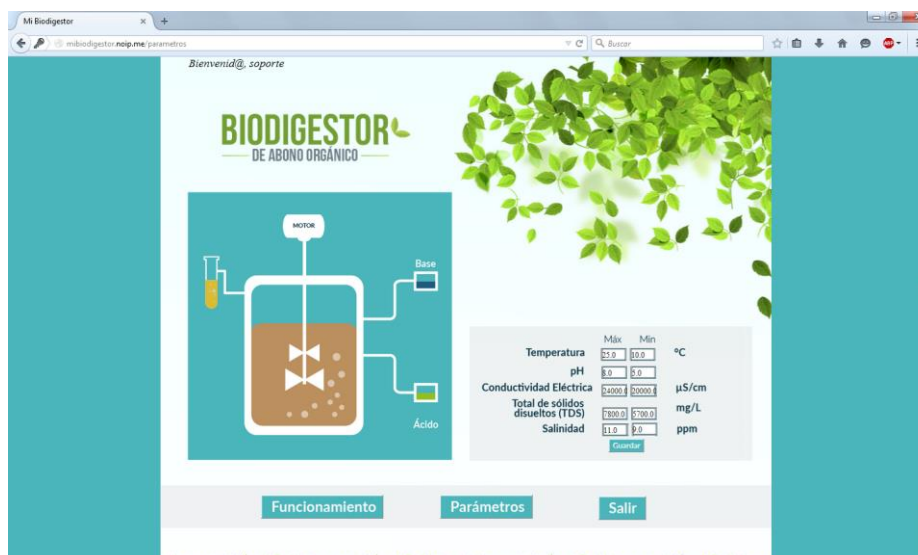
BIODIGESTOR
DE ABONO ORGÁNICO

Motor: ON OFF
Base: ON OFF
Ácido: ON OFF

Temperatura	23.312	°C
pH	8.351	
Conductividad Eléctrica	3320	µS/cm
Total de sólidos disueltos (TDS)	161	mg/L
Salinidad	2.3	ppm

Funcionamiento Parámetros Salir

Parámetros



BIODIGESTOR
DE ABONO ORGÁNICO

Motor: ON OFF
Base: ON OFF
Ácido: ON OFF

Temperatura	Máx: 57.0	Mín: 00.0	°C
pH	0.0	0.0	
Conductividad Eléctrica	00000	00000	µS/cm
Total de sólidos disueltos (TDS)	0000.0	07000.0	mg/L
Salinidad	0.0	0.0	ppm

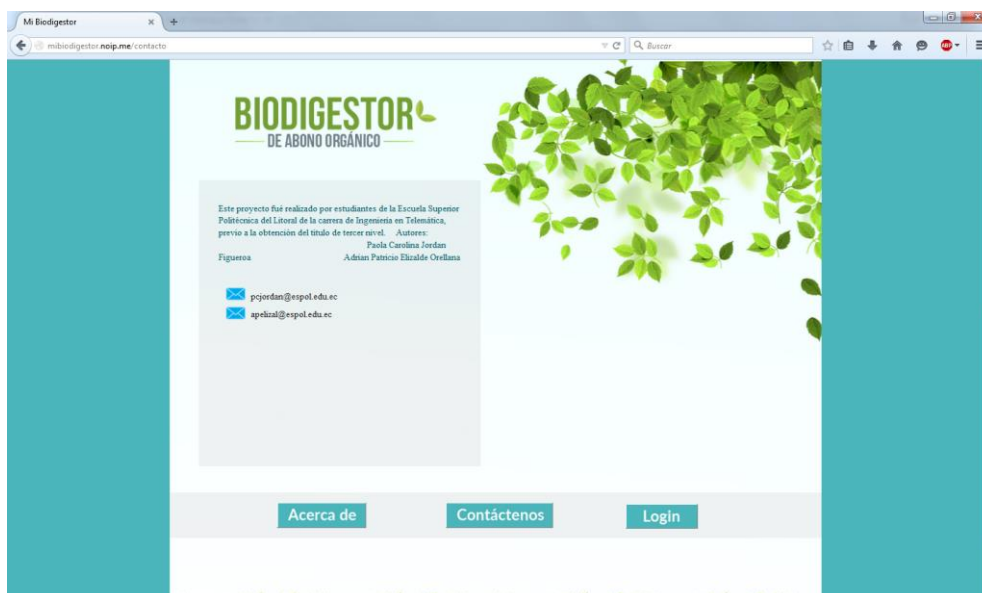
Guardar

Funcionamiento Parámetros Salir

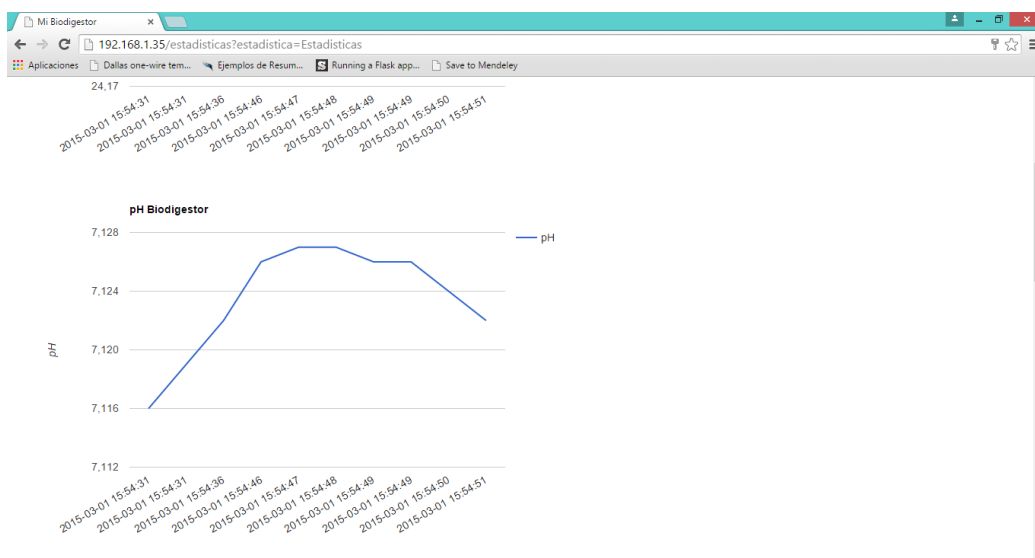
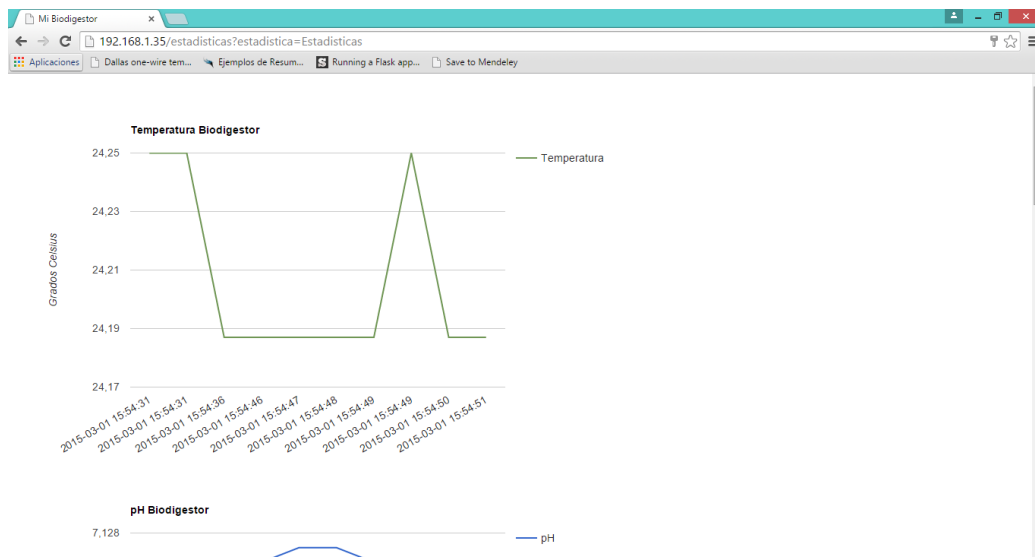
Acerca De

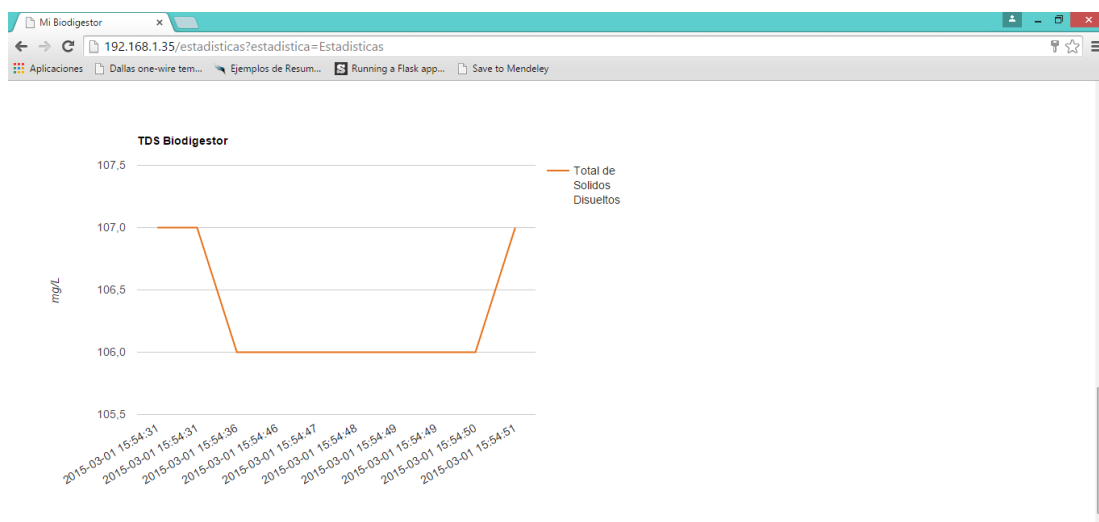
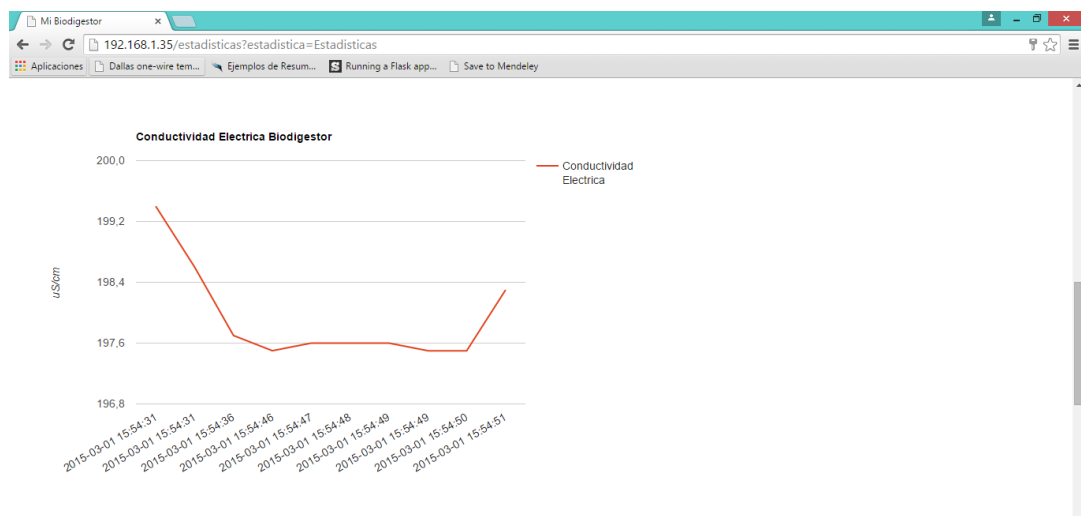


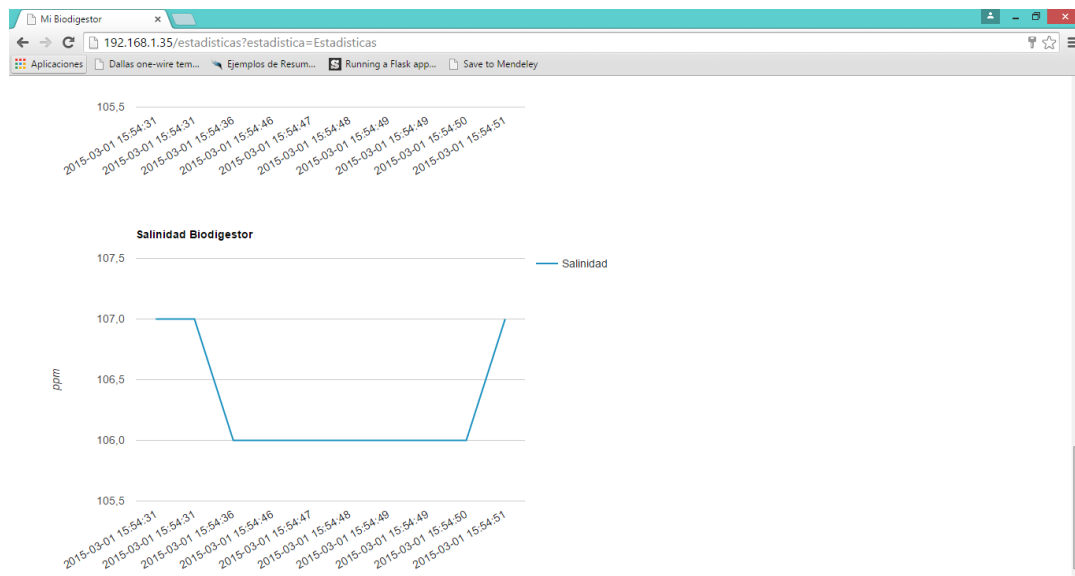
Contáctenos



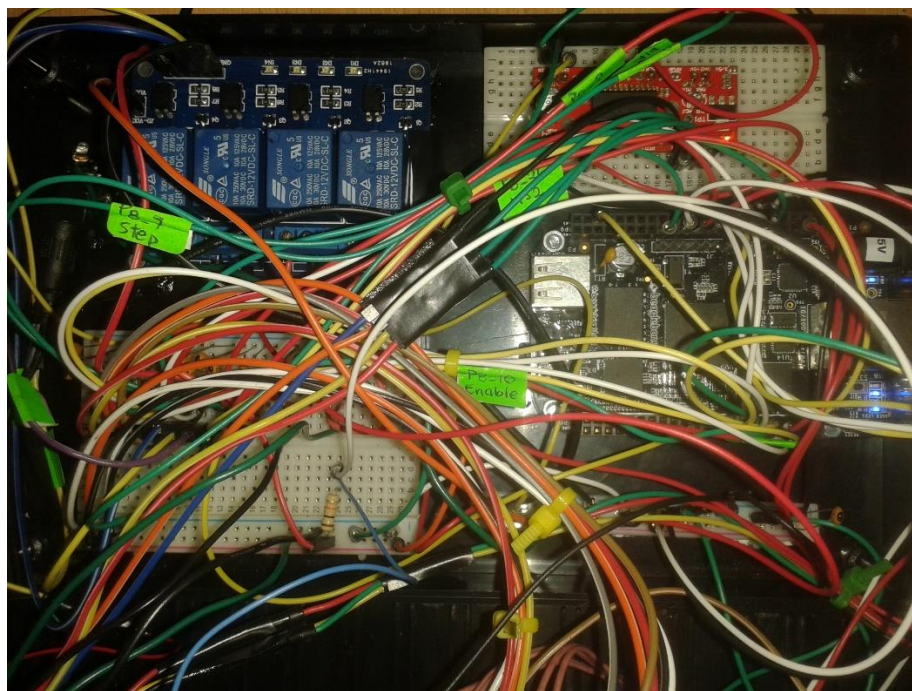
Estadísticas

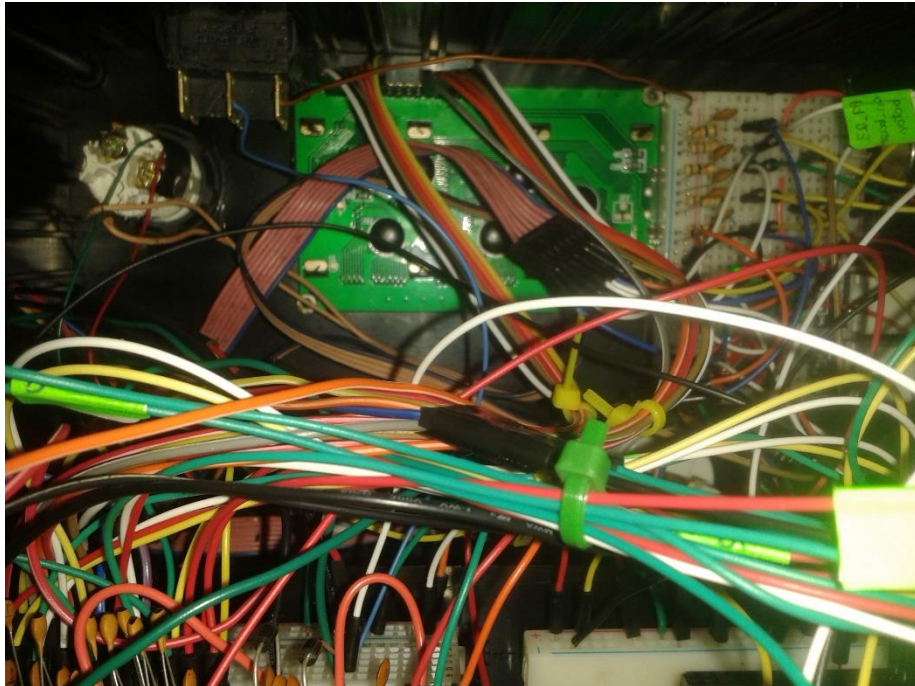






Cableado - Parte Interior – Tablero de Administración Local





Código Fuente

ARCHIVO login.py

```

import time
import os
from emailer import alertMe
from flask import Flask, flash, session, redirect, url_for, request, render_template, jsonify
from flaskext.mysql import MySQL
import Adafruit_BBIO.UART as UART
import Adafruit_BBIO.GPIO as GPIO
import serial
import StringIO
import csv
import time

mysql = MySQL()

app = Flask(__name__)

app.config['MYSQL_DATABASE_USER'] = 'root'
app.config['MYSQL_DATABASE_PASSWORD'] = 'root'
app.config['MYSQL_DATABASE_DB'] = 'Biodigestor'
app.config['MYSQL_DATABASE_HOST'] = 'localhost'

```

```

app.secret_key = os.urandom(24)

mysql.init_app(app)

conn = mysql.connect()
cursor = conn.cursor()
conn.autocommit(True)

GPIO.setup("P9_27", GPIO.OUT)
GPIO.setup("P9_30", GPIO.OUT)

GPIO.setup("P9_16", GPIO.IN)

GPIO.output("P9_27", GPIO.HIGH)
GPIO.output("P9_30", GPIO.HIGH)

GPIO.setup("P9_12", GPIO.OUT)
GPIO.setup("P8_7", GPIO.OUT)

GPIO.setup("P8_9", GPIO.OUT)
GPIO.setup("P8_26", GPIO.OUT)

def motorpaso():
    GPIO.output("P8_9", GPIO.LOW)
    GPIO.output("P8_26", GPIO.LOW)

    GPIO.output("P9_12", GPIO.HIGH)

    for i in range(0,4000):
        GPIO.output("P8_7", GPIO.HIGH)
        time.sleep(0.00074)
        GPIO.output("P8_7", GPIO.LOW)
        time.sleep(0.00074)

    time.sleep(0.5)
    GPIO.output("P9_12", GPIO.LOW)

    for i in range(0,4000):
        GPIO.output("P8_7", GPIO.HIGH)
        time.sleep(0.00074)
        GPIO.output("P8_7", GPIO.LOW)
        time.sleep(0.00074)

```

```

cont = 0
cont1 = 0
cont2 = 0
line1 = ""
line2 = ""
lecturaEC = ""
lecturaTDS = ""
lecturaSal = ""
lecturapH = ""
lectura_temp = ""
data1 = ""
data2 = ""

```

```

@app.route("/")
def index():
    if 'logged_in' in session:
        nombre = session['nombre']
        apellido = session['apellido']
        bienvenido = "Bienvenid@, " + nombre + " " + apellido

        if session['tipo'] == "administrador":
            return render_template("Funcionamiento.html",oculto="submit",
bienvenido=bienvenido)
        if session['tipo'] == "operario":
            return render_template("Funcionamiento.html",oculto="hidden",
bienvenido=bienvenido)
    else:
        return render_template("paginalogin.html")

```

```

@app.route("/", methods=['POST'])
def Authenticate():
    message = "Username y Password incorrectos. Intente de nuevo"
    username = request.form['username']
    password = request.form['password']
    cursor.execute("SELECT * from User where BINARY username='" + username + "'
and BINARY password='" + password + "'")
    data = cursor.fetchone()

    if data is None:
        return render_template("paginalogin.html", message=message)
    else:
        session['logged_in'] = True
        session['tipo'] = data[5]
        session['nombre'] = data[1]
        session['apellido'] = data[2]
        if data[5] == "administrador":
            nombre = session['nombre']

```

```

    apellido = session['apellido']
    bienvenido = "Bienvenid@, " + nombre + " " + apellido
    return render_template("Funcionamiento.html",
bienvenido=bienvenido)

```

oculto="submit",

```

    if data[5] == "operario":
        nombre = session['nombre']
        apellido = session['apellido']
        bienvenido = "Bienvenid@, " + nombre + " " + apellido
        return render_template("Funcionamiento.html",
bienvenido=bienvenido)

```

oculto="hidden",

```
@app.route("/hello")
```

```
def hello():
```

```

    global cont
    global cont1
    global cont2
    global line1
    global line2
    global lecturaEC
    global lecturaTDS
    global lecturaSal
    global lecturaph
    global data1
    global data2

```

```

    conn = mysql.connect()
    cursor = conn.cursor()
    conn.autocommit(True)

```

```

    cursor.execute("SELECT tempMax from parametros")
    data = cursor.fetchone()
    tempMax = data[0]

```

```

    cursor.execute("SELECT phMax from parametros")
    data = cursor.fetchone()
    phMax = data[0]

```

```

    cursor.execute("SELECT phMin from parametros")
    data = cursor.fetchone()
    phMin = data[0]

```

```
while True:
```

```

    cursor.execute("SELECT * from valores")
    data1 = cursor.fetchone()

```

```
lecturaEC = data1[2]
```



```

lecturaTDS = data1[3]
lecturaSal = data1[4]
lecturapH = data1[1]
lectura_temp = data1[5]

if cont1 == 0:
    if lecturapH > phMax:
        cont1 = 1
        alertMe("Alerta!", "Se ha detectado un nivel alto de pH, Ambiente
alcalino pH="+str(lecturapH)+" .Porfavor tomar las precauciones debidas.")
    if lecturapH < phMax:
        cont1 = 0

if cont2 == 0:
    if lecturapH < phMin:
        cont2 = 1
        alertMe("Alerta!", "Se ha detectado un nivel bajo de pH, Ambiente
acido pH="+str(lecturapH)+" .Porfavor tomar las precauciones debidas.")
    if lecturapH > phMin:
        cont2 = 0

return jsonify(result=lectura_temp, ec=lecturaEC, tds=lecturaTDS,
sal=lecturaSal, pH=lecturapH)

cursor.close()
conn.close()

@app.route("/estadisticas")
def estadisticas():
    datos = []
    datosph = []
    datosec = []
    datostds = []
    datossal = []
    cursor.execute("SELECT * from Mtemperatura")
    data = cursor.fetchall()
    for row in data:
        datos.append(row[1])
        datos.append(row[2])
    fecha1 = datos[0]
    fecha2 = datos[2]
    fecha3 = datos[4]
    fecha4 = datos[6]
    fecha5 = datos[8]
    fecha6 = datos[10]
    fecha7 = datos[12]
    fecha8 = datos[14]

```

```
fecha9 = datos[16]
fecha10 = datos[18]

temp1 = datos[1]
temp2 = datos[3]
temp3 = datos[5]
temp4 = datos[7]
temp5 = datos[9]
temp6 = datos[11]
temp7 = datos[13]
temp8 = datos[15]
temp9 = datos[17]
temp10 = datos[19]

        cursor.execute("SELECT * from Mph")
data = cursor.fetchall()
for row in data:
        datosph.append(row[2])
ph1 = datosph[0]
ph2 = datosph[1]
ph3 = datosph[2]
ph4 = datosph[3]
ph5 = datosph[4]
ph6 = datosph[5]
ph7 = datosph[6]
ph8 = datosph[7]
ph9 = datosph[8]
ph10 = datosph[9]

        cursor.execute("SELECT * from Mec")
data = cursor.fetchall()
for row in data:
        datosec.append(row[2])
ec1 = datosec[0]
ec2 = datosec[1]
ec3 = datosec[2]
ec4 = datosec[3]
ec5 = datosec[4]
ec6 = datosec[5]
ec7 = datosec[6]
ec8 = datosec[7]
ec9 = datosec[8]
ec10 = datosec[9]

        cursor.execute("SELECT * from Mtds")
data = cursor.fetchall()
for row in data:
        datostds.append(row[2])
```

```

tds1 = datostds[0]
tds2 = datostds[1]
tds3 = datostds[2]
tds4 = datostds[3]
tds5 = datostds[4]
tds6 = datostds[5]
tds7 = datostds[6]
tds8 = datostds[7]
tds9 = datostds[8]
tds10 = datostds[9]

```

```

cursor.execute("SELECT * from Msal")

```

```

data = cursor.fetchall()

```

```

for row in data:

```

```

    datossal.append(row[2])

```

```

sal1 = datossal[0]

```

```

sal2 = datossal[1]

```

```

sal3 = datossal[2]

```

```

sal4 = datossal[3]

```

```

sal5 = datossal[4]

```

```

sal6 = datossal[5]

```

```

sal7 = datossal[6]

```

```

sal8 = datossal[7]

```

```

sal9 = datossal[8]

```

```

sal10 = datossal[9]

```

```

return

```

```

render_template("chart.html", fecha1=fecha1, fecha2=fecha2, fecha3=fecha3, fecha4=f
echa4, fecha5=fecha5, fecha6=fecha6, fecha7=fecha7, fecha8=fecha8, fecha9=fecha9, f
echa10=fecha10,

```

```

temp1=temp1, temp2=temp2, temp3=temp3, temp4=temp4, temp5=temp5, temp6=tem
p6, temp7=temp7, temp8=temp8, temp9=temp9, temp10=temp10,

```

```

ph1=ph1, ph2=ph2, ph3=ph3, ph4=ph4, ph5=ph5, ph6=ph6, ph7=ph7, ph8=ph8, ph9=ph9
, ph10=ph10,

```

```

ec1=ec1, ec2=ec2, ec3=ec3, ec4=ec4, ec5=ec5, ec6=ec6, ec7=ec7, ec8=ec8, ec9=ec9, e
c10=ec10,

```

```

tds1=tds1, tds2=tds2, tds3=tds3, tds4=tds4, tds5=tds5, tds6=tds6, tds7=tds7, tds8=tds8, t
ds9=tds9, tds10=tds10,

```

```

sal1=sal1, sal2=sal2, sal3=sal3, sal4=sal4, sal5=sal5, sal6=sal6, sal7=sal7, sal8=sal8, sal
9=sal9, sal10=sal10)

```

```

@app.route("/paginainfo1")
def paginainfo1():
    return render_template("pagina-mas-info1.html")

@app.route("/acercaDe")
def acercaDe():
    return render_template("pagina-mas-info2.html")

@app.route("/contacto")
def contacto():
    return render_template("contacto.html")

@app.route("/funcionamiento")
def funcionamiento():
    nombre = session['nombre']
    apellido = session['apellido']
    bienvenido = "Bienvenid@, " + nombre + " " + apellido
    if session['tipo'] == "administrador":
        return render_template("Funcionamiento.html",oculto="submit",
bienvenido=bienvenido)
    if session['tipo'] == "operario":
        return render_template("Funcionamiento.html",oculto="hidden",
bienvenido=bienvenido)

@app.route("/parametros")
def parametros():
    nombre = session['nombre']
    apellido = session['apellido']
    bienvenido = "Bienvenid@, " + nombre + " " + apellido

    cursor.execute("SELECT tempMax from parametros")
    data = cursor.fetchone()
    tempMax = data[0]

    cursor.execute("SELECT phMax from parametros")
    data = cursor.fetchone()
    phMax = data[0]

    cursor.execute("SELECT ecMax from parametros")
    data = cursor.fetchone()
    ecMax = data[0]

    cursor.execute("SELECT tdsMax from parametros")
    data = cursor.fetchone()
    tdsMax = data[0]

    cursor.execute("SELECT salMax from parametros")
    data = cursor.fetchone()

```

```

salMax = data[0]

cursor.execute("SELECT tempMin from parametros")
data = cursor.fetchone()
tempMin = data[0]

cursor.execute("SELECT phMin from parametros")
data = cursor.fetchone()
phMin = data[0]

cursor.execute("SELECT ecMin from parametros")
data = cursor.fetchone()
ecMin = data[0]

cursor.execute("SELECT tdsMin from parametros")
data = cursor.fetchone()
tdsMin = data[0]

cursor.execute("SELECT salMin from parametros")
data = cursor.fetchone()
salMin = data[0]

return render_template("Parametros.html", temperatura=tempMax,
pH=phMax, CE=ecMax, TDS=tdsMax, Salinidad=salMax, temperatura2=tempMin,
pH2=phMin, CE2=ecMin, TDS2=tdsMin, Salinidad2=salMin, bienvenido=bienvenido)

@app.route("/parametros", methods=['POST'])
def guardarValores():
    nombre = session['nombre']
    apellido = session['apellido']
    bienvenido = "Bienvenid@, " + nombre + " " + apellido

    tempMax = request.form['temperatura']
    phMax = request.form['pH']
    ecMax = request.form['CE']
    tdsMax = request.form['TDS']
    salMax = request.form['Salinidad']

    tempMin = request.form['temperatura2']
    phMin = request.form['pH2']
    ecMin = request.form['CE2']
    tdsMin = request.form['TDS2']
    salMin = request.form['Salinidad2']

    if tempMax.replace(".", "", 1).isdigit():
        cursor.execute("UPDATE parametros SET tempMax=" + tempMax)
    else:

```

```
tempMax = "NaN"

if phMax.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET phMax=" + phMax)
else:
    phMax = "NaN"

if ecMax.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET ecMax=" + ecMax)
else:
    ecMax = "NaN"

if tdsMax.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET tdsMax=" + tdsMax)
else:
    tdsMax = "NaN"

if salMax.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET salMax=" + salMax)
else:
    salMax = "NaN"

if tempMin.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET tempMin=" + tempMin)
else:
    tempMin = "NaN"

if phMin.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET phMin=" + phMin)
else:
    phMin = "NaN"

if ecMin.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET ecMin=" + ecMin)
else:
    ecMin = "NaN"

if tdsMin.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET tdsMin=" + tdsMin)
else:
    tdsMin = "NaN"

if salMin.replace(".", "", 1).isdigit():
    cursor.execute("UPDATE parametros SET salMin=" + salMin)
else:
    salMin = "NaN"
```

```

return render_template("Parametros.html", temperatura=tempMax, pH=phMax,
CE=ecMax, TDS=tdsMax, Salinidad=salMax, temperatura2=tempMin, pH2=phMin,
CE2=ecMin, TDS2=tdsMin, Salinidad2=salMin, bienvenido=bienvenido)

```

```

@app.route("/on_base")
def on_base():
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        GPIO.output("P9_27", GPIO.LOW)
        if session['tipo'] == "administrador":
            return render_template("Funcionamiento.html",oculto="submit")
        if session['tipo'] == "operario":
            return render_template("Funcionamiento.html",oculto="hidden")

```

```

@app.route("/off_base")
def off_base():
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        GPIO.output("P9_27", GPIO.HIGH)
        if session['tipo'] == "administrador":
            return render_template("Funcionamiento.html",oculto="submit")
        if session['tipo'] == "operario":
            return render_template("Funcionamiento.html",oculto="hidden")

```

```

@app.route("/on_acido")
def on_acido():
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        GPIO.output("P9_30", GPIO.LOW)
        if session['tipo'] == "administrador":
            return render_template("Funcionamiento.html",oculto="submit")
        if session['tipo'] == "operario":
            return render_template("Funcionamiento.html",oculto="hidden")

```

```

@app.route("/off_acido")

```

```

def off_acido():
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        GPIO.output("P9_30", GPIO.HIGH)
    if session['tipo'] == "administrador":
        return render_template("Funcionamiento.html",oculto="submit")
    if session['tipo'] == "operario":
        return render_template("Funcionamiento.html",oculto="hidden")

@app.route("/on_motor")
def on_motor():
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        motorpaso()
    if session['tipo'] == "administrador":
        return render_template("Funcionamiento.html",oculto="submit")
    if session['tipo'] == "operario":
        return render_template("Funcionamiento.html",oculto="hidden")

@app.route("/logout")
def logout():
    session.pop('logged_in', None)
    return redirect(url_for('index'))

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True, threaded=True)

```

ARCHIVO biodigestor.py

```

import math
import time
import Adafruit_CharLCD as LCD
import Adafruit_BBIO.UART as UART
import Adafruit_BBIO.GPIO as GPIO
import serial

```



```

import StringIO
import csv
import MySQLdb
from emailer import alertMe

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'root'
DB_NAME = 'Biodigestor'

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()      # Crear un cursor
conn.autocommit(True)

lcd_rs = 'P8_8'
lcd_en = 'P8_10'
lcd_d4 = 'P8_12'
lcd_d5 = 'P8_14'
lcd_d6 = 'P8_16'
lcd_d7 = 'P8_18'

lcd_columns = 20
lcd_rows = 4
lcd = LCD.Adafruit_CharLCD(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7,
lcd_columns, lcd_rows)

GPIO.setup("P8_17", GPIO.OUT)

GPIO.output("P8_17", GPIO.HIGH)

cont = 0
ec = 0
tds = 0
s = 0
ph = 0
cont1 = 0
cont2 = 0

while True:
    cursor.execute("SELECT * from valores")
    data = cursor.fetchone()

    lectura_temp = float(data[5])
    temperatura = round(lectura_temp, 1)

    cursor.execute("SELECT tempMax from parametros")

```

```

data1 = cursor.fetchone()
tempMax = data1[0]

    cursor.execute("SELECT phMax from parametros")
data1 = cursor.fetchone()
phMax = data1[0]

    cursor.execute("SELECT phMin from parametros")
data1 = cursor.fetchone()
phMin = data1[0]

    if cont1 == 0:
        if data[1] > phMax:
            cont1 = 1
            alertMe("Alerta!", "Se ha detectado un nivel alto de pH, Ambiente
alcalino pH="+str(data[1])+".Porfavor tomar las precauciones debidas.")
        if data[1] < phMax:
            cont1 = 0

    if cont2 == 0:
        if data[1] < phMin:
            cont2 = 1
            alertMe("Alerta!", "Se ha detectado un nivel bajo de pH,
Ambiente acido pH="+str(data[1])+".Porfavor tomar las precauciones debidas.")
        if data[1] > phMin:
            cont2 = 0

    if data[1] > phMax or data[1] < phMin:
        GPIO.output("P8_17", GPIO.LOW)
    else:
        GPIO.output("P8_17", GPIO.HIGH)

    ph = round(data[1], 0)
    sal = round(data[4], 0)
    lcd.message("Temp:" + str(temperatura) + "C pH: " + str(ph) + "\nEC: " +
str(data[2]) + "[uS/cm]\nTDS: " + str(data[3]) + "[mg/L]\n" + "SAL: " + str(sal) +
"[ppm]    ")

```

ARCHIVO boton.py

```

import Adafruit_BBIO.GPIO as GPIO
import time
import MySQLdb

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'root'
DB_NAME = 'Biodigestor'

```

```

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()        # Crear un cursor
conn.autocommit(True)

GPIO.setup("P9_23", GPIO.IN)
GPIO.setup("P8_11", GPIO.IN)
GPIO.setup("P9_15", GPIO.IN)
GPIO.setup("P9_41", GPIO.IN)
GPIO.setup("P8_15", GPIO.IN)

GPIO.setup("P9_16", GPIO.IN)

GPIO.setup("P9_27", GPIO.OUT)
GPIO.setup("P9_30", GPIO.OUT)

GPIO.setup("P9_12", GPIO.OUT)
GPIO.setup("P8_7", GPIO.OUT)

GPIO.setup("P8_9", GPIO.OUT)
GPIO.setup("P8_26", GPIO.OUT)

GPIO.output("P9_27", GPIO.HIGH)
GPIO.output("P9_30", GPIO.HIGH)

switch = 0
contmotor = 0

def motorpaso():

    GPIO.output("P8_9", GPIO.LOW)
    GPIO.output("P8_26", GPIO.LOW)

    GPIO.output("P9_12", GPIO.HIGH)

    for i in range(0,4000):
        GPIO.output("P8_7", GPIO.HIGH)
        time.sleep(0.00074)
        GPIO.output("P8_7", GPIO.LOW)
        time.sleep(0.00074)

    time.sleep(0.5)
    GPIO.output("P9_12", GPIO.LOW)

```

```

for i in range(0,4000):
    GPIO.output("P8_7", GPIO.HIGH)
    time.sleep(0.00074)
    GPIO.output("P8_7", GPIO.LOW)
    time.sleep(0.00074)

```

```

switch = 0
contmotor = 0

```

```

while True:
    cursor.execute("SELECT * from valores")
    data = cursor.fetchone()
    if GPIO.input("P9_16"):
        switch = 1
    else:
        switch = 0

    if switch == 1:
        if GPIO.input("P9_23"):
            GPIO.output("P9_27",GPIO.LOW)
            print "on base"
            time.sleep(1)

        if GPIO.input("P8_11"):
            GPIO.output("P9_27",GPIO.HIGH)
            print "off base"
            time.sleep(1)

        if GPIO.input("P9_15"):
            GPIO.output("P9_30",GPIO.LOW)
            print "on acido"
            time.sleep(1)

        if GPIO.input("P9_41"):
            GPIO.output("P9_30",GPIO.HIGH)
            print "off acido"
            time.sleep(1)

        if GPIO.input("P8_15"):
            motorpaso()
            print "on motor"
            time.sleep(1)

    else:
        cursor.execute("SELECT phMax from parametros")
        data1 = cursor.fetchone()

```

```

phMax = data1[0]

cursor.execute("SELECT phMin from parametros")
data1 = cursor.fetchone()
phMin = data1[0]

    contmotor = contmotor + 1
    if contmotor == 70000:
        motorpaso()
        contmotor = 0

    if data[1] > phMax:
        GPIO.output("P9_30", GPIO.LOW)
    if data[1] < phMin:
        GPIO.output("P9_27", GPIO.LOW)
    if data[1] < phMax and data[1] > phMin:
        GPIO.output("P9_27", GPIO.HIGH)
        GPIO.output("P9_30", GPIO.HIGH)

```

ARCHIVO EC.py

```

import Adafruit_BBIO.UART as UART
import serial
import StringIO
import csv
import MySQLdb

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'root'
DB_NAME = 'Biodigestor'

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()      # Crear un cursor

UART.setup("UART1")

ser = serial.Serial(port = "/dev/ttyO1", baudrate=9600)
ser.write("L,1\r")
ser.write("C,1\r")
line = ""
data = ""
cont = 0

```

```

j = 0
while True:
    data = ""
    try:
        while(data != "\r"):
            data = ser.read()
            if (data != "\r"):
                line = line + data
    except serial.serialutil.SerialException:
        pass
    if(data == "\r"):
        if(line[0:1] != "**"):
            reader = csv.reader(line.split('\n'), delimiter=',')
            for row in reader:
                ec = row[0]
                tds = row[1]
                s = row[2]

                cursor.execute("UPDATE valores SET EC=" + ec)
                cursor.execute("UPDATE valores SET TDS=" + tds)
                cursor.execute("UPDATE valores SET SAL=" + s)

            line = ""

        if cont < 10:
            cursor.execute("insert into ec values(NOW(), "+str(ec)+)")
            cursor.execute("insert into tds values(NOW(), "+str(tds)+)")
            cursor.execute("insert into sal values(NOW(), "+str(s)+)")
            cont = cont + 1
    else:
        cursor.execute("SELECT * from ec")
        data = cursor.fetchall()

        cursor.execute("truncate ec")
        cont = 0
        for row in data:
            cursor.execute("UPDATE Mec SET tiempo="+str(row[0])+",
            EC="+str(row[1])+ " where idec="+str(j+1))
            j = j + 1
        j = 0
        cursor.execute("SELECT * from tds")
        data = cursor.fetchall()

        cursor.execute("truncate tds")
        for row in data:
            cursor.execute("UPDATE Mtds SET tiempo="+str(row[0])+",
            TDS="+str(row[1])+ " where idtds="+str(j+1))

```

```

        j = j + 1
    j = 0
        cursor.execute("SELECT * from sal")
    data = cursor.fetchall()

    cursor.execute("truncate sal")
    for row in data:
        cursor.execute("UPDATE Msal SET tiempo="+str(row[0])+",
SAL="+str(row[1])+" where idsal="+str(j+1))
        j = j + 1
    j = 0

    conn.commit()

cursor.close()          # Cerrar el cursor
conn.close()           # Cerrar la conexn

```

ARCHIVO pH.py

```

import Adafruit_BBIO.UART as UART
import serial
import StringIO
import MySQLdb

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'root'
DB_NAME = 'Biodigestor'

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()        # Crear un cursor
conn.autocommit(True)

UART.setup("UART4")

ser1 = serial.Serial(port = "/dev/ttyO4", baudrate=9600)
ser1.write("L,1\r")
ser1.write("C,1\r")
line = ""
data = ""
cont = 0
j = 0
while True:

```

```

data = ""
while(data != "\r"):
    data = ser1.read()
    if (data != "\r"):
        line = line + data
if(data == "\r"):
    if(line[0:1] != "**"):
        ph = line
        cursor.execute("UPDATE valores SET pH=" + ph)
    line = ""

if cont < 10:
    cursor.execute("insert into ph values(NOW(), "+str(ph)+)")
    cont = cont + 1
else:
    cursor.execute("SELECT * from ph")
    data = cursor.fetchall()

    cursor.execute("truncate ph")
    cont = 0
    for row in data:
        cursor.execute("UPDATE Mph SET tiempo="+str(row[0])+",
pH="+str(row[1])+ " where idph="+str(j+1))
        j = j + 1
    j = 0

```

ARCHIVO temp.py

```

import time
import MySQLdb

w1="/sys/bus/w1/devices/28-000006886a4c/w1_slave"

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'root'
DB_NAME = 'Biodigestor'

datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

conn = MySQLdb.connect(*datos) # Conectar a la base de datos
cursor = conn.cursor()      # Crear un cursor

cont = 0
j = 0
while True:
    raw = open(w1, "r").read()
    lectura_temp = float(raw.split("t=")[-1])/1000

```



```

    cursor.execute("UPDATE valores SET Temp=" + str(lectura_temp))
    if cont < 10:
        cursor.execute("insert into temperatura
values(NOW(), "+str(lectura_temp)+")")
        cont = cont + 1
    else:
        cursor.execute("SELECT * from temperatura")
        data = cursor.fetchall()

        cursor.execute("truncate temperatura")
        cont = 0
        for row in data:
            cursor.execute("UPDATE Mtemperatura SET tiempo="+str(row[0])+",
temp="+str(row[1])+ " where idTemp="+str(j+1))
                j = j + 1
        j = 0
        conn.commit()

cursor.close()          # Cerrar el cursor
conn.close()           # Cerrar la conexión

```