FUELDIS

Prototipo de dispenser de combustible

Alumnos:

Bermegui, Verónica - Palumbo, Nicolás - Panchenko, Ivanna - Rodriguez Matiz, Federico
Estudiantes de Ingeniería en Sistemas de Información

Grupo Docente:

Ing. Rapallini, José - Ing. Mazzeo, Hugo
Aplicaciones en Tiempo Real [ATR]
Ingeniería en Sistemas de Información
Regional La Plata, Universidad Tecnológica Nacional
60 y 126, BERISSO, Prov. De Bs. As. Argentina

***Abstract*— Desarrollo de un prototipo de dispenser de combustible para la materia Aplicaciones de Tiempo Real. Se desriben las partes de un sistema real. Se describen los componentes utilizados para el desarrollo del prototipo y sus conexiones, se analizan los distintos estados del sistema y se expone la rutina de actualización de datos provenientes de un sensor desarrollada en arduino.**

Keywords—FUELDIS; dispenser de combustible; Aplicaciones de Tiempo Real; ATR; arduino

I. INTRODUCCIÓN

El objetivo del trabajo es construir un prototipo de surtidor o dispenser de combustible. La motivación principal es explorar el funcionamiento de los sistemas de tiempo real en aplicaciones industriales.

II. PROPUESTA DE TRABAJO

Nos proponemos construir un prototipo que bombee líquido, desde un tanque contenedor, que hace las veces del tanque de la estación de combustible, a un receptáculo más pequeño, que hace las veces del tanque de combustible de un vehículo, controlando el encendido/apagado de la bomba, conteo de la cantidad [en litros] bombeada y el cálculo del precio de la carga.

III. PARTES COMPONENTES Y DESCRIPCIÓN DE UN SISTEMA REAL SIMPLIFICADO

Se describen las distintas partes de un sistema real. En la *Fig. 1.* Se puede ver un diagrama en bloques con todas las partes mencionadas.

A. Alimentación:

Utiliza alimentación trifásica, de 380V para el motor externo que mueve la bomba. 220V para el dispenser electrónico, y 220V para los switches del motor y válvulas.

B. Motor externo:

Está relacionado con la bomba por medio de una polea.

C. Tanque de la estación:

Es un receptáculo que contiene varios miles de litros de combustible ubicado debajo del piso de la estación.

D. Tanque del vehículo:

Es el tanque hacia donde se bombea el combustible.

E. Bomba:

La bomba succiona el combustible desde el tanque de la estación hacia el tanque del vehículo.

F. Boquilla:

Permite la salida del combustible hacia el tanque, tiene una palanca de control que permite regular el flujo del mismo.

G. Switch Magnético:

Cuando la boquilla está colgada en el dispenser el switch magnético está cerrado y el motor externo se encuentra apagado. Al sacar la boquilla del dispenser el switch magnético se abre y el motor externo empieza a funcionar. Al colocar la boquilla en el dispenser, el switch magnético se cierra y el motor vuelve a su estado de apagado.

H. Medidor / Sensor de flujo:

Es generalmente un medidor de desplazamiento positivo de pistón oscilante, que permite generar corriente eléctrica conforme a la velocidad de movimiento de los pistones provocada por el paso de combustible.

I. Placa Madre:

El componente principal es el microcontrolador que cuenta el volumen de líquido transferido al tanque del vehículo. Recibe como entrada los pulsos del sensor de flujo y el estado del switch magnético ubicado en el dispenser. Al sacar la boquilla del dispenser, el switch magnético cambia de estado, el microcontrolador lo detecta y arranca un motor externo que hace funcionar la bomba a través de una placa de potencia. El combustible fluye entonces en un circuito que tiene retorno hacia el tanque de la estación, al mismo tiempo se abre una válvula solenoide que está conectada al medidor de flujo, de manera de que sólo este abierto el paso de combustible a la manguera cuando hay presión en el sistema. El combustible es entonces bombeado desde el tanque de la estación hacia la manguera dispensadora. Una palanca en la boquilla controla la cantidad de flujo dispensada por la boquilla.

J. Válvula Solenoide:

Abre el paso de líquido hacia la manguera y el tanque de combustible del vehículo cuando se retira la boquilla del surtidor.

K. Placa de Potencia:

Contiene los switches del motor externo y válvulas que controlan la succión y salida del combustible.

L. Teclado:

Permite especificar el volumen de líquido o la cantidad de dinero a cargar. Cuando la carga es automática, el flujo de combustible corta automáticamente cuando se llega a la cantidad o monto deseado de combustible. También sirve para ejecutar operaciones de diagnóstico y consultas al sistema.

M. Pantalla:

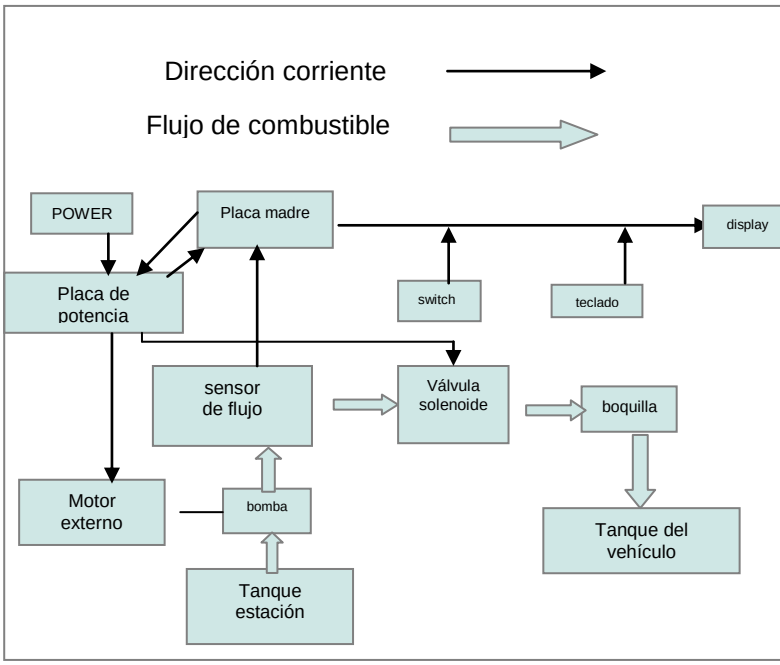
Permite ver la cantidad de combustible dispensado, costo y precio por unidad de volumen. También indica códigos de error, resultados de consultas y operaciones de diagnóstico.

IV. DESARROLLO DEL PROTOTIPO

El prototipo permite 3 modos de carga:

- Modo manual.
- Modo automático de volumen. Carga determinada cantidad de volumen [mililitros] seleccionada previamente a través de una interfaz de botones.
- Modo automático de importe. Carga un determinado importe [pesos] seleccionado previamente a través de una interfaz.

Fig. 1. Diagrama en bloques del sistema simplificado.



A. Particionamiento Básico:

Se utiliza un microcontrolador Atmel Atmega1280 montado en una placa Arduino para realizar todo el procesamiento en tiempo real y tareas de control.

Se utiliza un flujómetro de efecto Hall para medir el flujo/caudal dispensado.

Se utiliza una electroválvula de agua para controlar el curso del flujo.

Se utiliza un Shield Arduino con LCD de 2 líneas de 16 caracteres y teclado incorporado. La pantalla se usa para presentar información sobre el volumen de combustible dispensado y su costo. El teclado se utiliza para establecer la cantidad de combustible o dinero para la carga desatendida.

Se utiliza un arreglo de relés para comandar la electroválvula y encender la bomba.

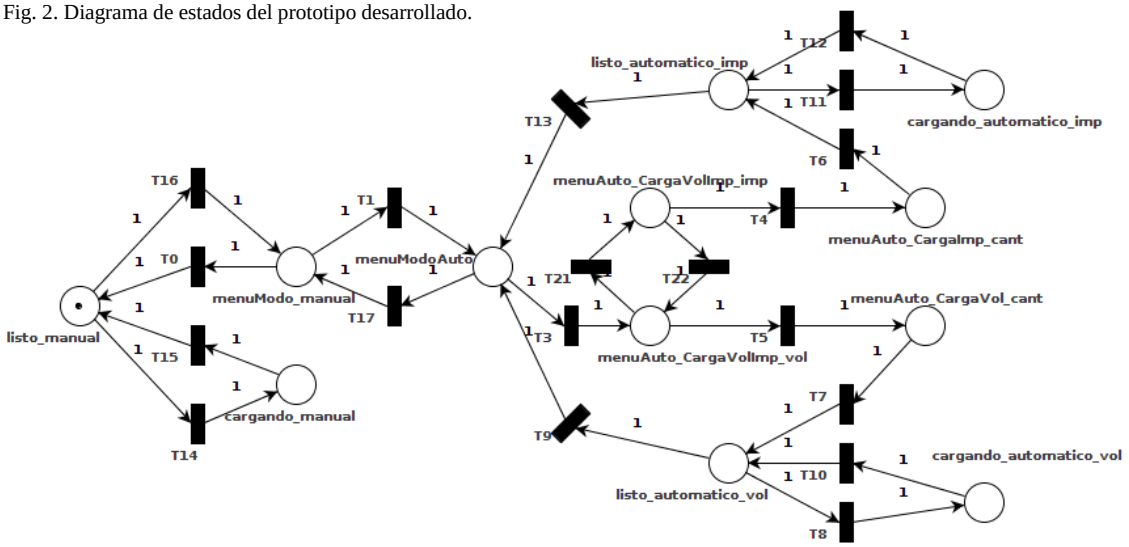
B. Estados del sistema:

En la *Fig.* 2 se puede observar un diagrama de estados del prototipo desarrollado.

El sistema comienza en el estado listo_manual. A partir de ahí, el usuario puede pasar al estado cargando_manual al descolgar la boquilla. [T14], se enciende la bomba, y a partir de ese momento el sensor empieza a contar el flujo que pasa cuando se presiona la boquilla, y actualiza volumen e importe en pantalla.

Cuando se termina de cargar, se cuelga la boquilla [T15] y el sistema vuelve al estado listo_manual, luego de cierto tiempo se desactiva la bomba. Desde aquí es posible ingresar al menú de selección de modo (menuModo_manual) al presionar el botón select [T16], luego con los botones arriba y abajo es posible cambiar la opción de manual a automático y viceversa [T1 y T17].

Fig. 2. Diagrama de estados del prototipo desarrollado.



Para confirmar el modo se debe presionar el botón OK. Si se selecciona manual, se vuelve al modo listo_manual [T0].

Si se selecciona modo automático [T3], vamos a una pantalla de selección que nos permite elegir si queremos que corte por volumen o importe. Y podemos cambiar entre esas opciones con los botones arriba y abajo[T21 y T22].

Si elegimos que corte por volumen [T5], vamos a una pantalla que nos permite elegir con los botones arriba y abajo, la cantidad de volumen que se quiere cargar, una vez elegida la cantidad con el botón OK, pasamos al estado listo_automático.

Desde ahí tenemos la opción de volver al menú de selección de modo presionando el botón select [T9], o podemos realizar la carga para la cantidad que habíamos elegido. En ese caso, levantamos la boquilla y pasamos al estado cargando_automatico_vol [T8]. En este estado, se enciende la bomba, y el sensor de flujo cuenta la cantidad de líquido que pasa y se actualiza en pantalla cantidad con respecto al total de volumen seleccionado e importe, cuando se llega al final el sistema corta solo. En cualquier momento se puede dejar de cargar y volver al estado listo_automatico_vol [T10]. Siempre que se vuelve a este estado, se resetean los contadores de volumen e importe, luego de cierto tiempo se apaga la bomba.

El modo de carga automática seleccionando importe, es análogo al de volumen. El corte cuando se llega a la cantidad máxima de volumen o importe se realiza a través de la electroválvula.

C. Diseño de cada bloque del sistema:

1) Microcontrolador y Placa Madre.

Se utilizó un Arduino MEGA (micro Atmel Atmega 1280), del cual se aprovecha el puerto serie, las entradas y salidas digitales.

Se conecta una entrada de interrupción a los pulsos del sensor [Pin 21], y otra de las entradas de interrupción para detectar que se levantó la boquilla [Pin 18].

Se utilizan las salidas para: activar y desactivar la electroválvula, [Pin 16].

Actualizar el LCD que indica el volumen dispensado y el costo. [Shield LCD].

2) Flujómetro de efecto Hall.

Consiste de una válvula de cuero plástico, un rotor de agua, y un sensor de efecto hall. Cuando el agua fluye a través del rotor, el mismo gira generando un tren de pulsos proporcional al flujo que pasa a través del sensor.

El sensor posee 3 cables, uno de ellos la salida de pulsos, se conecta, como se mencionó anteriormente a una de las entradas de interrupción del Arduino [Pin 21]. Los cables remanentes son GND y voltaje de referencia, este último se debe conectar a 5V [1].

Las especificaciones del flujómetro utilizado se pueden ver en la TABLA I.

El flujómetro genera una salida de pulsos, cuya frecuencia en Hz, es igual a 7.5Q, donde Q es el caudal en L / min, y tiene un error del orden de +/- 3%. Las especificaciones de la señal de salida se pueden ver en la TABLA II.

TABLA I. ESPECIFICAICONES DEL FLUJÓMETRO

Tensión de funcionamiento	5V-24V
Corriente Máxima	15 mA (DC 5V)
Peso	43 g.
Diámetro Externo	20 mm.
Rango de Flujo medible	1 – 30 L / min.
Rango de temperatura	0°C ~ 80°C
Temperatura del Líquido	Debe ser menor a 120 °C
Rango de humedad	35% a 90% RH
Presión soportada	Menor a 1.2 Mpa
Temperatura de almacenaje	-25°C ~ 80°C

TABLA II. ESPECIFICAICONES DE LA SEÑAL DE SALIDA DEL FLUJÓMETRO

Pulso de Salida (Nivel alto)	Tensión de la señal, mayor a 4.5V (entrada DC 5V)
Pulso de Salida (Nivel bajo)	Tensión de la señal, menor a 0.5V (entrada DC 5V)
Precisión	3% (con caudal desde 1 L / min a 10L / min)
Ciclo activo de la señal de salida	40%~60%

3) Electroválvula para agua.

Utilizamos agua en lugar de combustible, ya que al tratarse de un prototipo de aprendizaje, para utilizar técnicas de tiempo real no consideramos valioso el uso combustibles inflamables. Las especificaciones de la electroválvula usada se pueden ver en la TABLA III.

TABLA III. ESPECIFICAICONES DE LA ELECTROVÁLVULA AQT15SP

Tamaño del tubo	¾''
Material	Plástico
Temperatura de trabajo	1°C-75°C
Flujo de trabajo	0.02 Mpa – 3 L/min, 0.1 Mpa – 12 L/min, 0.8 Mpa – 35L/min
Tensión	DC12V
Rango de Tensión	15%
Resistencia	4.75KΩ ±0.25KΩ (20ºC)
Entorno de trabajo	Agua
Vida Útil	Más de 1,000,000 de aperturas

4) Placa de Potencia.

Como placa de potencia, usamos un arreglo de relés de 4 canales, para permitir el comando de los relés mediante las salidas digitales del Arduino.

Cada relé consume 20 mA a 5V, y dispone de un contacto simple inversor de 10ª a 250 VAC ó 30 VDC. Cada canal dispone de un LED indicador de estado, la placa además dispone de un LED indicador de alimentación general.

Las entradas de control son compatibles con 5V TTL. El conexionado de potencia se realiza mediante bornes de tornillo. Utilizamos un relé para cortar la carga, por ejemplo cuando se llegó a la cantidad deseada. El [Pin 16] del Arduino se conecta a la entrada, del relé y en la salida, se conecta la electroválvula, junto con una batería de gel de 12V que permite activarla,

5) Shield LCD con teclado incorporado.

Usamos un shield compatible Arduino, que dispone de un display LCD alfanumérico de 2 hileras de 16 caracteres, junto con 6 pulsadores.

En nuestra aplicación los pulsadores son usados como (select, izquierda, arriba, abajo, derecha y ok).

6) Bomba.

Para el prototipo usamos una bomba sumergible de pecera ATMAN AT-104. Las especificaciones se pueden encontrar en la TABLA IV.

Por cuestiones de simplicidad omitimos el encendido y apagado de la bomba a través de relés y decidimos hacerlo en forma manual a través de un interruptor de seguridad.

TABLA IV. ESPECIFICACIONES DE LA BOMBA SUMERGIBLE

Caudal	1700 litros / hora
Elevación	2 Mts.
Potencia	38 watts a 220 v.
Dimensiones	Largo: 11 cm – Alto: 8,5 cm – Ancho: 7 cm
Diámetro de Salida	19 mm externo, 14 mm interno.

7) Boquilla.

Utilizamos una canilla de plástico como reemplazo de la boquilla. Por simplicidad omitimos el switch magnético que detecta el descuelgue del surtidor, y lo reemplazamos por un switch electrónico estándar, conectado al [Pin 18] del Arduino.

V. SOFTWARE

El software fue desarrollado en el lenguaje especial de Arduino, muy parecido al lenguaje C. Se usó la biblioteca LiquidCrystal [2], para el manejo del LCD y la biblioteca FiniteStateMachine [3], para implementar los distintos estados del sistema. En la Fig. 3, se expone la rutina de actualización de datos del sensor, basada en la implementación del proyecto Practical Arduino [4], más propiamente en el código fuente abierto disponible en Github [5].

Fig. 3. Rutina de actualización de datos del sensor.

```
/* El sensor de flujo tira aproximadamente 4.5
pulsos por segundo por litro/minuto de flujo.*/
float factorCalibracion = 4.5;

volatile int pulsos; // cuenta pulsos del sensor
float flujo // Flujo en L/Min
float volMiliLitros; // Volumen en mililitros
unsigned long tAnterior;

pulsos = 0;
flujo = 0.0;
volMiliLitros = 0;
tAnterior = 0;

// VOLUMEN E IMPORTE ACTUAL
float volActual = 0;
float impActual = 0;
byte nroInterrupcion = 2; // pin 21

void deshabilitaSensor() {
  detachInterrupt(nroInterrupcion);
}

void actualizarDatosSensor() {
// Procesa contadores una vez por segundo
  if((millis() - tAnterior) > 1000)
  {
    /* deshabilita interrupciones mientras se
hace el cálculo */
    deshabilitaSensor();

    /* Como puede haber pasado más de 1 segundo
    se usa el número de milisegundos que pasaron
desde el cálculo anterior. También aplicamos el
factor de calibración para ajustar la salida
basado en el número de pulsos por segundo por
unidad de medida (litros/minuto en este caso) */

    // FLUJO EN L/MIN
    flujo = ((1000.0 / (millis() - tAnterior)) *
pulsos) / factorCalibracion;

    /* Guardamos el tiempo de este proceso. Como
deshabilitamos las interrupciones, la función
millis() no va a estar incrementando en este
punto, pero de todas maneras, retornara el valor
que tenía justo antes de deshabilitar las
interrupciones */
    tAnterior = millis();

    /* Dividimos el flujo en litros/minuto por 60
para determinar cuántos litros pasaron por el
sensor en este intervalo, y multiplicamos por
1000 para convertir a mililitros. */
    volMiliLitros = (flujo / 60) * 1000;

    // Acumulamos los mililitros de este segundo
    volActual += volMiliLitros;

    impActual= volActual * PRECIO;

    // Dibuja el volumen en el LCD
    imprimirVolumen(volActual);
    // Dibuja el importe en el LCD
    imprimirImporte(impActual);

    /* Reseteo el contador de pulsos antes de
habilitar las interrupciones */
    pulsos = 0;
  }
}
```

VI. Conclusión

En este trabajo pudimos enfrentar de lleno la complejidad de realizar una aplicación de tiempo real. Pudimos trabajar con microcontroladores y darnos cuenta de la importancia de las interrupciones. Si bien para el prototipo, trabajamos con un diseño improvisado adaptado a los distintos componentes electrónicos que pudimos conseguir.

El código del Arduino está implementado como una máquina de estados finitos, que está más cerca de nuestra formación. Combinar el poder de las interrupciones con ese concepto, nos permitió obtener un código legible, pese a la multiplicidad de cosas a controlar.

Por nuestras limitaciones en electrónica tratamos de elegir los componentes de la forma más sencilla posible, utilizamos una batería de gel para simplificar la alimentación de la electroválvula, y solamente utilizamos una resistencia para el circuito de lectura del sensor, que sacamos de la documentación.

Consideramos que el desarrollo de este trabajo fue muy bueno para adquirir un primer conocimiento de los sistemas de tiempo real.

VII. Referencias

[1] Reading Flow Rate with Water flow Sensor 2010 [online], http://www.seeedstudio.com/forum/viewtopic.php?f=4&t=989&p=3632#p3632 (Disponible: 30 de Junio 2013).

[2] Arduino Liquid Crystal Library [online], http://arduino.cc/en/Reference/LiquidCrystal (Disponible: 30 de Junio 2013).

[3] Arduino Finite State Machine Library [online], http://www.arduino.cc/playground/code/FiniteStateMachine (Disponible: 30 de Junio 2013).

[4] Practical Arduino: Water Flow Gauge 2009 [online], http://www.practicalarduino.com/projects/water-flow-gauge (Disponible: 30 de Junio 2013).

[5] Github Practical Arduino Water Flow Gauge source code 2009 [online], https://github.com/practicalarduino/WaterFlowGauge (Disponible: 30 de Junio 2013).