ANEXO I. Código de programación

MEJORAS EN UNA EXPLTACIÓN CAPRINA

2014

Contenido

1. Explicación general

Lectura RFID

En principio el sistema comprobará si se encuentra en estado manual o automático. Si está en modo manual se abrirá la válvula del depósito y el sistema permanecerá en reposo hasta activar el modo automático. Una vez que el modo automático es activado procederá a la lectura del chip de la cabra. Si la lectura se realiza con éxito enviará al PC el dato con la identificación. Para indicar al ordenador el final del envío se enviará un "B" al final de la cadena del nº de identificación. La lectura se efectúa cuando se activa el pulsador situado en la pezonera. Una vez leída la identificación de la cabra se habilita el ordeño y se cierra el depósito receptor de leche. Y así comienza la lectura del nivel.

Lectura de nivel y Control del ordeño

Para la lectura del nivel se deben conocer los valores de las salidas de los sensores de efecto hall. Puesto que son 16 sensores y la placa arduino sólo dispone de 6 entradas analógicas, se dispondrán de unos multiplexores de ocho entradas y una salida. Para manejar el multiplexor se usaran tres salidas digitales del arduino las cuales serán el nº de 0 a 8 en binario. Obtendremos los valores del sensor en un instante y se determinará en qué posición se encuentra el imán-flotador. Una vez determinado el nivel del depósito se estudiará la variación del nivel en un tiempo determinado para determinar cuándo finalizar el ordeño. La comunicación con el PC es simple. Si no ha finalizado el ordeño enviará una C y si por el contrario ya ha finalizado enviará el dato del volumen. Una vez cumplida la condición de fin de ordeño el sistema procederá a la parada automática del mismo desactivando la salida 9 conectada al Relé y a el vaciado del depósito activando la electroválvula con la salida 10.

Si se activa el pulsador de forzar parada y el sistema está en modo automático entonces el sistema parara el ordeño desactivando el relé, activará la electroválvula cerrando el depósito y enviará el dato de nivel al ordenador.

2. Programa arduino

```
#include <SoftwareSerial.h> //Biblioteca necesaria para la comunicacion serial
const int select [ ] ={2,3}; //Posicionadores de los multiplexores A B C +
float valor[30]; //Vector donde se almacenan los datos obttenidos de los senso float volumen; // Dato del ultimo registro de nivel obtenido
int contcabras; // Contador que sirve para simular 5 cabras
int inByte; //Entero donde se almacena el Byte recibido en el puerto
int entra=0; //Bandera que sirve para saber cuando entra en condición de posic
int finordeno; // 0 Sigue ordeñando ; 1 fin de ordeóño

//Variables para determinar el fin de ordeño,

int tiempo1=0; int tiempo2=0;
int diferenciatiempo;
float volumen1=0;
float volumen2;
 int estado; // Entero que utilizamos para valorar el estado de los pulsadores
 int direccion=1;
int estadoRFID=0;//0 no recibe RFID 1 Recibe RFID
int dataRFID[12];
int contRFID; //DEFINICIONES DE PIN EN ARDUINO
#define RFIDRX 4 // Pin de Recepcion Puerto serie
#define RFIDTX 5 // Pin de Transmision Puerto serie
SoftwareSerial SerialRFID(RFIDRX, RFIDTX); // Declaracion Puerto serie


void setup()
{

Serial.begin(9600);//Configuración de velocidad de comunicacion con el puerto

//Configuracion de pines como entradas o salidas

for(int bit = 0; bit < 2; bit++)

        {

                pinMode(select[ bit ], OUTPUT); //Posicionadores de los multiplexores A B C +

        }

pinMode(7, INPUT); // Interruptor de lectura RFID
pinMode(8, INPUT); // LED Indicador de RFID
```

```
pinMode(9, OUTPUT); // Relé que habilita el ordeño de ordeño +
pinMode(10, OUTPUT); // Válvula del depósito +
pinMode(11, INPUT); //Pulsador de parada ordeño +
pinMode(12, INPUT); //Interruptor manual-automático HIGH=automático LOW-Manu
}

void loop()
{
estado=digitalRead(12);
        while(estado==LOW) // Compruebo si el ordeño está en modo manual o
        automátic
                {
                digitalWrite(9,HIGH); //Activa relé que habilita el ordeño de ordeño
                digitalWrite(10,LOW);//Abre la válvula del depósito
                }
         while(estado==HIGH)
                {
                 estado=digitalRead(11);
        if (estado==HIGH)//Compruebo el pulsador de forzado fin de ordeño.
                {
                digitalWrite(10,LOW);//Abre la válvula del depósito
                digitalWrite(9,LOW);//Desactiva relé que habilita el ordeño de ordeño
                Serial.print(volumen);
                Serial.print("B");
                delay(10);
                finordeno=0;
                } else if(estado==LOW)
                {
                // LECTURA RFID

                        if(finordeno==1)
                        {
                        digitalWrite(8,HIGH);//ENCIENDE la luz hasta RECIBIR DATO
                        rfid
                        int pulsadorRFID=digitalRead(7);
                                if(pulsadorRFID==HIGH)
                                {
                                SerialRFID.flush(); // Vacio el buffer de RS485
                                SerialRFID.write("zr-CRC"); //envía el comando
                                memset(dataRFID, 0, sizeof(dataRFID)); // Se vacia el
                                array de datos
                                contRFID=0;
                                delay(100); //Retardo // Espero la respuesta del lector
                                        while(SerialRFID.available()) // Si el lector envia
                                        dato
```

```
                                        {
                                        delay(5);
                                        dataRFID[contRFID]= SerialRFID.read();
                                        contRFID++;
                                        }
                                //Cuando lee RFID de la cabra habilita el ordeño
                                digitalWrite(10,LOW); //Cierra la válvula del depósito
                                digitalWrite(9,HIGH); //Activa relé que habilita el ordeño
                                de ordeño
                                digitalWrite(8,LOW);//Apaga la luz hasta requerir lectura
                                finordeno=0;//Comienza el ordeño
                                }
                        }
        //LECTURA NIVEL
        int contsensor; //lectura multiplexores
        for(int canal = 0; canal < 8; canal++)
                {
                //Guarda en vector los canales del multiplexor pedidos
                getValor(canal);
                } //Recorre los valores del multiplexor y
                determina el nivel
                int nivel;
                for(int b=0 ;b<16;b++)
                        {
                                if(valor[b]>=560&&valor[b+1]>=560)
                                        {
                                        nivel=b+0.5; entra=1;
                                        } else
                                        if(valor[b]>=560&&valor[b+1]<560&&entra!=1)
                                         {
                                        nivel=b; entra=2;
                                         }else
                                        if(valor[b]<560&&valor[b+1]<560&&entra==0)
                                         {
                                        nivel=-1;
                                         }
                                tiempo2= millis()/1000;
                         }
entra=0;

// Condicion de fin de ordeño

volumen=nivel*(5/16);
volumen2=volumen;
```

```
diferenciatiempo=(tiempo2-tiempo1);

if(diferenciatiempo>10)
        {
         if(volumen1==volumen2)
        {
        finordeno=1;
        digitalWrite(10,LOW);//Abre la válvula del depósito
        digitalWrite(9,LOW);//Desactiva relé que habilita el ordeño de ordeñ
        digitalWrite(8,HIGH);//Enciende la luz hasta recibir dato de RFID
        }
        tiempo1=tiempo2;
        volumen1=volumen2;
        }

//Contador para simular 5 cabras(RFID)

if (contcabras>5)
        {
        contcabras=0;
        }

//Lectura del puerto y envío de datos

if (Serial.available() > 0)
        { // sólo si algo ha llegado
        inByte = Serial.read(); // lo lee
        Serial.flush();
        }

 if(direccion==48&finordeno==1)
        {

        //Envío el RFID de las Cabras

        Serial.print("A000");
        Serial.print(contcabras);
        contcabras++;
        Serial.println("B"); //Envio Volumen
        /* for(int b=0 ; b<16 ; b++) //Para imprimir los valores de los {
        Serial.print(valor[b]); }*/
        Serial.print(volumen);
        Serial.print("C");
        delay (10);//Para sincronizar el envío y la recepción
```

```
        }
}
}
 } //FINAL

//FUNCION

//Función que lee las señales de los multiplexores y guarda los datos en el vectores

void getValor( int canalg)
        {
         //function que returna el valor del canal seleccionado en el mux
        for(int bit = 0; bit < 3; bit++)
        {
         int pin = select [ bit ];//Se pondrá el pin correspondiente,2,3 y 4 suce
        int isBitSet = bitRead(canalg, bit);//bit read te devuelve el bit del nu
        digitalWrite(pin, isBitSet); // Serial.print(isBitSet);
        }
valor[canalg] = analogRead(A0);
valor[canalg+8] = analogRead(A1);
 }
```

3. PROCESSING

```
import processing.serial.*;
 import java.io.*;
Serial puerto;
int datosEntrantes; //Entero para guardar el byte de entrada
String direccionFichero;//Direccion del fichero de texto donde se almacenan
 int contadordir=0;//Dirección correspondiente al puesto de ordeño y sitema ins int
contRFID=0;//Posicion del vector de datos de RFID
int RFID1[];//Vector donde se almacenan los datos de RFID
 int contnivel;//Posicion del vector de datos de nivel
int nivel[];//Vector donde se almacenan los datos de nivel //Enteros que sirven para ir
leyendo los distintos datos de la trama
int banderaestado=0;
 int banderaescribeestado=0;
```

```
 int banderaescribeRFID=0;
int estado; //Entero para fecha y hora
int mes;
 int dia;
 int year;
 int mesanterior=0;
int diaanterior=0;
int yearanterior=0;
int hora;
int min;
int seg; //Objetos para escribir en ficheros
 DataOutputStream salida;
FileOutputStream fileOut;
 FileWriter file;




void setup()

        {
        println(Serial.list());// puertos serie disponibles
        puerto = new Serial(this, Serial.list()[0], 9600); // Configuración del puer
        //Inicialización de vectores
        RFID1=new int[14];
        nivel=new int[14];
         //FECHA
        mes=month();
         dia=day();
        year=year();
        direccionFichero=("D:volumencabras.txt");//Direccion del fichero Se creará u
        }

        //COMIENZO PROGRAMA

void draw()
 {
//Comprobar y escribir fecha
if(dia!=diaanterior&mes!=mesanterior&year!=yearanterior)
        {
        diaanterior=dia;
        mesanterior=mes;
```

```
        yearanterior=year;
        escribefecha(direccionFichero);
        escriber(direccionFichero);//Escribe salto de linea
        }
//Inicio del contador de la dccn

if(contadordir>0)
        {
         contadordir=0;
        //SOLO HAY UNA DIRECCION 0
        println("pone el contador a "+contadordir);
        }
//Escribe dccn en el puerto
puerto.clear();//borro el buffer del puerto para no tener problemas de sobreescritura
puerto.write(String.valueOf(contadordir));// Escribe en el puerto la dirección
delay(100);//Para sincrinización de envío y recepción.
 println(contadordir);//Imprime en pantalla de processing la dirección enviada

//Lectura del puerto
while (puerto.available() > 0)//Mientras el puerto este disponible
        {
        datosEntrantes = puerto.read();//Lee Byte del puerto
        //Lee el primer byte que define el estado en el que se encuentra el orde
        if(banderaestado==0)
                {
                 estado=datosEntrantes; banderaestado=1;
                 }
        //ESTADO A FIN ORDEÑO

        if(estado==65)//Estado A de finalizado ordeño
                {
                if(banderaescribeestado==0)
                        {
                        escribefecha(direccionFichero);
                        escribet(direccionFichero);
                        escribepuesto(direccionFichero);
                        escribet(direccionFichero);
                         escribehora(direccionFichero);
                        escribet(direccionFichero);
                        banderaescribeestado=1;
                        }else if(banderaescribeRFID==0)
                        {
                        RFID1[contRFID]=datosEntrantes;//Guarda dato en vector
                        println("guarda en"+contRFID+"dato"+datosEntrantes);
```

```
                    contRFID++; // Comprueba los datos que lee hasta llegar a la
                    condicon
                    if(datosEntrantes==66)
                            {
                            escribeRFID0 (direccionFichero); // Escribo el nº de
                            escribet(direccionFichero);//
                             println("aumenta el contador a"+contadordir);
                            contRFID=0; banderaescribeRFID=1;
                            }
                    }else
                    {
                    //ESCRIBE NIVEL
                    nivel[contnivel]=datosEntrantes;
                    println("guarda en"+contnivel+"dato"+datosEntrant contnivel++;
                    if(datosEntrantes==67)
                            {
                            escribet(direccionFichero);
                            escribenivel(direccionFichero);
                            escriber(direccionFichero);
                            puerto.clear();
                            contnivel=0;
                             banderaescribeRFID=0;
                            banderaescribeestado=0;
                            banderaestado=0;
                             contadordir++;
                            contnivel=0;
                             println("aumenta el contador a"+contadordi break;
                            }
                }
}
 }
 }//Final Drow

//FUNCIONES

//Funcion escribeRFID

void escribeRFID0(String direccionFicherof)
        {
        try
                 {
                fileOut=new FileOutputStream(direccionFicherof,true);
                 //Crea un fichero
                salida=new DataOutputStream(fileOut);
                salida.writeInt(RFID1[0]);
```

```
println("Escribe en"+direccionFichero+"dato:"+RFID1[0]);
salida.writeInt(RFID1[1]);
println("Escribe en"+direccionFichero+"dato:"+RFID1[1]);
salida.writeInt(RFID1[2]);
println("Escribe en"+direccionFichero+"dato:"+RFID1[2]);
salida.writeInt(RFID1[3]);
println("Escribe en"+direccionFichero+"dato:"+RFID1[3]);
salida.writeInt(RFID1[4]);
 println("Escribe en"+direccionFichero+"dato:"+RFID1[4]);
fileOut.close();
 }
catch(Exception e)
 {
println("Error: Can't open file!");
}
 }
```

```
//FUNCION ESCRIBE TABULACION

void escribet(String direccionFicherof)
{
try
{
file=new FileWriter(direccionFicherof,true);
 file.write("\t");
file.close();
}
catch(Exception e)
{
println("Error: Can't open file!");
}
}

//FUNCION ESCRIBE HORA

void escribehora(String direccionFicherof)
{
try
{
 fileOut=new FileOutputStream(direccionFicherof,true); //Crea un fichero
salida=new DataOutputStream(fileOut);
  hora=hour();
```

```
                min=minute();
                seg=second();
                salida.write((hora/10)+48);
                salida.write((hora%10)+48);
                salida.write(58);
                salida.write((min/10)+48);
                salida.write((min%10)+48);
                salida.write(58);
                salida.write((seg/10)+48);
                salida.write((seg%10)+48);
                fileOut.close();
                }
        catch(Exception e)
                {
                println("Error: Can't open file!");
                }
        }


//Funcion escribe NIVEL

void escribenivel(String direccionFicherof)
        {
         try
                {
                fileOut=new FileOutputStream(direccionFichero,true);
                salida=new DataOutputStream(fileOut);
                for(int cont=0;cont<5;cont++)
                        {
                         salida.writeInt(nivel[cont]);
                         println("Escribe en"+direccionFichero+"dato:"+nivel[contnivel]);
                         } fileOut.close(); }
                        catch(Exception e)
                                {
                                println("Error: Can't open file!");
                                }
        }

//fUNCION ESCRIBE SALTO DE LINEA

void escriber(String direccionFicherof)
        {
        try
                {
```

```
                file=new FileWriter(direccionFichero,true);
                file.write("\r\n");
                println("Escribe en"+direccionFichero+"salto de linea"); file.close();
                }
        catch(Exception e)
                {
                println("Error: Can't open file!");
                }
        }

//FUNCION ESCRIBE Nº PUESTO

void escribepuesto(String direccionFicherof)
        {
                try
                        {
                        fileOut=new FileOutputStream(direccionFichero,true);
                        salida=new DataOutputStream(fileOut);
                        salida.writeInt(contadordir+48);
                        println("Escribe en"+direccionFichero+"dato:"+nivel[contnivel]);
                        fileOut.close();
                        }
                catch(Exception e)
                {
                println("Error: Can't open file!");
                }
        }

//FUNCION ESCRIBE ESTADO DEL PUESTO

void escribeestado(String direccionFicherof, int estadof)
        {
        try
                {
                fileOut=new FileOutputStream(direccionFichero,true);
                salida=new DataOutputStream(fileOut);
                salida.writeInt(estadof);
                println("Escribe en"+direccionFichero+"dato:"+nivel[contnivel]);
                fileOut.close();
                }
        catch(Exception e)
                {
                println("Error: Can't open file!");
                }
```

```
        }

//FUNCION ESCRIBE Fecha

void escribefecha(String direccionFicherof)
        {
                try
                {
                fileOut=new FileOutputStream(direccionFicherof,true); //Crea un fichero
                salida=new DataOutputStream(fileOut);
                 salida.write((dia/10)+48);
                salida.write((dia%10)+48);
                salida.write(47);
                salida.write((mes/10)+48);
                 salida.write((mes%10)+48);
                salida.write(47);
                 salida.write((year/1000)+48);
                salida.write(((year%1000)/100)+48);
                salida.write((((year%1000)%100)/10)+48);
                salida.write((((year%1000)%100)%10)+48);
                fileOut.close();
                }
                catch(Exception e)
                 {
                 println("Error: Can't open file!");
                 }
        }
```