

ESCUELA SUPERIOR DE INFORMÁTICA

UNIVERSIDAD DE CASILLA-LA MANCHA



Automatización Industrial

Parking Sexta práctica

Jose Domingo López López
josed.lopez1@alu.uclm.es

Raúl Arias García
raul.arias2@alu.uclm.es

Grupo: ELF 08 (Tarde)
7 de Mayo del 2009

Índice de contenidos

1.	Introducción.....	1
2.	Imágenes del prototipo junto a su referencia.....	2
3.	Descripción del funcionamiento del programa.....	7
4.	Videos de demostración.....	12
5.	Conclusiones.....	13
6.	Bibliografía.....	14
7.	Apéndice I: Código fuente.....	15
	Program.cs	15

Índice de figuras

Ilustración 1: Prototipo del parking con la barrera levantada.....	7
Ilustración 2: Detalle de la barrera cerrada.....	8
Ilustración 3: Detalle de los pulsadores que controlan la barrera.....	9
Ilustración 4: Detalle del sin-fin, el motor y la rueda dentada.	10

1. Introducción

Esta es la práctica más compleja de las que hemos realizado a lo largo del curso, en cuanto a montaje y a programación.

Sobre todo nos ha llevado un mayor esfuerzo en el montaje del prototipo debido a la complejidad del mismo, desde su estructura hasta su circuitería. Además hemos tenido complicaciones diversas por ejemplo con la longitud de los cables, los cuales son en su mayoría demasiado cortos para las conexiones.

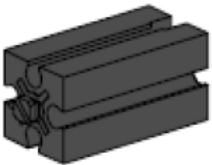
Ha sido el prototipo en el que más piezas hemos requerido para su montaje y más tiempo hemos invertido en su diseño, que hemos intentado que sea lo más fiel posible a la imagen proporcionada en el enunciado de la práctica.

Además ha sido la práctica en la que mas componentes eléctricos y mecánicos hemos utilizado.


2. Imágenes del prototipo junto a su referencia

A continuación se muestra un listado de las piezas utilizadas para realizar el diseño, así como su referencia y su imagen:


5 piezas, ref.:32879

	32879	Baustein 30 Building block 30
---	-------	----------------------------------


2 piezas, ref.:32882

	32882	Baustein 15 mit 2 Zapfen Building block 15 with 2 pins
--	-------	---


Una pieza, ref.:32321

	32321	Baustein 15 mit Ansenkung Building block 15 with counterbore
---	-------	---


2 piezas, ref.:32881

	32881	Baustein 15 Building block 15
---	-------	----------------------------------

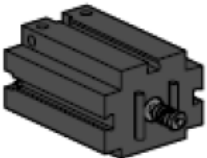
10 piezas, ref.:31360

		Litze 2-adrig rot/grün Lead red/green
	36229	120 mm
	36210	250 mm
	36977	300 mm
	36382	1000 mm
	31360	2000 mm
	77312	3000 mm
	36035	7000 mm

18 piezas, ref.:31336 y 18 piezas, ref.:31337

		Flachstecker montiert Flat plug
	31337	rot / red
	31336	grün / green

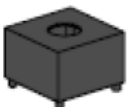
Una pieza, ref.:32293

	32293	S-Motor 6-9V $\overline{\text{---}}$
		S-Motor 6-9V $\overline{\text{---}}$


Una pieza, ref.:36134

	36134	Fototransistor montiert Photo transistor assembled


Una pieza, ref.:36532

	36532	Störlichtkappe, Öffnung 6 Masking light cap, hole 6,0

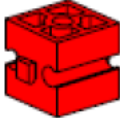
Tres piezas, ref.:38216

	38216	Leuchtstein mit Steckfassung Plug in light holder
---	-------	--


Tres piezas, ref.:37869

	37869	Kugelstecklampe 9V $\overline{\sim}$ 0,1 A Bulb lamp 9V $\overline{\sim}$ 0,1 A
---	-------	--

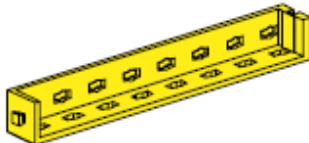
3 piezas, ref.:32064

	32064	Baustein 15 mit Bohrung Building block 15 with bore
---	-------	--

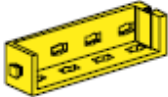
Una pieza, ref.:35969

	35969	Reedkontakt- und Kabelhalter Reed contact and cable clamp
---	-------	--

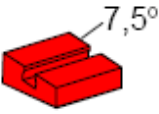
Una pieza, ref.:36294

	36294 36293	Winkelträger 120 Angle girder 120 gelb/yellow schwarz/black
---	----------------	--


3 piezas, ref.:36297

	36297 36921	Winkelträger 60 Angle girder 60 gelb/yellow schwarz/black
---	----------------	--

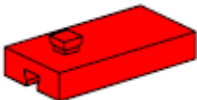
Tres piezas, ref.:32071

	32071	Winkelstein 7,5° Angular block 7,5°
---	-------	--


Una pieza, ref.:37468

	37468	Baustein 7,5 Building block 7,5
---	-------	------------------------------------

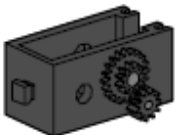
Dos piezas, ref.:35049

	35049	Baustein 15x30x5 mit Nut und Zapfen Building block 15x30x5 with groove and pin
---	-------	---


Una pieza, ref.:37237

	37237	Baustein 5 Building block 5
---	-------	--------------------------------


Una pieza, ref.:31078

	31078	U-Getriebe Motor reducing gearbox
---	-------	--------------------------------------

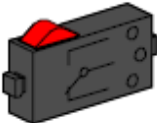
Una pieza, ref.:35072

	35072	Rastschnecke Worm
---	-------	----------------------


Una pieza, ref.:35405

	35405	V-Achse 4x80 V-Axle 4x80
---	-------	-----------------------------


Cuatro piezas, ref.:37783

	37783	Mini-Taster Mini switch
---	-------	----------------------------

Una pieza, ref.:31779

	31779	Kettenrad Z20 Chain wheel T20
---	-------	----------------------------------

Una pieza, ref.:31058

	31058	Nabenmutter mit Scheibe Hub nut
---	-------	------------------------------------

3. Descripción del funcionamiento del programa

El prototipo cuenta con dos pulsadores que controlan la entrada y la salida al parking, uno para cada acción. Una vez pulsado uno de ellos la barrera se abre. Una vez abierta del todo, espera a que el vehículo pase entre la célula fotovoltaica y la bombilla, interrumpiendo su conexión, de este modo la barrera se cerrará y se actualizará el contador de coches que han entrado o salido. Si se abre la barrera, pero no pasa ningún vehículo durante cinco segundos, se produce una incidencia: con lo cual el zumbador suena y se toma parte en el fichero de incidencias tomando todos los datos.

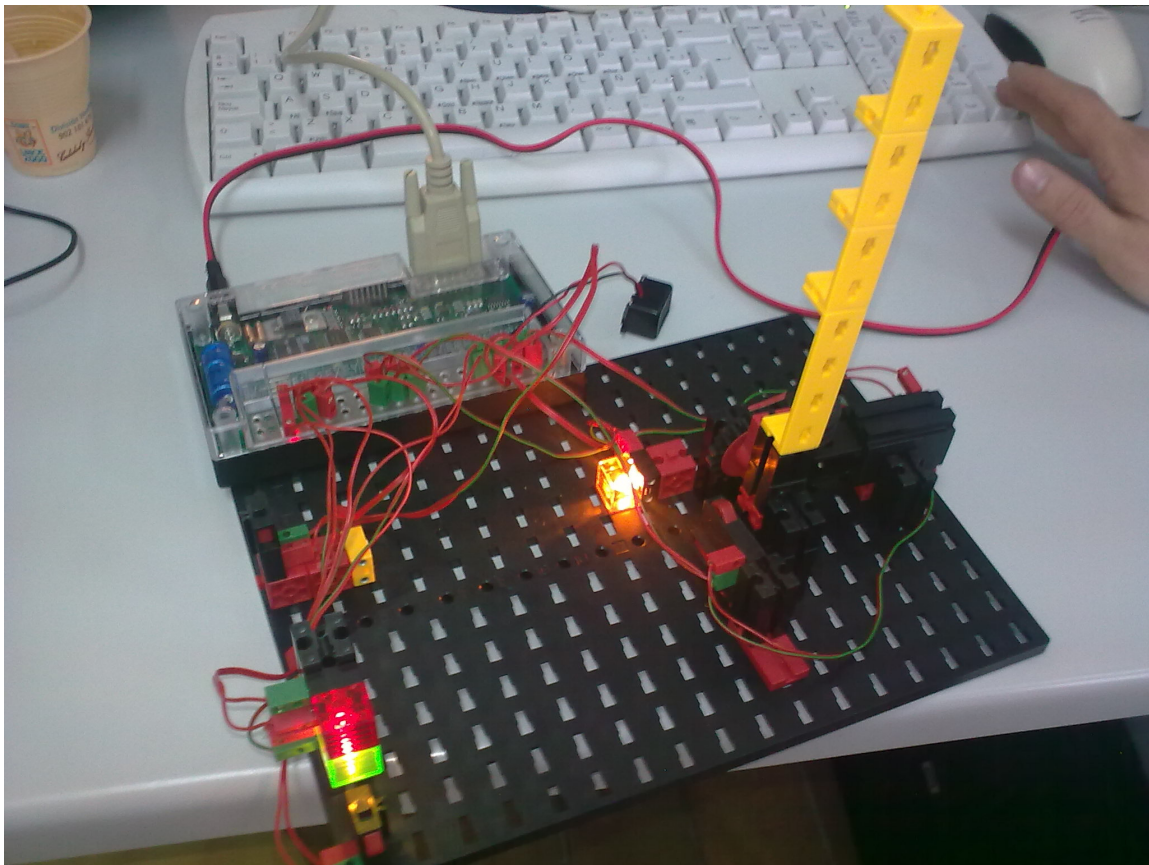


Ilustración 1: Prototipo del parking con la barrera levantada.

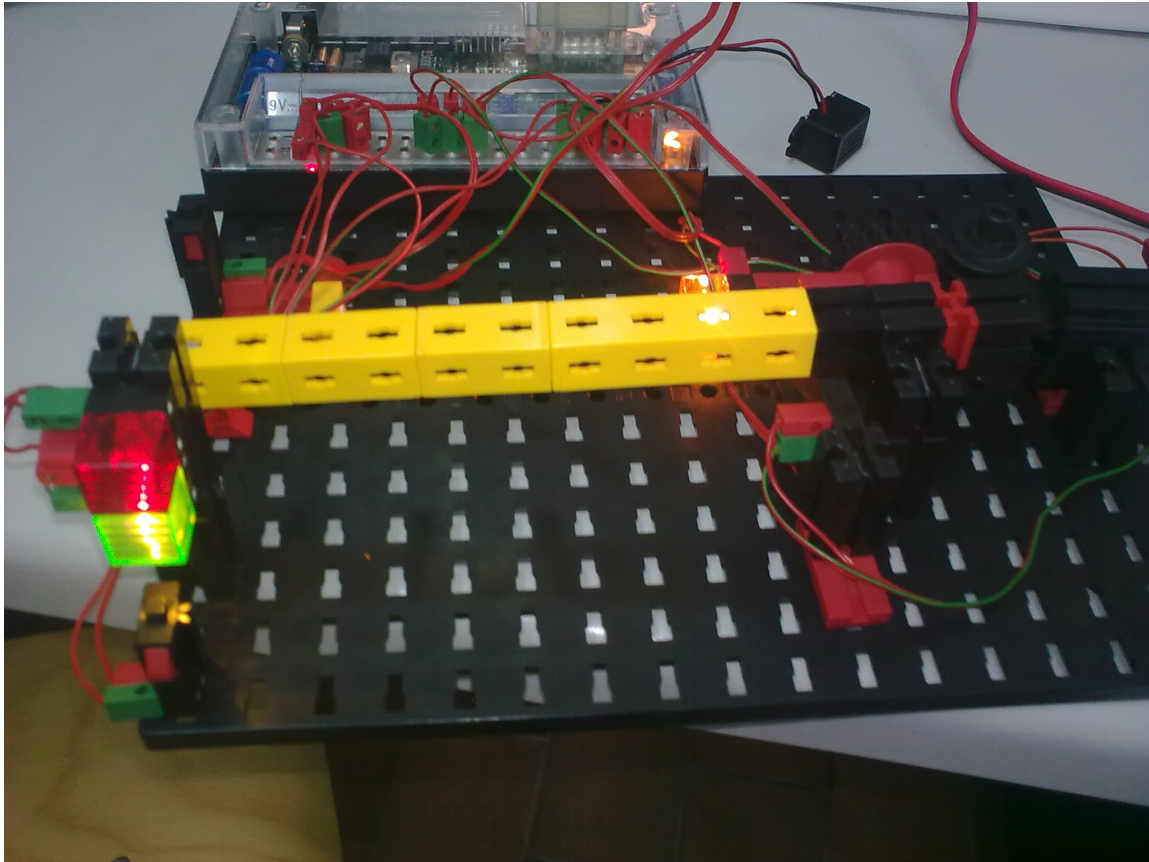


Ilustración 2: Detalle de la barrera cerrada.

El tamaño máximo del parking lo hemos establecido en dos coches con el fin de mostrar los ejemplos en menos tiempo y más fácilmente.

Además, el prototipo cuenta con otros dos pulsadores que actúan como finales de carrera para controlar el movimiento de la barrera. De este modo, dichos pulsadores paran el motor cuando la barrera se ha abierto del todo o se ha cerrado del todo.

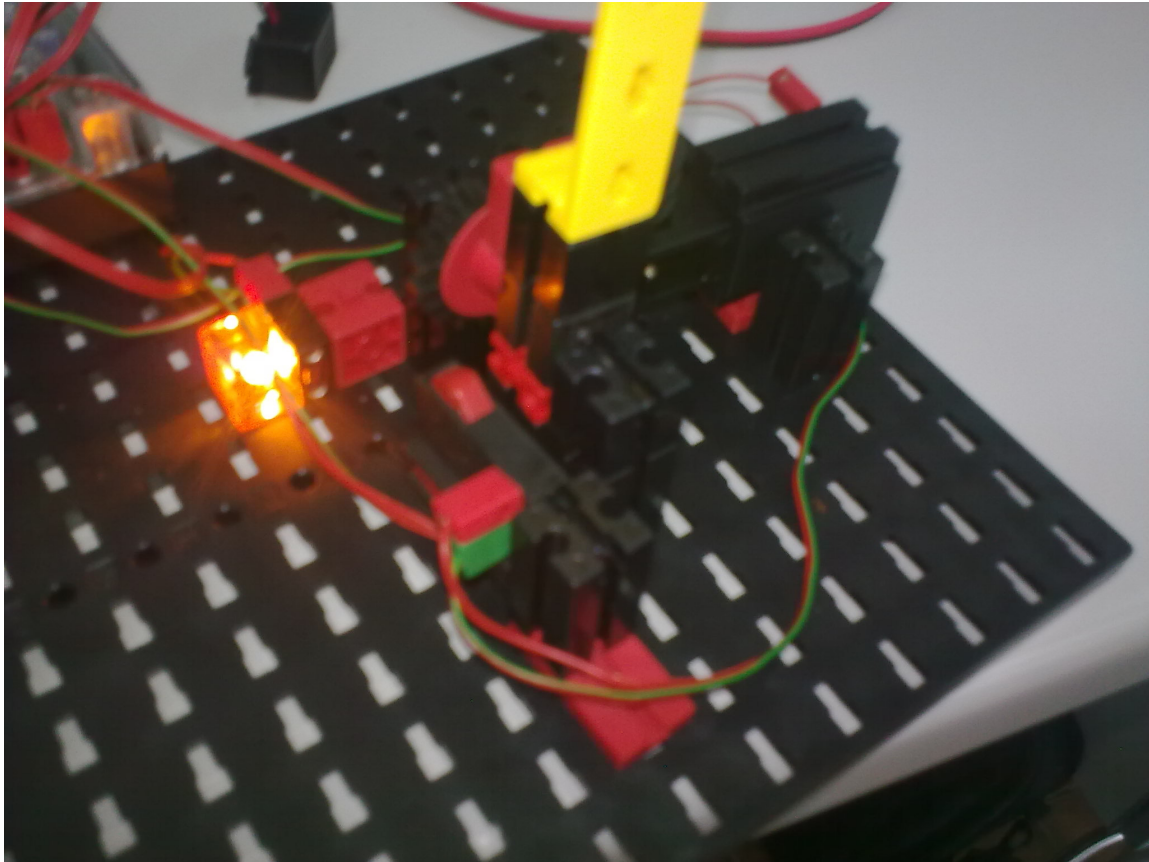


Ilustración 3: Detalle de los pulsadores que controlan la barrera.

La barrera es movida por el motor, que conectado a una reductora hace mover a un sin-fín que mueve una rueda dentada en un sentido u otro según convenga: abrir la barrera levantándola o cerrar la barrera bajándola.

Las luces del semáforo indican cuando el parking tiene plazas disponibles (luz verde) y cuando está lleno de coches (luz roja). Para poder conectar las luces del semáforo hemos tenido que conectar los positivos a la entrada M1 y a masa. Para poder conectar ambos a masa hemos tenido que realizar un empalme. Gracias a esta conexión funciona correctamente el semáforo, como se indica en el enunciado de la práctica y tal y como se muestra en el video de demostración.

Es destacable explicar dos aspectos del diseño que nos han causado dificultades:

- Como ajustar el motor para que el sin-fín que está conectado a la reductora del propio motor encaje bien con la rueda dentada que levanta la barrera con su

movimiento. Dicha dificultad la salvamos con tres piezas ref.:32071 (ver apartado anterior) que nos permitieron ajustar la altura del motor y así, la del sin-fin encajando perfectamente con la rueda dentada.

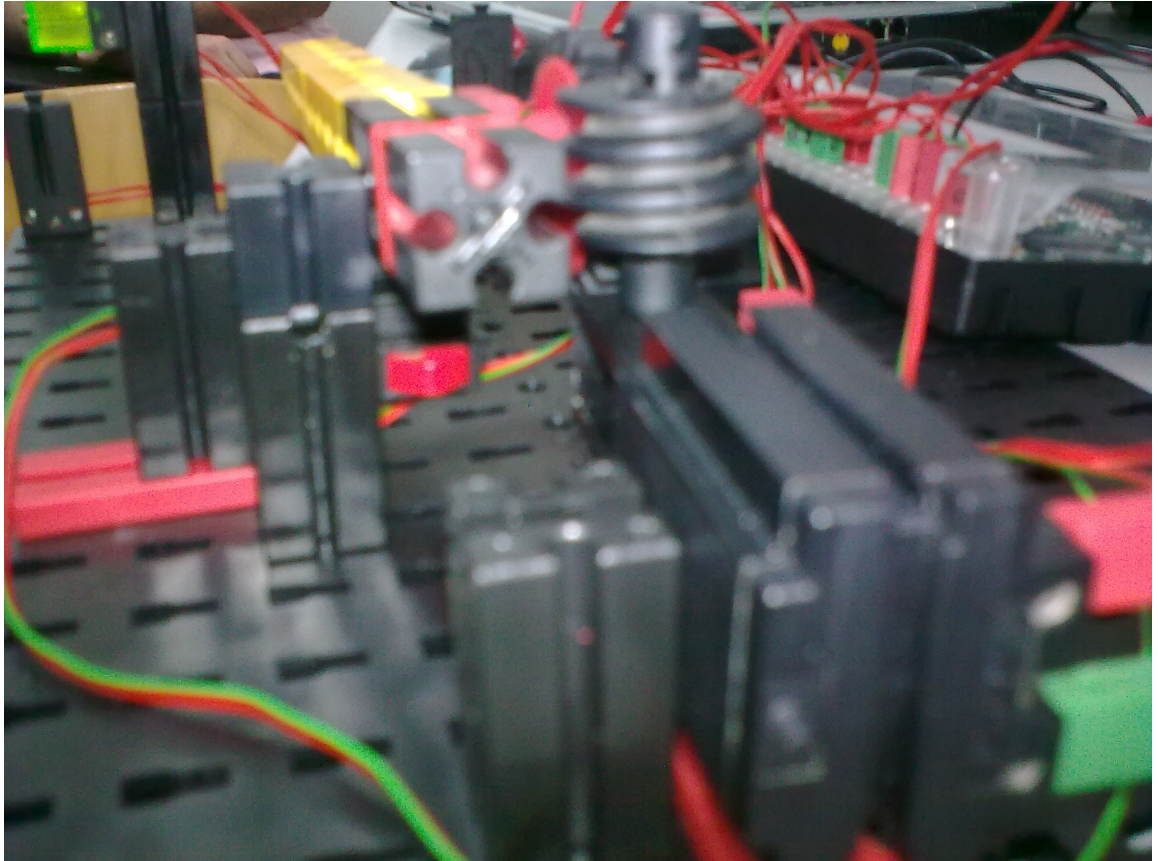


Ilustración 4: Detalle del sin-fin, el motor y la rueda dentada.

- Como transmitir el movimiento giratorio de la rueda dentada a la barrera para levantarla. Para ello, pasamos una pieza en forma de palo a través de la pieza ref.:32321 del agujero que tiene. La rueda tiene un saliente que está encajado en un hueco puesto en la barrera, en este apoyo se basa el movimiento de la misma, ya que la rueda dentada en su giro, levanta la barrera mediante dicho apoyo.

En cuanto a la programación del prototipo, la aplicación desarrollada genera dos ficheros de salida en formato CSV (comma-separated values). Hemos elegido usar ese formato con la finalidad de poder abrir los ficheros con hojas de cálculo como pueden ser Excel y Openoffice. Un fichero de salidas/entradas de los coches al parking; otro fichero de

incidencias que se dan a conocer mediante el prototipo haciendo pitar al zumbador. Ambos ficheros tienen el mismo formato para mostrar los datos **Acción; Fecha; Hora**, siendo:

- Acción: Entrada o Salida.
- Fecha: Que muestra la fecha del día que se produjo la entrada o la salida, con el formato día/mes/año. Ejemplo: 04/05/2009
- Hora: Indica la hora exacta cuando empezó la acción, con el formato Hora:Minuto:Segundo. Ejemplo: 9:45:57

Como se puede comprobar, se separan por puntos y coma los distintos campos existentes.

4. Videos de demostración

En los videos que acompañan esta práctica se puede comprobar cómo funciona correctamente el prototipo que hemos diseñado de acuerdo a las exigencias del enunciado.

- ✓ En el video 1: Demostración, se puede ver como entran dos coches seguidos sin ningún problema, uno después de otro. En ambos casos, llega el coche, se activa el pulsador de entrada y se abre la barrera. Una vez abierta del todo el coche entra interrumpiendo la recepción de luz por parte de la célula fotovoltaica y activando la acción de cerrar la barrera. El segundo coche actúa del mismo modo. El parking está lleno pues como se ha comentado antes hemos reducido su capacidad a 2 coches para mostrar mejor su funcionamiento.

Al llegar un tercer coche, e intentar entrar muestra un mensaje por consola advirtiéndole que el parking está completo, además de la luz roja del semáforo que permanece encendida.

Cuando se activa el interruptor de salir, se abre la barrera para que salga el coche. Una vez fuera el coche el semáforo se pone en verde, ya que hay una plaza libre.

Por último se muestra un ejemplo de cómo se activa el pulsador para entrar al parking pero no pasa ningún coche en cinco segundos, en este caso se produce una incidencia y se activa el zumbador, que suena hasta que se pulsa el botón enter.

De todas las entradas, salidas e incidencias se guardan los informes correspondientes en el ordenador.

- ✓ En el video 2: Ficheros, se puede ver como se han generado los ficheros correspondientes a las entradas, salidas e incidencias de la demostración mostrada en el video anterior.

5. Conclusiones

Las prácticas de esta asignatura, y más concretamente esta última, nos han dado una visión general de los pasos a seguir para diseñar y construir un prototipo de un sistema automatizado. De todos modos, pensamos que estos sistemas que hemos construido en el laboratorio dependen mucho del software y que en realidad son más dependientes del hardware: hablamos de contadores, etc.

Debido a nuestra experiencia podemos concluir que se pueden realizar prototipos de proyectos reales, ya que este mismo prototipo se podría llevar a cabo a la realidad funcionando tal y como está previsto. Por ello, este juego de piezas y la interfaz inteligente son herramientas muy útiles a la hora de diseñar y desarrollar prototipos de proyectos.

A modo de sugerencia, sería interesante que no sólo se realizasen las prácticas con fishertechnik sino que se incluyesen algunas prácticas con Lego Mindstorm ya que se puede hacer de él un uso más complejo y profesional.

6. Bibliografía

- Gartoon Icons (GNU General Public License):
<http://www.zeusboxstudio.com>
- Página de Ulrich-Müller
<http://www.ulrich-mueller.de/csecke.htm>
- Wikipedia: La enciclopedia libre
http://es.wikipedia.org/wiki/C_Sharp
http://es.wikipedia.org/wiki/Microsoft_Visual_Studio
- Página de #develop (Sharp Develop):
<http://www.icsharpcode.net/OpenSource/SD/>
- Joseph Albahari; Ben Albahari, “C# 3.0 in a Nutshell, 3rd Edition”.

7. Apéndice I: Código fuente

Program.cs

```
using System;
using System.Collections.Generic;
using System.Text;
using FishFa30;
using System.IO;

namespace Parking
{
    class Program
    {
        // Instancia de la interfaz
        public static FishFace fish = null;

        // Entradas
        public static Nr pulsadorEntrada = Nr.E1;
        public static Nr pulsadorSalida = Nr.E2;
        public static Nr finCarreraCerrada = Nr.E3;
        public static Nr celulaAcceso = Nr.E4;
        public static Nr finCarreraAbierta = Nr.E5;

        // Salidas
        public static Nr lamparaVerde = Nr.M1;
        public static Nr lamparaRoja = Nr.M1;
        public static Nr zumbador = Nr.M2;
        public static Nr lamparaCelulaOptica = Nr.M3;
        public static Nr motorBarrera = Nr.M4;

        // Variables de control
        public static int capacidadActual = 0;
        public static int capacidadMaxima = 2;
        public static int deadline = 5;
        public static Boolean parkingLleno = false;

        public static void Main(string[] args)
        {
            fish = new FishFace(true, false, 0);
            try
            {
                Boolean pulsadorDetectado = false;
                DateTime inicio;
                DateTime fin;
                Boolean dentro = false;
                Boolean fuera = false;

                Help();
                Console.WriteLine("\nPresione ENTER para continuar...");
                Console.ReadLine();
                fish.OpenInterface(Port.COM1);
                Console.WriteLine("Iniciando sistema...");
                // Encender la lampara
                fish.SetMotor(lamparaCelulaOptica, Dir.On);
                actualizarLamparas();
                Console.WriteLine("Sistema iniciado.");
                while (true)
                {
                    pulsadorDetectado = fish.GetInput(pulsadorEntrada);
```

```

// Si se pulsa el pulsador de entrada al parking...
if (pulsadorDetectado)
{
    if (!parkingLleno)
    {
        Console.WriteLine("Abriendo barrera...");
        abrirBarrera();
        Console.WriteLine("Barrera abierta. Entre, por
favor.");

        inicio = DateTime.Now;
        fin = DateTime.Now;
        while ((fin - inicio).Seconds < deadline) &&
!dentro)
        {
            if (!fish.GetInput(celulaAcceso)) dentro = true;
            fin = DateTime.Now;
        }
        if (dentro) // El coche ha pasado dentro del deadline
        {
            escribirAccesos(1);
            dentro = false;
            cerrarBarrera();
            capacidadActual++;
            actualizarLamparas();
        }
        else // El coche NO ha pasado dentro del deadline
        {
            escribirIncidencias(1);
            fish.SetMotor(zumbador, Dir.On);
            Console.WriteLine(":: ERROR: No ha entrado ningun
coche...");
            Console.WriteLine("Pulse ENTER para desactivar la
alarma.");

            Console.ReadKey();
            dentro = false;
            fish.SetMotor(zumbador, Dir.Off);
            cerrarBarrera();
        }
    }
    else
    {
        Console.WriteLine("Parking lleno");
        fish.Pause(500);
    }
}
pulsadorDetectado = fish.GetInput(pulsadorSalida);
// Si se pulsa el pulsador de salida del parking...
if (pulsadorDetectado)
{
    Console.WriteLine("Abriendo barrera...");
    abrirBarrera();
    Console.WriteLine("Barrera abierta. Salga, por favor");
    inicio = DateTime.Now;
    fin = DateTime.Now;
    while (((fin - inicio).Seconds < deadline) && !fuera)
    {
        if (!fish.GetInput(celulaAcceso)) fuera = true;
        fin = DateTime.Now;
    }
    if (fuera)
    {
        escribirAccesos(2);
        fuera = false;
    }
}

```

```

        cerrarBarrera();
        capacidadActual--;
        actualizarLamparas();
    }
    else
    {
        escribirIncidentes(2);
        fish.SetMotor(zumbador, Dir.On);
        Console.WriteLine(":: ERROR: No ha entrado ningun
coche...");
        Console.WriteLine("Pulse ENTER para desactivar la
alarma.");

        Console.ReadKey();
        fuera = false;
        fish.SetMotor(zumbador, Dir.Off);
        cerrarBarrera();
    }
    }
}

catch (FishFaceException)
{
    Console.WriteLine("ERROR:: No se pudo abrir el interfaz");
    fish.CloseInterface();
    Console.ReadKey();
}

}

public static void escribirFichero(String fichero, String linea)
{
    String accessFile = System.AppDomain.CurrentDomain.BaseDirectory +
fichero;
    File.AppendAllText(accessFile, linea);
}

public static void escribirIncidentes(int tipoAcceso)
{
    String date = DateTime.Now.ToString("yyyy/MM/dd");
    String time = DateTime.Now.ToString("HH:mm:ss");
    String file = "incidencias.csv";
    String acceso = "Unknown operation"; // tipo == 1 (entrada); tipo ==
2 (salida)
    if (tipoAcceso == 1) acceso = "Entrada";
    if (tipoAcceso == 2) acceso = "Salida";
    String line = acceso + ";" + date + ";" + time + "\n";
    escribirFichero(file, line);
    Console.WriteLine("Incidencia registrada: " + line);
}

public static void escribirAccesos(int tipoAcceso)
{
    String date = DateTime.Now.ToString("yyyy/MM/dd");
    String time = DateTime.Now.ToString("HH:mm:ss");
    String file = "accesos.csv";
    String acceso = "Unknown operation"; // tipo == 1 (entrada); tipo ==
2 (salida)
    if (tipoAcceso == 1) acceso = "Entrada";
    if (tipoAcceso == 2) acceso = "Salida";
    String line = acceso + ";" + date + ";" + time + "\n";
    escribirFichero(file, line);
    Console.WriteLine("Evento registrado: " + line);
}
}

```

```

public static void abrirBarrera()
{
    Boolean stop = false;
    fish.SetMotor(motorBarrera, Dir.Right);
    while (!stop)
    {
        stop = fish.GetInput(finCarreraAbierta);
    }
    fish.SetMotor(motorBarrera, Dir.Off);
}

public static void cerrarBarrera()
{
    Boolean stop = false;
    fish.SetMotor(motorBarrera, Dir.Left);
    while (!stop)
    {
        stop = fish.GetInput(finCarreraCerrada);
    }
    fish.SetMotor(motorBarrera, Dir.Off);
}

public static void actualizarLamparas()
{
    if (capacidadActual < capacidadMaxima)
    {
        fish.SetMotor(lamparaVerde, Dir.Left);
        parkingLleno = false;
    }
    else
    {
        fish.SetMotor(lamparaRoja, Dir.Right);
        parkingLleno = true;
    }
}

public static void Help()
{
    Console.WriteLine("Autores: José Domingo López López y Raúl Arias
García");
    Console.WriteLine("Asignatura: Automatización Industrial");
    Console.WriteLine("Grupo: 08 (tarde)");
    Console.WriteLine("Curso: 4, 2008-009");
    Console.WriteLine();
    Console.WriteLine("Esta aplicación crea dos ficheros para mantener
una traza de incidencias y entrada/salida de vehículos. Ambos ficheros son
generados con formato CSV (comma-separated values) y serán creados en el
directorio de la aplicación:\n" + System.AppDomain.CurrentDomain.BaseDirectory);
    Console.WriteLine();
    Console.WriteLine("La conexión de los periféricos al interfaz debe
ser la siguiente: ");
    Console.WriteLine("\t* Salida M1 -> Lámparas de semáforo (cableado en
GND)");
    Console.WriteLine("\t* Salida M2 -> Zumbador");
    Console.WriteLine("\t* Salida M3 -> Lámpara incandescente en célula
óptica");
    Console.WriteLine("\t* Salida M4 -> Motor de accionamiento de la
barrera");
    Console.WriteLine("\t* Entrada E1 -> Pulsador para ENTRAR al
parking");
    Console.WriteLine("\t* Entrada E2 -> Pulsador para SALIR del
parking");
    Console.WriteLine("\t* Entrada E3 -> Interruptor de fin de carrera

```

```
para CERRAR la barrera");  
    Console.WriteLine("\t* Entrada E4 -> Célula óptica de acceso al  
parking");  
    Console.WriteLine("\t* Entrada E5 -> Interruptor de fin de carrera  
para la APERTURA de la barrera");  
    }  
}  
}
```