Comunicacións industriais

Práctica 01: Visualización do valor dun byte en binario en oito leds

Descrición da práctica:

Visualización dun dato de tipo byte (0-255) en binario en oito leds e envío a un documento de Excel.

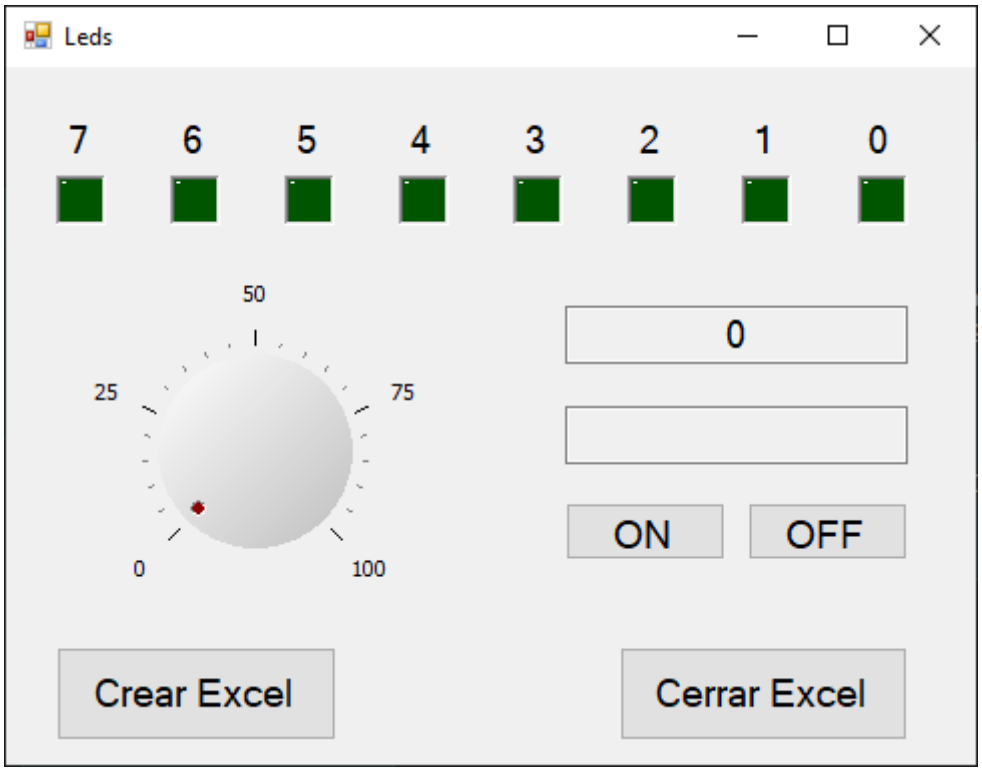
Coñecementos previos:

- Programación en C#.
- Elaboración dunha librería de enlace dinámico.
- Instalación dos compoñentes iocomp para C#.

Material necesario:

- Contorna de programación Visual Studio 2017 con C# instalado.
- Controles ActiveX de iocomp.
- Microsoft Excel (son válidas as versións actuais).

O aspecto da aplicación (que constará dun so formulario) será algo parecido ao seguinte:



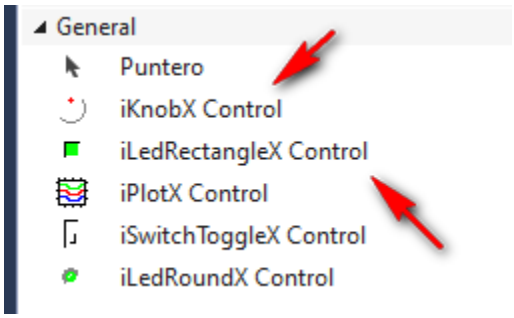
Especificacións:

- O potenciómetro xerará valores comprendidos entre 0 e 255 (1 byte) aínda que a escala do mesmo vaia de 0 a 100.
- Os dous cadros de texto para visualizar o dato decimal e binario será de só lectura. Os valores virán dados polo potenciómetro.
- Existirán catro botóns para crear o documento de Excel, para iniciar e parar o envío de datos e para pechar a conexión co mesmo.
- Na folla de cálculo de Excel visualizaranse tres columnas, unha coa data, a seguinte coa hora e unha terceira co dato xerado polo potenciómetro. O dato visualizarase en formato número enteiro.

- Os datos visualizaranse en sucesivas filas como si dunha táboa se tratara.
- O documento de Excel chamarase **datos.xlsx**
- Dentro da carpeta da aplicación haberá una carpeta chamada Excel onde se atopará o ficheiro creado.
- O envío de datos a Excel farase cada 5 segundos.

Solución:

Empregaremos os compoñentes de iocomp que representan a un led rectangular (**iLedRectangleXControl**) e o potenciómetro (**iKnobXControl**).



Nesta práctica traballamos cunha librería de enlace dinámico (dll) que temos creada previamente (csari) e que contén funcións necesarias para facer conversións entre sistemas de numeración.

Engadimos dita librería e a de iocomp necesaria para traballar cos obxectos mencionados.

```
10  using AxisDigitalLibrary;
11  using csari;
```

Declaramos as variables necesarias para resolver a tarefa.

```
17          MyFunc funciones = new MyFunc();
18          AxiLedRectangleX[] leds = new AxiLedRectangleX[8];//Declara a matriz de leds
19
20          string ruta = Application.StartupPath;
21
22          Microsoft.Office.Interop.Excel.Application xlApp;
23          Microsoft.Office.Interop.Excel.Workbook xlWorkBook;
24          Microsoft.Office.Interop.Excel.Worksheet xlWorkSheet;
25
26          int posicion = 2;
27          bool archivo_creado = false;
```

Na liña 17 creamos un obxecto (funciones) para acceder ás funcións da dll que temos creada.

Na liña 18 declaramos unha matriz de leds na que imos colocar os oito leds para crear código máis simple á hora de resolver o algoritmo.

As liñas da 20 á 27 son necesarias para acceder a un documento de Microsoft Excel para enviarlle datos.

No evento *Load* do formulario enchemos a matriz de leds (os leds noméanse como L0, L1, L2.....L7).

```
34  private void Frm_Principal_Load(object sender, EventArgs e)
35  {
36      for (int i = 0; i < 8; i++)
37      {
38          leds[i] = (AxiLedRectangleX)this.Controls["L" + Convert.ToString(i)];//Inicializa la matriz de leds
39      }
40
41      Txt_Decimal.Text = Convert.ToString(Math.Round(Poten.Position * 2.55));//Iniciar con el valor 0 cargado
42  }
```

Nos eventos *click* dos botóns ON e OFF poñemos en marcha e paramos o temporizador que enviará datos ao documento de Excel.

```
73      private void Btn_ON_Click(object sender, EventArgs e)
74      {
75          Temporizador.Enabled = true;//Activa o temporizador
76      }
77
        1 referencia
78      private void Btn_OFF_Click(object sender, EventArgs e)
79      {
80          Temporizador.Enabled = false;//Desactiva o temporizador
81      }
```

No evento *Click* do botón Crear creamos o documento do Excel e facémolo visible.

```
83  private void Btn_Crear_Click(object sender, EventArgs e)
84  {
85      archivo_creado = true;
86
87      object misValue = System.Reflection.Missing.Value;//Proceso para crear la hoja de cálculo
88
89      xlApp = new Microsoft.Office.Interop.Excel.ApplicationClass();
90      xlWorkBook = xlApp.Workbooks.Add(misValue);
91
92      xlWorkSheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWorkBook.Worksheets.get_Item(1);
93      xlWorkSheet.Cells[1, 1] = "Fecha:";
94      xlWorkSheet.Cells[1, 2] = "Hora:";
95      xlWorkSheet.Cells[1, 3] = "Dato:";
96
97      //En C# poñemos dúas veces \\ para que non interprete como unha secuencia de escape
98      //A seguinte liña emprégase a primeira vez que temos que crear o documento de Excel
99      xlWorkBook.SaveAs(ruta + "\\Excel\\datos.xls", Microsoft.Office.Interop.Excel.XlFileFormat.xlWorkbookNormal,
100         misValue, misValue, misValue, misValue, Microsoft.Office.Interop.Excel.XlSaveAsAccessMode.xlExclusive,
101         misValue, misValue, misValue, misValue, misValue);
102
103     xlApp.Application.Visible = true;//Para ter visible a folla de cálculo mentres se lle envían datos
104 }
```

No evento *Click* do botón Cerrar pechamos a conexión co documento de Excel.

```
106 private void Btn_Cerrar_Click(object sender, EventArgs e)
107 {
108     archivo_creado = false;
109
110     object misValue = System.Reflection.Missing.Value;//Proceso para pechar unha folla de cálculo
111
112     xlWorkBook.Close(true, misValue, misValue);
113     xlApp.Quit();
114     releaseObject(xlWorkSheet);
115     releaseObject(xlWorkBook);
116     releaseObject(xlApp);
117 }
```

O método *releaseObject* temos que crealo, porque non existe por defecto.

```
139         private void releaseObject(object obj) //Método releaseObject usado para pechar a conexión co ficheiro
140         {
141             try
142             {
143                 System.Runtime.InteropServices.Marshal.ReleaseComObject(obj);
144                 obj = null;
145             }
146             catch (Exception ex)
147             {
148                 obj = null;
149                 MessageBox.Show("Ocorreu unha excepción ao liberar o obxecto " + ex.ToString());
150             }
151             finally
152             {
153                 GC.Collect();
154             }
155         }
```

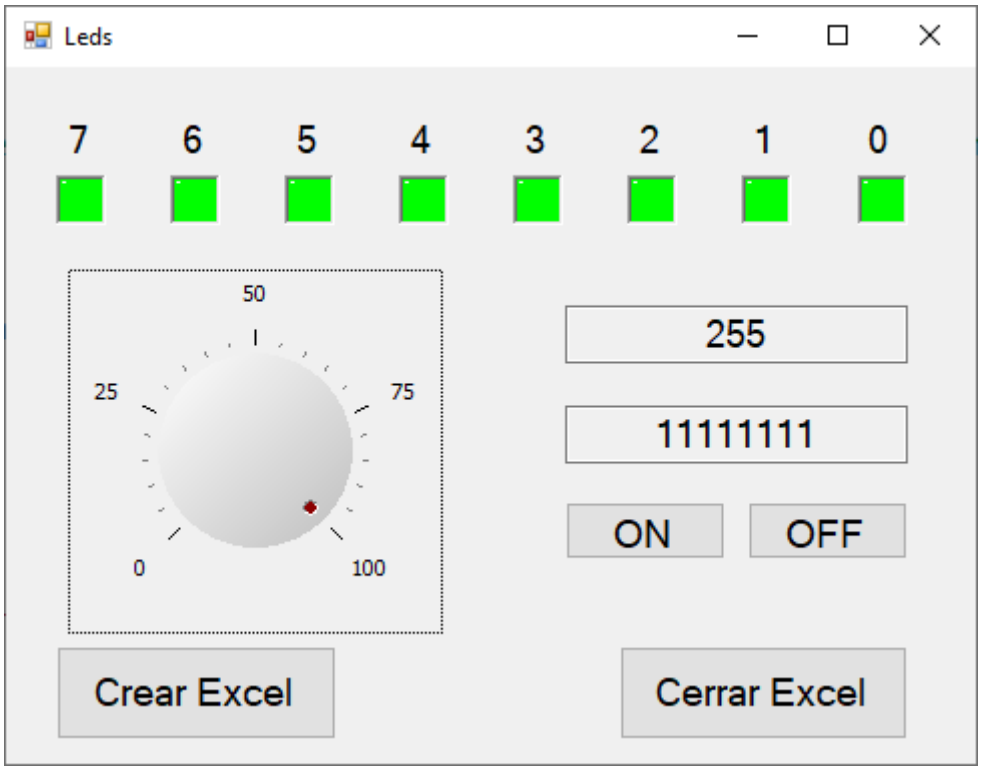
No evento *Tick* do temporizador enviamos o dato ao documento de Excel.

```
119         private void Temporizador_Tick(object sender, EventArgs e)
120         {
121             if (archivo_creado)//Check de que hai excel creado para que non rompa ao pulsar ON se non hai un
122             {
123                 object misValue = System.Reflection.Missing.Value;//Proceso para enviar un dato á folla de cálculo
124 
125                 xlWorkBook = xlApp.Workbooks.Open(ruta + "\\Excel\\datos", 0, false, 5, null, null, false,
126                     Microsoft.Office.Interop.Excel.XlPlatform.xlWindows, null, true, false, 0, true, false, false);
127 
128                 xlWorkSheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWorkBook.Worksheets.get_Item(1);
129                 xlWorkSheet.Cells[posicion, 1] = DateTime.Now.ToString("dd/MM/yyyy"); //Poñer a data
130                 xlWorkSheet.Cells[posicion, 2] = DateTime.Now.ToString("HH:mm:ss"); //Poñer a hora
131                 xlWorkSheet.Cells[posicion, 3] = Txt_Decimal.Text; //Poñer o dato do potenciómetro
132                 xlWorkBook.Save();
133 
134                 posicion++;
135             }
136             else Temporizador.Enabled = false;
137         }
```

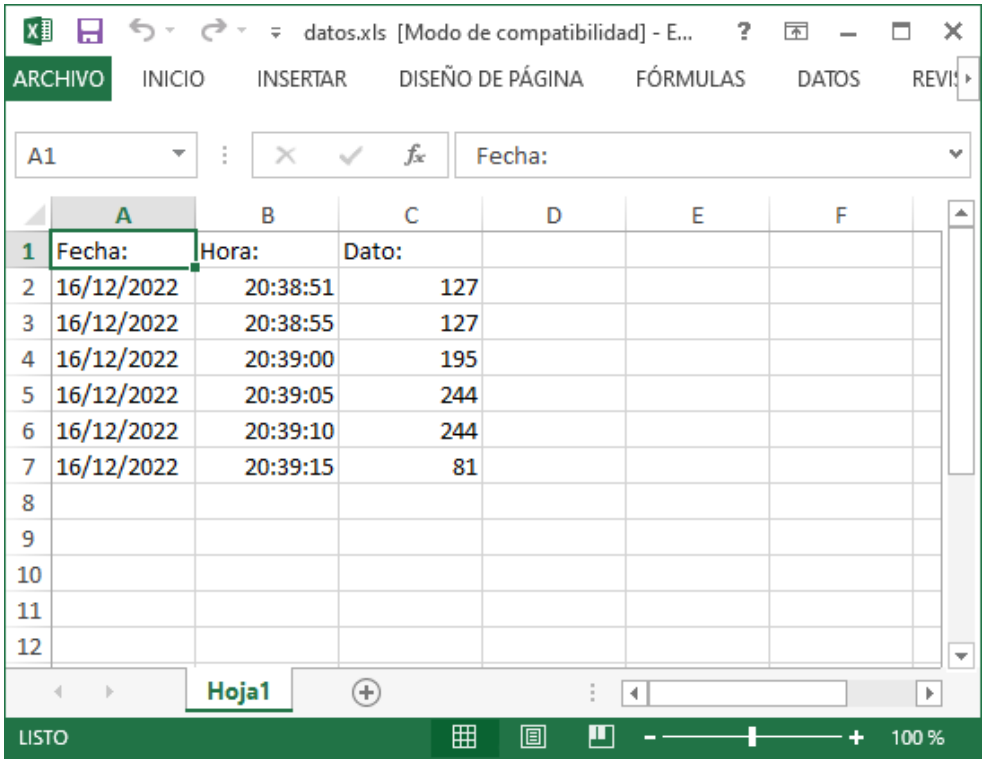
Por último, no evento *OnPositionChange* do potenciómetro collemos o valor do mesmo, convertémolo a binario e visualizamos nos leds dito valor.

```
44          private void Poten_OnPositionChange(object sender, EventArgs e)
45          {
46              Txt_Decimal.Text = Convert.ToString(Math.Round(Poten.Position*2.55));
47 
48              string cadena = "";
49              int longitud = 0;
50 
51              cadena = funciones.DecBin(Convert.ToInt32(Txt_Decimal.Text));
52 
53              longitud = cadena.Length;//Lonxitude do num. binario
54 
55              for (int i = 0; i < 8 - longitud; i++)//Enche con ceros á esquerda ata oito
56              {
57                  cadena = "0" + cadena;
58              }
59 
60              Txt_Binario.Text = cadena;
61 
62              //Visualizamos leds
63              for (int i = 0; i < 8; i++)
64              {
65                  if (cadena[7 - i] == '1')
66                  {
67                      leds[i].Active = true;
68                  }
69                  else leds[i].Active = false;
70              }
71          }
```

So queda comprobar o funcionamento.



E comprobar como se envían os datos ao documento de Excel.



	A	B	C	D	E	F
1	Fecha:	Hora:	Dato:			
2	16/12/2022	20:38:51	127			
3	16/12/2022	20:38:55	127			
4	16/12/2022	20:39:00	195			
5	16/12/2022	20:39:05	244			
6	16/12/2022	20:39:10	244			
7	16/12/2022	20:39:15	81			
8						
9						
10						
11						
12						

Observamos que, neste exemplo estamos a traballar cun documento en formato Excel antiguo (*.xls). Dependendo da versión que teñamos instalada, traballaremos con extensión **.xls** ou **.xlsx**