Manual de usuario 0.1

icaro

Robotica Educativa Libre

Valentin Basel

Manual de usuario 0.1 icaro
Robotica Educativa Libre
Edición 0

Autor Valentin Basel *valentinbasel@valentinbasel.com*

Copyright © 2011 | You need to change the HOLDER entity in the es-ES/icaro.ent file | This material may only be distributed subject to the terms and conditions set forth in the GNU Free Documentation License (GFDL), V1.2 or later (the latest version is presently available at *http://www.gnu.org/licenses/fdl.txt*).

Prefacio **v**
1. Convenciones del Documento v
1.1. Convenciones Tipográficas v
1.2. Convenciones del documento vi
1.3. Notas y Advertencias vii

1. Introduccion **1**
1.1. robotica educativa 1
1.2. lenguaje icaro 1
1.3. ICARO y python. 3
1.4. ICARO y TURTLEART. 5
1.5. Pycaro 7

2. Hardware **9**
2.1. Placa np024-2 9
2.2. PCB placa np024-2 9
2.3. Placa np024-3 11
2.4. Firmware del PIC 13
2.5. Protocolo de comunicacion del PIC 15
2.6. listado de componentes de la placa np024-3 16

3. instrucciones icaro 0.6.1 **17**
3.1. instrucciones de lenguaje icaro 0.6.1 17

4. usando `tortucaro` **19**
4.1. primitivas **`torutcaro`** 19
4.2. ejemplo de uso de **`torutcaro`** 19

A. Revision History **21**

Índice **23**

Prefacio

1. Convenciones del Documento

Este manual utiliza varias convenciones para resaltar algunas palabras y frases y llamar la atención sobre ciertas partes específicas de información.

En ediciones PDF y de papel, este manual utiliza tipos de letra procedentes de *Liberation Fonts*[1]. Liberation Fonts también se utilizan en ediciones de HTML si están instalados en su sistema. Si no, se muestran tipografías alternativas pero equivalentes. Nota: Red Hat Enterprise Linux 5 y siguientes incluyen Liberation Fonts predeterminadas.

1.1. Convenciones Tipográficas

Se utilizan cuatro convenciones tipográficas para llamar la atención sobre palabras o frases específicas. Dichas convenciones y las circunstancias en que se aplican son las siguientes:

`Negrita monoespaciado`

Utilizada para resaltar la entrada del sistema, incluyendo comandos de shell, nombres de archivo y rutas. También se utiliza para resaltar teclas claves y combinaciones de teclas. Por ejemplo:

> Para ver el contenido del archivo **`my_next_bestselling_novel`** en su directorio actual de trabajo, escriba el comando **`cat my_next_bestselling_novel`** en el intérprete de comandos de shell y pulse **`Enter`** para ejecutar el comando.

El ejemplo anterior incluye un nombre de archivo, un comando de shell y una tecla clave. Todo se presenta en negrita-monoespaciado y distinguible gracias al contexto.

Las combinaciones de teclas se pueden distinguir de las teclas claves mediante el guión que conecta cada parte de una combinación de tecla. Por ejemplo:

> Pulse **`Enter`** para ejecutar el comando.
>
> Pulse **`Control`**+**`Alt`**+**`F2`** para cambiar a la primera terminal virtual. Pulse **`Control`**+**`Alt`**+**`F1`** para volver a su sesión de Ventanas-X.

La primera oración resalta la tecla clave determinada que se debe pulsar. La segunda resalta dos conjuntos de tres teclas claves que deben ser presionadas simultáneamente.

Si se discute el código fuente, los nombres de las clase, los métodos, las funciones, los nombres de variables y valores de retorno mencionados dentro de un párrafo serán presentados en **`Negrita-monoespaciado`**. Por ejemplo:

> Las clases de archivo relacionadas incluyen **`filename`** para sistema de archivos, **`file`** para archivos y **`dir`** para directorios. Cada clase tiene su propio conjunto asociado de permisos.

Negrita proporcional

Esta denota palabras o frases encontradas en un sistema, incluyendo nombres de aplicación, texto de cuadro de diálogo, botones etiquetados, etiquetas de cajilla de verificación y botón de radio; títulos de menú y títulos del sub-menú. Por ejemplo:

[1] https://fedorahosted.org/liberation-fonts/

Seleccionar **Sistema** → **Preferencias** → **Ratón** desde la barra del menú principal para lanzar **Preferencias de Ratón**. En la pestaña de **Botones**, haga clic en la cajilla **ratón de mano izquierda** y luego haga clic en **Cerrar** para cambiar el botón principal del ratón de la izquierda a la derecha (adecuando el ratón para la mano izquierda).

Para insertar un caracter especial en un archivo de **gedit**, seleccione desde la barra del menú principal **Aplicaciones** → **Accessories** → **Mapa de caracteres**. Luego, desde la barra de menúes de **mapa de caracteres** elija **Búsqueda** → **Hallar…**, teclee el nombre del caracter en el campo **Búsqueda** y haga clic en **Siguiente**. El caracter buscado se resaltará en la **Tabla de caracteres**. Haga doble clic en este caracter resaltado para colocarlo en el campo de **Texto para copiar** y luego haga clic en el botón de **Copiar**. Ahora regrese a su documento y elija **Editar** → **Pegar** desde la barra de menú de **gedit**.

El texto anterior incluye nombres de aplicación; nombres y elementos del menú de todo el sistema; nombres de menú de aplicaciones específicas y botones y texto hallados dentro de una interfaz gráfica de usuario, todos presentados en negrita proporcional y distinguibles por contexto.

Itálicas-negrita monoespaciado o ***Itálicas-negrita proporcional***

Ya sea negrita monoespaciado o negrita proporcional, la adición de itálicas indica texto reemplazable o variable. Las itálicas denotan texto que usted no escribe literalmente o texto mostrado que cambia dependiendo de la circunstancia. Por ejemplo:

Para conectar a una máquina remota utilizando ssh, teclee **ssh *nombredeusuario@dominio.nombre*** en un intérprete de comandos de shell. Si la máquina remota es **example.com** y su nombre de usuario en esa máquina es john, teclee **ssh john@example.com**.

El comando **mount -o remount *file-system*** remonta el sistema de archivo llamado. Por ejemplo, para volver a montar el sistema de archivo **/home**, el comando es **mount -o remount /home**.

Para ver la versión de un paquete actualmente instalado, utilice el comando **rpm -q *paquete***. Éste entregará el resultado siguiente: ***paquete-versión-lanzamiento***.

Observe las palabras en itálicas y negrita sobre — nombre de usuario, domain.name, sistema de archivo, paquete, versión y lanzamiento. Cada palabra es un marcador de posición, tanto para el texto que usted escriba al ejecutar un comando como para el texto mostrado por el sistema.

Aparte del uso estándar para presentar el título de un trabajo, las itálicas denotan el primer uso de un término nuevo e importante. Por ejemplo:

Publican es un sistema de publicación de *DocBook*.

1.2. Convenciones del documento

Los mensajes de salida de la terminal o fragmentos de código fuente se distinguen visualmente del texto circundante.

Los mensajes de salida enviados a una terminal se muestran en **romano monoespaciado** y se presentan así:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Los listados de código fuente también se muestran en **romano monoespaciado**, pero se presentan y resaltan de la siguiente manera:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

1.3. Notas y Advertencias

Finalmente, utilizamos tres estilos visuales para llamar la atención sobre la información que de otro modo se podría pasar por alto.

Nota

Una nota es una sugerencia, atajo o enfoque alternativo para una tarea determinada. Ignorar una nota no debería tener consecuencias negativas, pero podría perderse de algunos trucos que pueden facilitarle las cosas.

Importante

Los cuadros con el título de importante dan detalles de cosas que se pueden pasar por alto fácilmente: cambios de configuración únicamente aplicables a la sesión actual, o servicios que necesitan reiniciarse antes de que se aplique una actualización. Ignorar estos cuadros no ocasionará pérdida de datos, pero puede causar enfado y frustración.

Advertencia

Las advertencias no deben ignorarse. Ignorarlas muy probablemente ocasionará pérdida de datos.

Introduccion

ICARO es un software desarrollado con la intención de acercar de una manera trasparente al usuario los fundamentos de la programación (iteraciones, repeticiones, recursividad, si condicionales) aplicado a un contexto físico como puede ser un robot o cualquier sistema de automatización.

1.1. robotica educativa

Uno de los principales objetivos de la Robótica Educativa, es la generación de entornos de aprendizaje basados fundamentalmente en la práctica real de los estudiantes.

La idea es que los niños puedan concebir, desarrollar y poner en práctica diferentes robots educativos que les permitirán resolver algunos problemas y facilitarán, al mismo tiempo, ciertos aprendizajes.

La Robótica Educativa, dado su carácter polivalente y multidisciplinario, constituye una actividad que permite el abordaje de conocimientos complejos en torno a las nuevas tecnologías a partir de aportes provenientes de la electrónica, física, matemática e informática.

En particular, la enseñanza de las bases fundamentales de la lógica de los sistemas computacionales es posible trabajarla a través de la creación de un mecanismo autónomo electrónico (robótico).

Por otro lado, la posibilidad de desarrollar una serie de juguetes-objetos implica una experiencia que contribuye a expandir la creatividad y el pensamiento reflexivo y científico de los alumnos (en relación la formulación de hipótesis, la experimentación, la elaboración de conclusiones).

En otras palabras, se trata de crear las condiciones para la apropiación de conocimientos por parte de los niños y de permitir su transferencia en diferentes campos de problemas.

Se trata de otorgar a los alumnos un rol activo en sus aprendizajes, colocándoles como diseñadores de sus propios proyectos y constructores de conocimientos.

Nota

“Al enseñarle a pensar al ordenador, los chicos se embarcan en una exploración del modo en que ellos mismos piensan” Seymour Papert

1.2. lenguaje icaro

ICARO es un software desarrollado con la intención de acercar de una manera trasparente al usuario los fundamentos de la programación (iteraciones, repeticiones, recursividad, si condicionales) aplicado a un contexto físico como puede ser un robot o cualquier sistema de automatización.

Alumnos del colegio de oficios san cayetano soldando sus propias placas icaro np024-2.

Las placas icaro np024-2 esta diseñada para ser facil de armar y fabricar con minimos concocimientos.

Su funcionamiento se basa en escribir instrucciones en un archivo de texto plano para que el software (que funciona como un intérprete) vaya leyendo línea por línea y enviando la información traducida al puerto paralelo o puerto serie, y mediante un hardware de protección eléctrico, poder encender/apagar distintos mecanismos electrónicos como motores, motores paso a paso, solenoides, servos, etc. etc.

Nota.

ICARO trata de simplificar el componente técnico de la robótica para que el alumno pueda concentrarse en su desarrollo cognitivo

ICARO puede trabajar con dos tipos de placas de control.

Una placa de puerto paralelo sencilla y de facil fabricación y una placa de puerto serie que mediante un adaptador USB/serie puede trabajar con netbooks.

1.3. ICARO y python.

ICARO se integra con python a traves de una API (apicaro.py) que permite manejar el protocolo de comunicacion de la placa de puerto serie np024-3.

apicaro.py permite implementar una clase para manejar la placa np024-3 de mediante lenguaje python, simplemente importando el modulo y instanciando la clase **`icaro`**.

```
#!/usr/bin/env python
import apicaro
import time
icaro = apicaro.puerto()
icaro.PUERTO='/dev/ttyUSB0'
icaro.iniciar()
a=0
for a in range(255):
    icaro.activar(a)
    time.sleep(0.2)
```

Por defecto apicaro configura el puerto serie usando 9 variables:

```
    PUERTO='/dev/ttyUSB0'
    BAUDIOS=9600
    BYTESIZE=8
    PARITY='N'
    STOPBIT=1
    TIMEOUT=1
    XONXOFF=False
    RTSCTS=False
    DSRDTR=False
```

Excepto **`PUERTO`**, las demas variables se pueden dejar como estan porque son las configuraciones estandar para le puerto serie **`/dev/ttyUSB0`**

Nota.

apicaro.py solo funciona con el puerto serie.

Todavia no esta implementada para trabajar con el puerto paralelo

PUERTO configura en que direccion esta conectada la placa NP024-3, y permite manejar mas de una placa NP024-3.

```
#!/usr/bin/env python

import apicaro
import time
icaro = apicaro.puerto()
icaro.PUERTO='/dev/ttyUSB0'
```

De esta forma queda instanciada la clase **puerto()** dentro del modulo apicaro, y configurado el valor de la variable **PUERTO** para la direccion **/dev/ttyUSB0**.

Luego de configurar e instanciar a la clase **puerto()** se pueden usar sus 5 funciones para iniciar, activar, leer y cerrar el puerto.

Las funciones que implementa la clase **puerto()** de **apicaro** son:

- **iniciar()**
- **cerrar()**
- **activar(self,valor)**
- **leer(self,sensor)**
- **activar_servo(self,valor,servo)**

iniciar()

abre el puerto con la configuracion definida en la variables de la clase. Devuelve 0 si fue correctamente abierto el puerto.

activar(self,valor)

Manda la variable **valor** al puerto serie, donde la placa de multiplexara el byte en 8 bits (los 8 pines de salida del puerto).

valor solo puede ser una variable entero (de 0 a 256). Devuelve **True** si abrio el puerto exitosamente o **False** en caso contrario.

leer(self,sensor)

toma el valor de **sensor** y lee el respectivo pin de la placa np024-3. Solo se pueden enviar valores del 1 al 7, la funcion devuelve 0 o el valor de **sensor**

cerrar()

cierra el puerto definido en la variable global **PUERTO**.

`activar_servo(self,valor,servo)`

Activa el envio de **PWM**.

La variable **`valor`** representa un entero de 0 a 255, para regular el pulso de **PWM**, mientras que **`servo`** es una variable que representa los pines de salida donde esta conectado el servomotor (de 1 a 8).

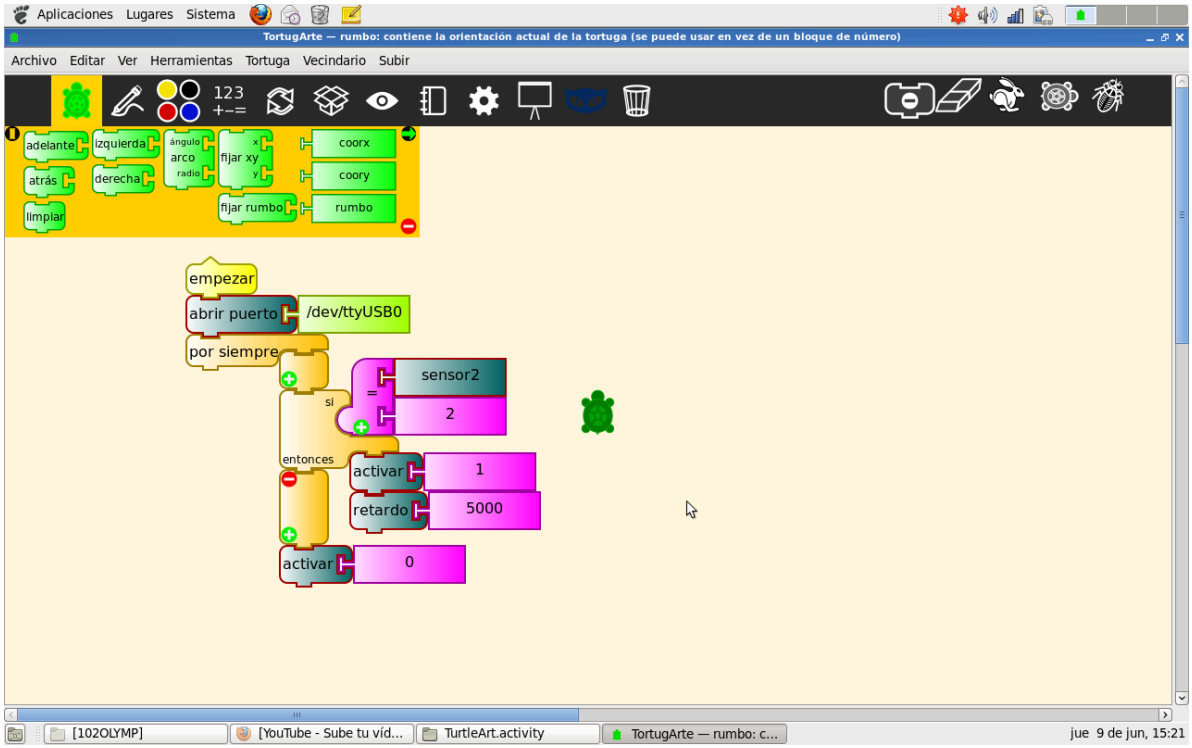
Nota

La version del firmware pic-caro 04 es la unica que implementa manejo de servomotores y solo puede controlar un servo a la ves.

Es importante dejar por lo menos 1500 Ms de espera para mover un servomotor, porque si no se puede "colgar" la placa.

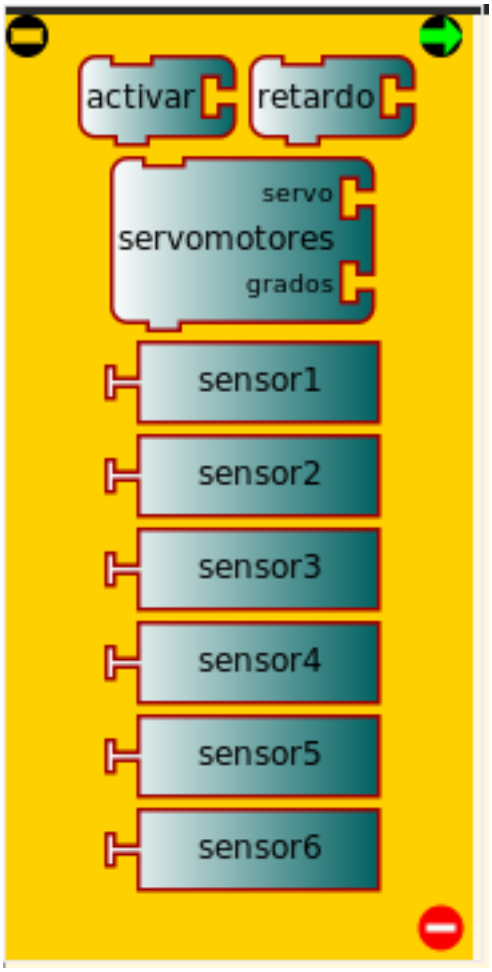
1.4. ICARO y TURTLEART.

Para facilitar el trabajo de los alumnos que usan las netbooks del proyecto OLPC, ICARO se integro la actividad TURTLEART como un plugin que permite agregar nuevas "primitivas" a turtleart para controlar la placa np024-3.



Pantalla de turtleart con primitivas icaro.

Un alumno probando icaro con una netbook classmate y turtleart



primitivas **tortucaro**

La ventaja de usar turtleart, es lo intuitivo que se vuelve programar la logica del robot y la cantidad de documentacion y actividades pedagogicas para el programa.

tortucaro (**turtlart** + **icaro**) funciona como un plugin, permite activar los pines de la placa np024-3 y leer sus 6 sensores individualmente. Para poder hacer iteraciones, repeticiones y calculos matematicos, se usan las primitivas de **turtleart** directamente.

Nota.

El plugin **tortucaro** funciona solamente a partir de la version 107 de **turtleart** y debe estar instalando **apicaro.0.1.py** en el sistema

1.5. Pycaro

pycaro es un software desarrollado en python que permite controlar directamente la placa np024-3 de forma grafica.

La ventaja de pycaro es que se puede usar directamente el hardware sin necesidad de programar nada y se puede usar como un pantallazo inicial de la forma de conexionado de las palcas np024-3 y np024-2.

Nota.

pycaro puede ser una buena ayuda a la hora de explicar la relacion entre los pines y el sistema binario de numeracion.

Hardware

2.1. Placa np024-2

El esquema de conexion de la placa np024-2 esta diseñado para poder trabajar de forma directa con los bits del puerto paralelo.

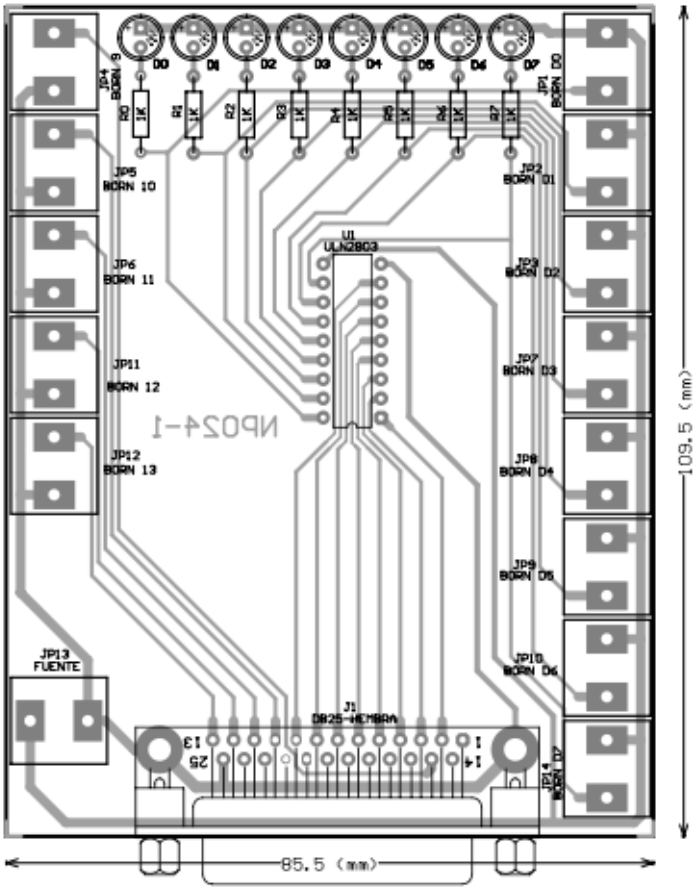
Este tipo de conexiones, es la forma mas sencilla y barata de trabajar con robotica, si bien el puerto paralelo esta en extinsion en la mayorias de las pcs, todavia es factible encontrarlo en computadoras antiguas.

El puerto paralelo de una PC es ideal para ser usado como herramienta de control de motores, relés, LED's, etc. El mismo posee un bus de datos de 8 bits (Pin 2 a 9) y muchas señales de control, algunas de salida y otras de entrada que también pueden ser usadas fácilmente.

Las PC's generalmente poseen solo uno de estos puertos (LPT1) pero con muy poco dinero se le puede adicionar una tarjeta con un segundo puerto paralelo (LPT2).

2.2. PCB placa np024-2

El PCB propuesto se basa en el integrado UNL2803, que es solamente un array de transistores Darlington y sirve para elevar la tension de alimentacion que entrega el puerto paralelo y poder controlar distintos dispositivos electronicos como motores, solenoides etc.



plano PCB de la placa np024-2 para puerto paralelo.

El puerto paralelo es una interfaz entre una computadora y un periférico cuya principal característica es que los bits de datos viajan juntos enviando un byte completo o más a la vez. Es decir, se

implementa un cable o una vía física para cada bit de datos formando un bus. El puerto paralelo de una PC es ideal para ser usado como herramienta de control de motores, relés, LED's, etc. El mismo posee un bus de datos de 8 bits (Pin 2 a 9) y muchas señales de control, algunas de salida y otras de entrada que también pueden ser usadas fácilmente.

Este tipo de placas es de las mas comunes y faciles de armar, hay mucha informacion en internet e ICARO deberia ser comatible con la mayoria de las soluciones propuestas para este puerto.

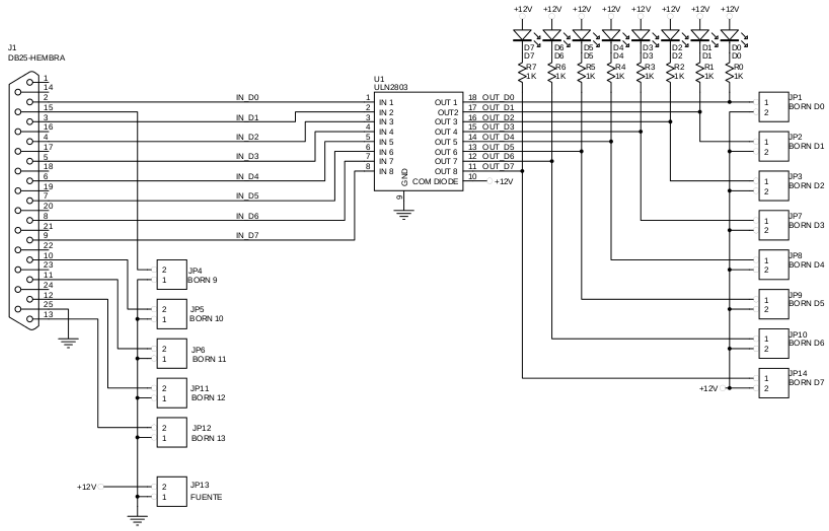


Grafico esquematico de la placa np024-2

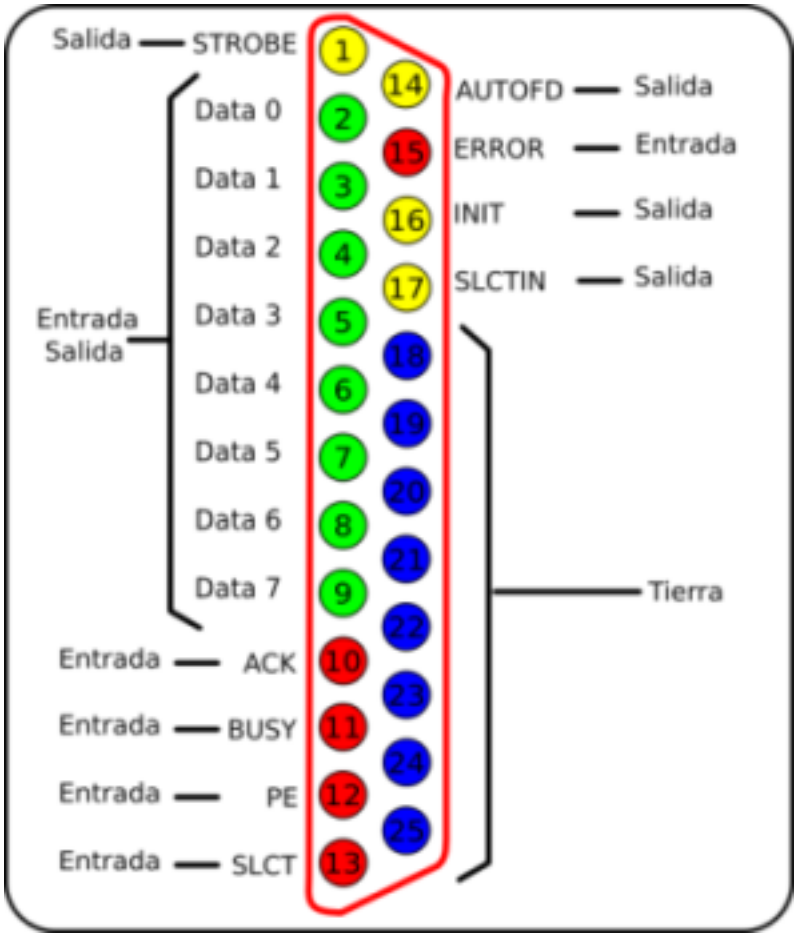


Diagrama de pines del conector DB25 del puerto paralelo.

Advertencia.

Nunca superar la tensión máxima de 12 V – 500 mA porque se puede quemar el integrado UNL2803.

2.3. Placa np024-3

La placa de conexión NP03 sirve de interface entre la computadora y los periféricos (sensores, actuadores) que queramos controlar con el lenguaje ICARO.

Se conecta a la computadora mediante un cable Serie o un adaptador Serie/USB y puede manejar hasta 8 dispositivos digitales (ON/OFF) simultáneos con una tension maxima de 12 V - 500 mA y la minima de 5 V - 500 mA

Para tener en cuenta.

Nunca superar la tensión máxima de 12 V – 500 mA porque se puede quemar el integrado UNL2803. Tampoco trabajar a menos de 5V porque el PIC 16f628A no funcionara adecuadamente.

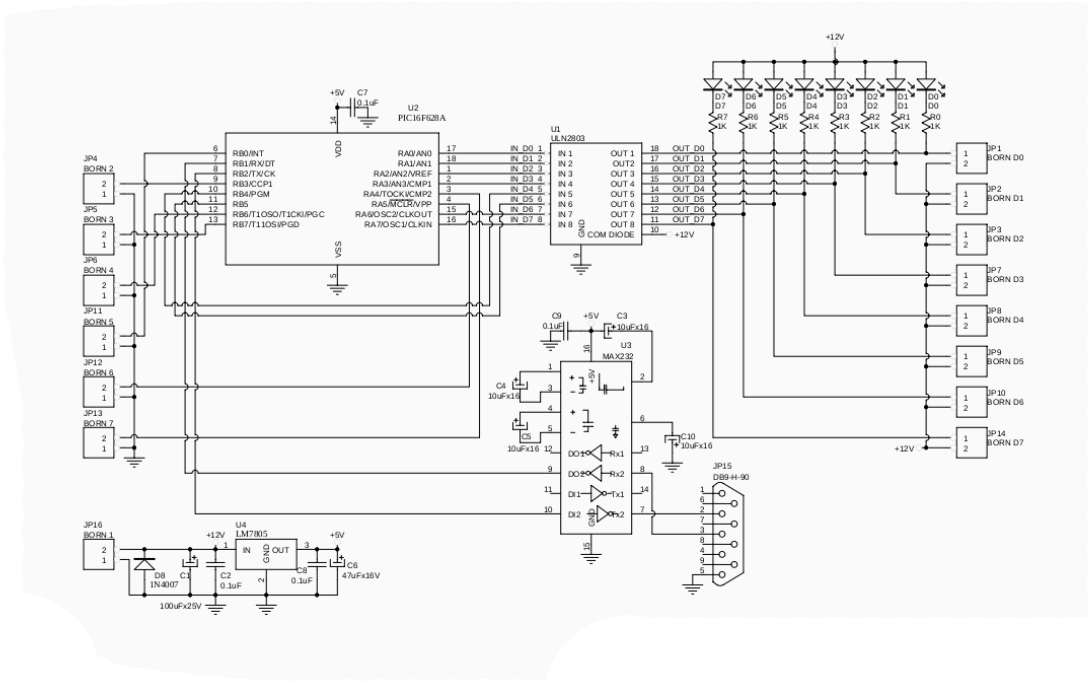
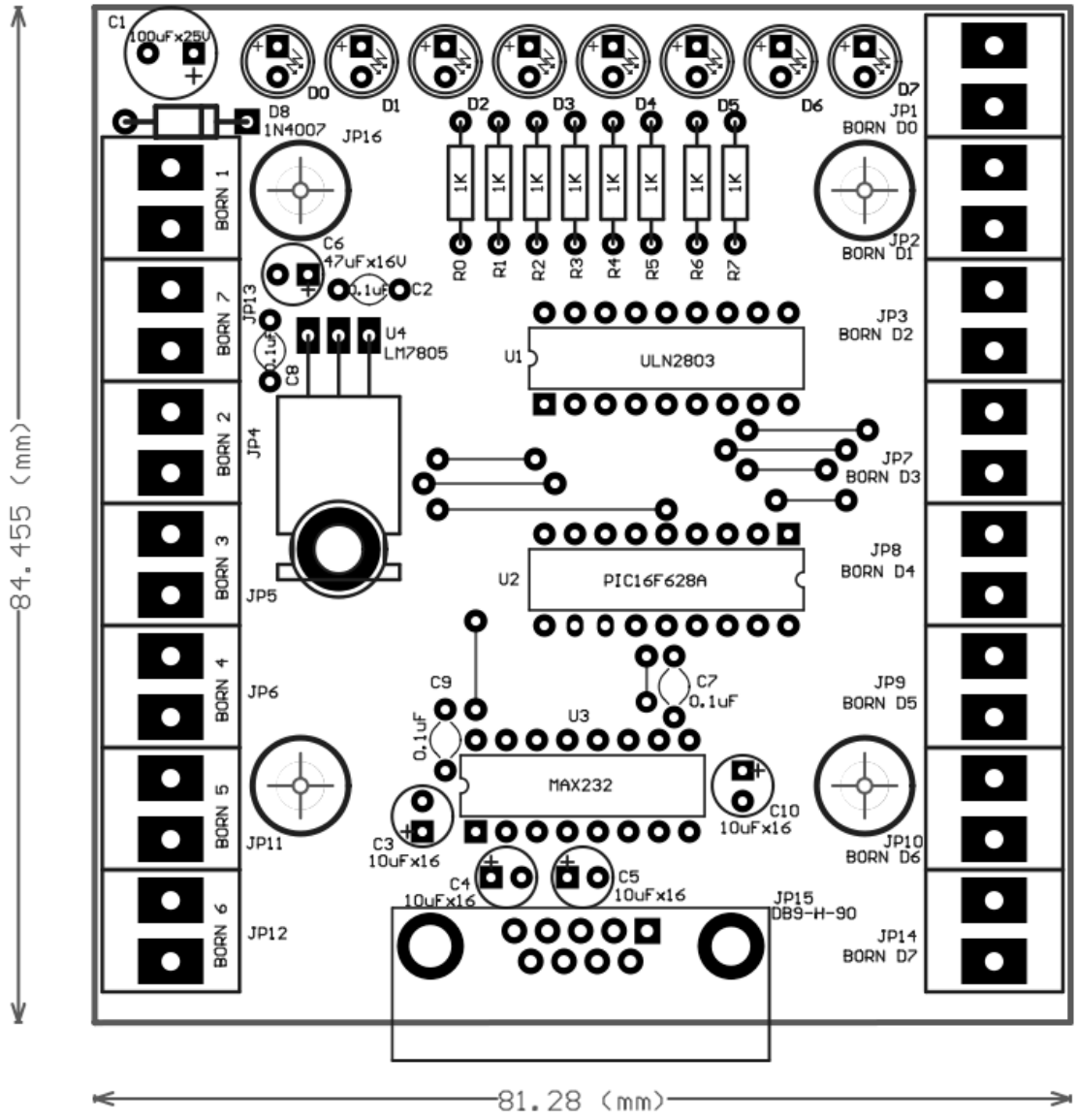


diagrama de la placa np024-3



PCB de la placa np024-3

2.4. Firmware del PIC

El firmware es el software que se cargar en el integrado pic 16f628A para que este pueda trabajar con el protocolo de transferencia para rs232. La versión mas actual del firmware a la fecha es la 0.2.1. Para actualizar o recuperar el firmware (en caso de falla o remplazo del PIC) descargar del siguiente enlace: *firmware*[1]

Ahi se encuentran los archivos ASM (codigo fuente) y los hex (binarios) del proyecto icaro

[1] http://rpmdev.proyectofedora.org/projects/icaro/wiki/Puerto_serie

Nota

para poder actualizar el firmware es necesario sacar el integrado 16f628A y ponerlo en un circuito programador de pics. Sin ese circuito **NO** se puede actualizar el firmware y es recomendable no sacar el PIC.

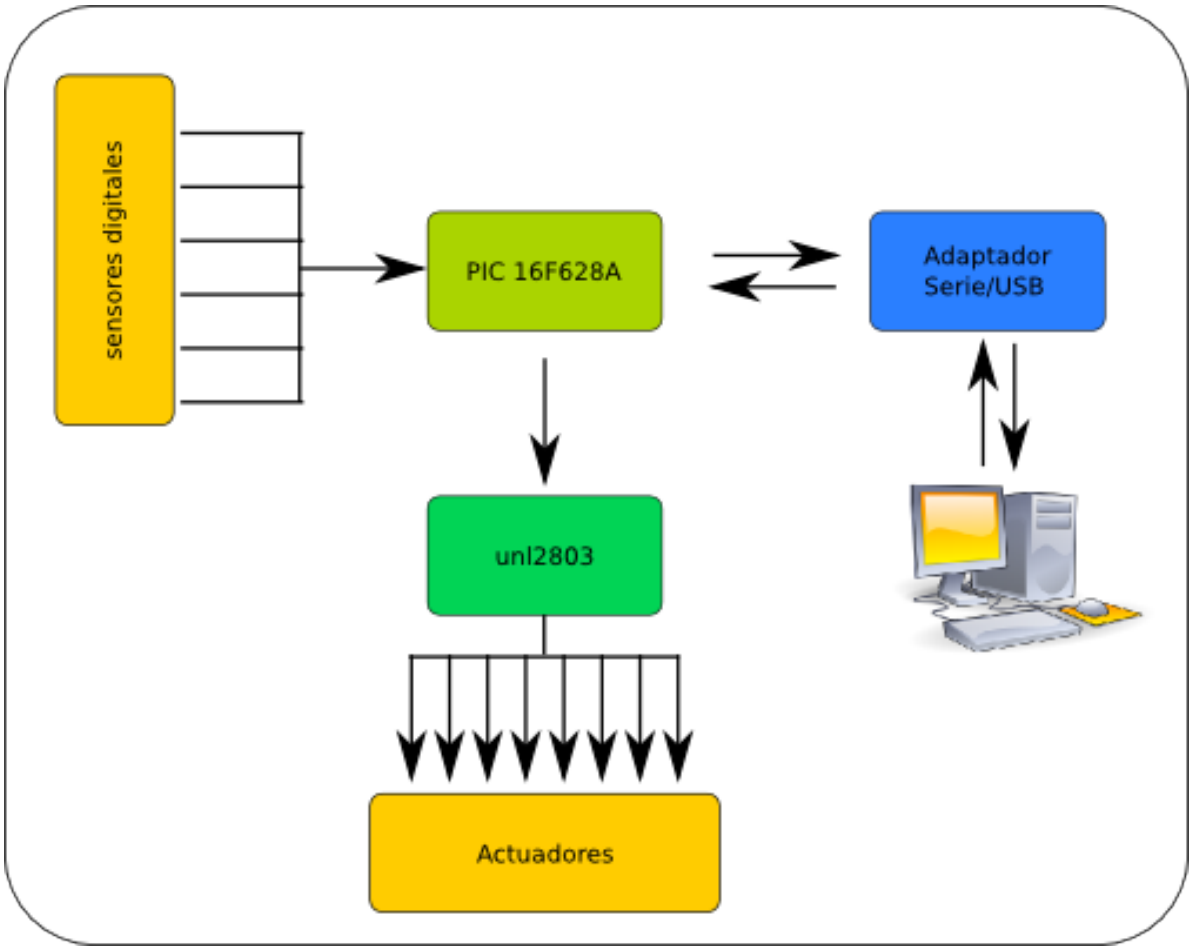
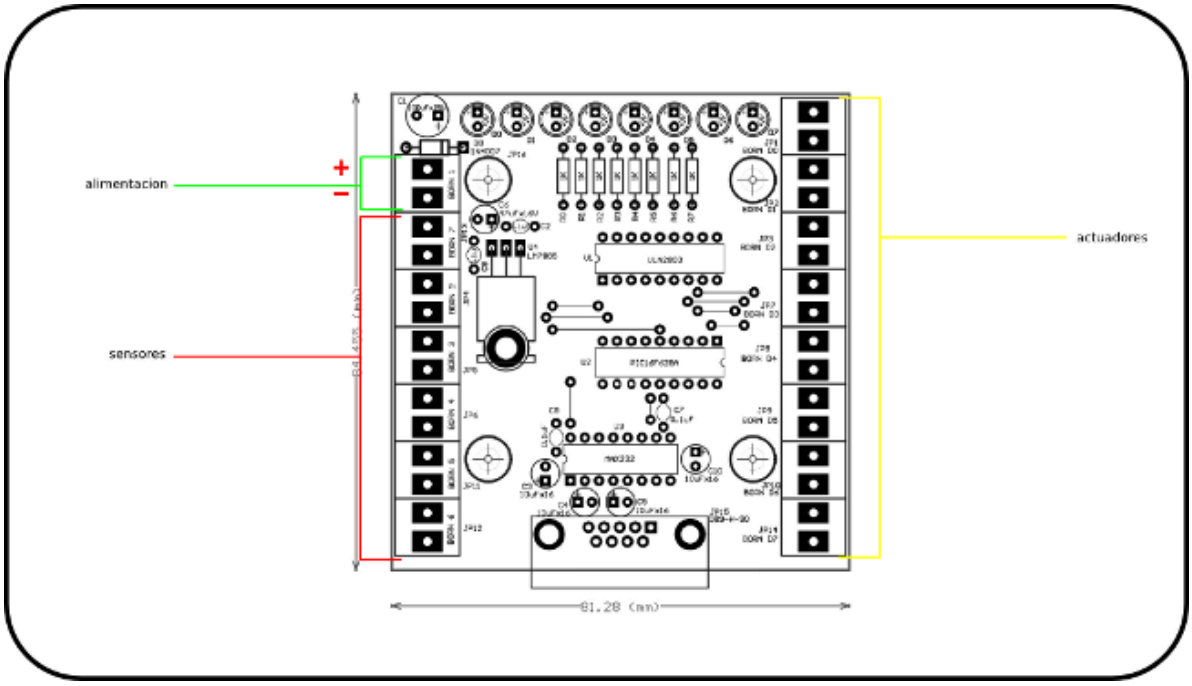


Diagrama de flujo de la placa np024-3



esquema de conexiones de la placa np024-3.

2.5. Protocolo de comunicacion del PIC

El protocolo de transferencia que usa la placa NP03 icaro se basa en enviar caracteres **`ascii`** a través del puerto serial de la computadora.

La placa NP03 nunca inicia una transferencia, solo espera y responde a las ordenes de la pc.

Los tipos de caracteres de inicio del protocolo son :

Tabla 2.1. Protocolo de comunicacion

Caracteres	Respuesta	Descripcion
b	*`icaro05`*	Es un pequeño mensaje de bienvenida con la versión del firmware
s	Espera un segundo valor para enviar al puerto	Después de enviar el código “s” la placa espera hasta recibir otro caracter y enviarlo a los puertos de salida del sistema
e	Espera un segundo valor (1 - 6) y Devuelve el valor de los sensores 1,2,3,4,5,6 (0 o el numero del sensor)	Después de enviar el código “e” la placa espera hasta recibir otro caracter (1,2,3,4,5,6) y devuelve el valor de pulsado del sensor
m	Espera un segundo valor y genera un pulso **`PWM`** para mover el servo	en la version 04 de icaro, solo se puede controlar un servo motor a la ves y dejando 1500 Ms de tiempo entre datos

2.6. listado de componentes de la placa np024-3

Tabla 2.2. Componentes discretos

Cantidad	Tipo
8	resistencias 1K
8	leds
4	cap ceramicos 0.1 uF 4
4	cap electroliticos 10 uFx16
1	cap electrolitico 100 uFx16
1	cap electrolitico 47 uFx16
1	diodo 1n4007

Tabla 2.3. Integrados

Cantidad	Tipo
1	LM7805
1	UNL2803
1	MAX232
1	PIC 16F628A

Tabla 2.4. Otros

Cantidad	Tipo
15	Borneras
1	Conector DB9 hembra
1	Zócalo 2x8 (para el max232)
2	Zócalo 2x9 (UNL2803 y PIC)

instrucciones icaro 0.6.1

3.1. instrucciones de lenguaje icaro 0.6.1

Aca va el texto dewl set de instrucciones

usando `tortucaro`

4.1. primitivas `torutcaro`

Aca va el texto de las primitivas

4.2. ejemplo de uso de `torutcaro`

Aca va el texto con ejemplos de uso de turtleart + icaro

Apéndice A. Revision History

Revisión 0-0 **Mon Jun 13 2011** **Valentin Basel** ***valentinbasel@gmail.com***

primera edicion del manual de usuario ICARO

Índice