Hecho por Víctor Orozco (tuxtor@shekalug.org)

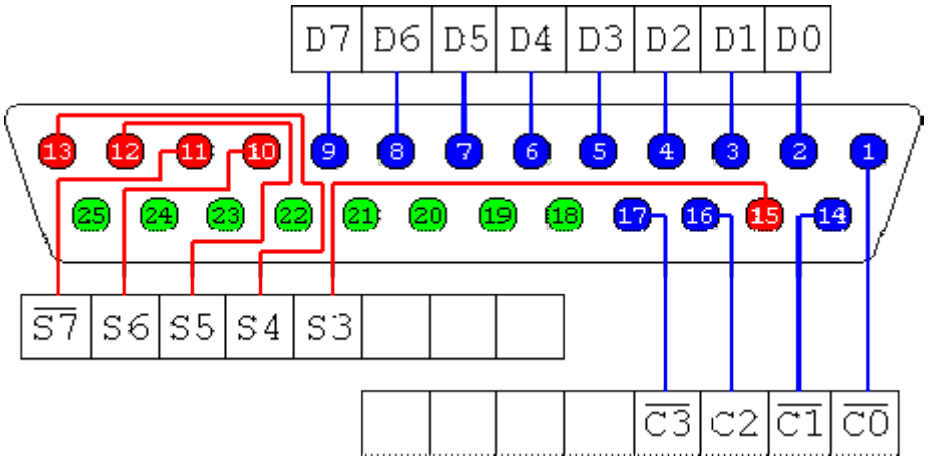
Puerto paralelo

Un puerto paralelo es una interfaz entre un ordenador 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 cable paralelo es el conector físico entre el puerto paralelo y el periférico. En un puerto paralelo habrá una serie de bits de control en vias aparte que irán en ambos sentidos por caminos distintos.

En contraposición al puerto paralelo está el Puerto serie, que envía los datos bit a bit por el mismo hilo.

Descripción del puerto



El puerto paralelo esta compuesto por:

- # 8 Pines de Salida [D0 hasta D7]
- # 5 Pines de Status [S4 hasta S7 y S3]
- # 4 Pines de Control [C0 hasta C3]
- # 8 Pines de Tierra [18 hasta 25]

En este orden de ideas los Pines que tienen una línea superior en su nombre son pines inversores, osea en pocas palabras ustedes indican encendido el pin niega la acción y queda con el estado contrario.

Tutorial de instalacion javax.comm api (Linux)

El API Java Communications (COMM) es un paquete opcional para la plataforma Java 2. Proporciona soporte para comunicación con dispositivos periféricos a través de los puertos serie y paralelo de un ordenador. Es un API especial en el sentido de que aunque está bien definido multi-plataforma, debes descargar una versión específica de las librerías COMM para utilizarlo realmente.

El API COMM no incluye soporte para comunicación sobre puertos Universal Serial Bus (USB) . El soporte para los puertos USB se proporcionara en un API separado que está ahora mismo bajo revisión pública a través del Java Community Process (JCP).

Para empezar, descarga y descomprime el API Java Communications.

Actualmente solo se encuentra disponible para plataformas linux y solaris, ya que se descontinuo su uso en windows por problemas en compatibilidad.

Alternativamente a esta api existe otra posible implementación rxtxSerial (para más información es posible visitar www.rxtx.org) .

Proceso de instalación

1. Paso 1: Descargar la api desde el sitio de java developer network

   http://java.sun.com/products/javacomm/

2. Paso 2: Instalar el soporte nativo para el sistema operativo

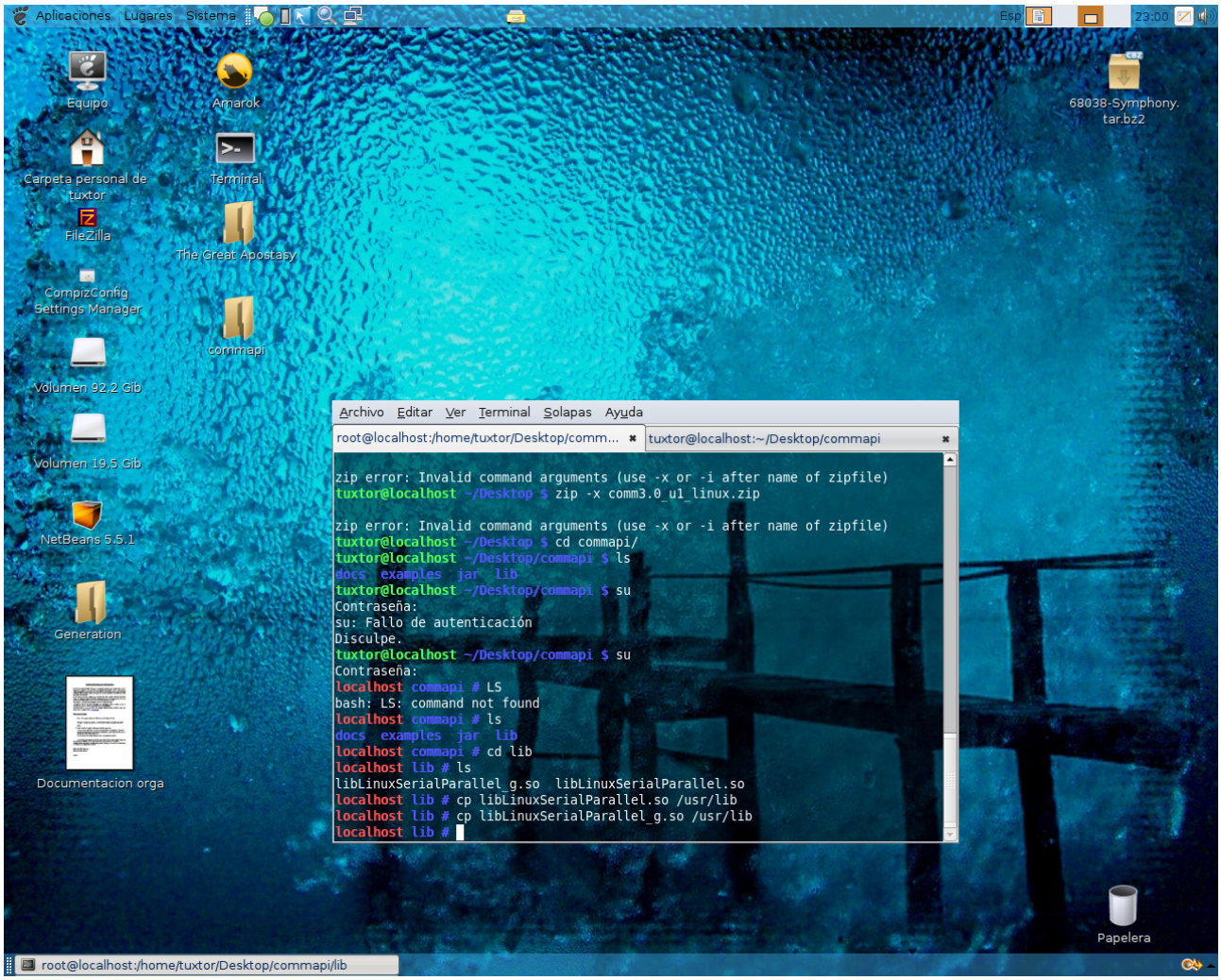
   Aunque nuestra aplicación en teoría sea independiente de plataforma, tendremos que instalar soporte para nuestro sistema operativo y cada versión del API a usar es distinta para cada sistema operativo. Ademas de esto la nueva version de la API ya no esta disponible para window$.

Ya que hemos descompreso el archivo zip en donde viene el api en primer lugar hay que instalar las bibliotecas en el sistema operativo, generalmente en cualquier distribución linux (probado en distibuciones gentoo, redhat), se encuentra en el directorio /usr/lib, asi que copiamos los archivos, todo esto con derechos de superusuario (su o sudo dependiendo de la distribución.

libLinuxSerialParallel_g.so

libLinuxSerialParallel.so

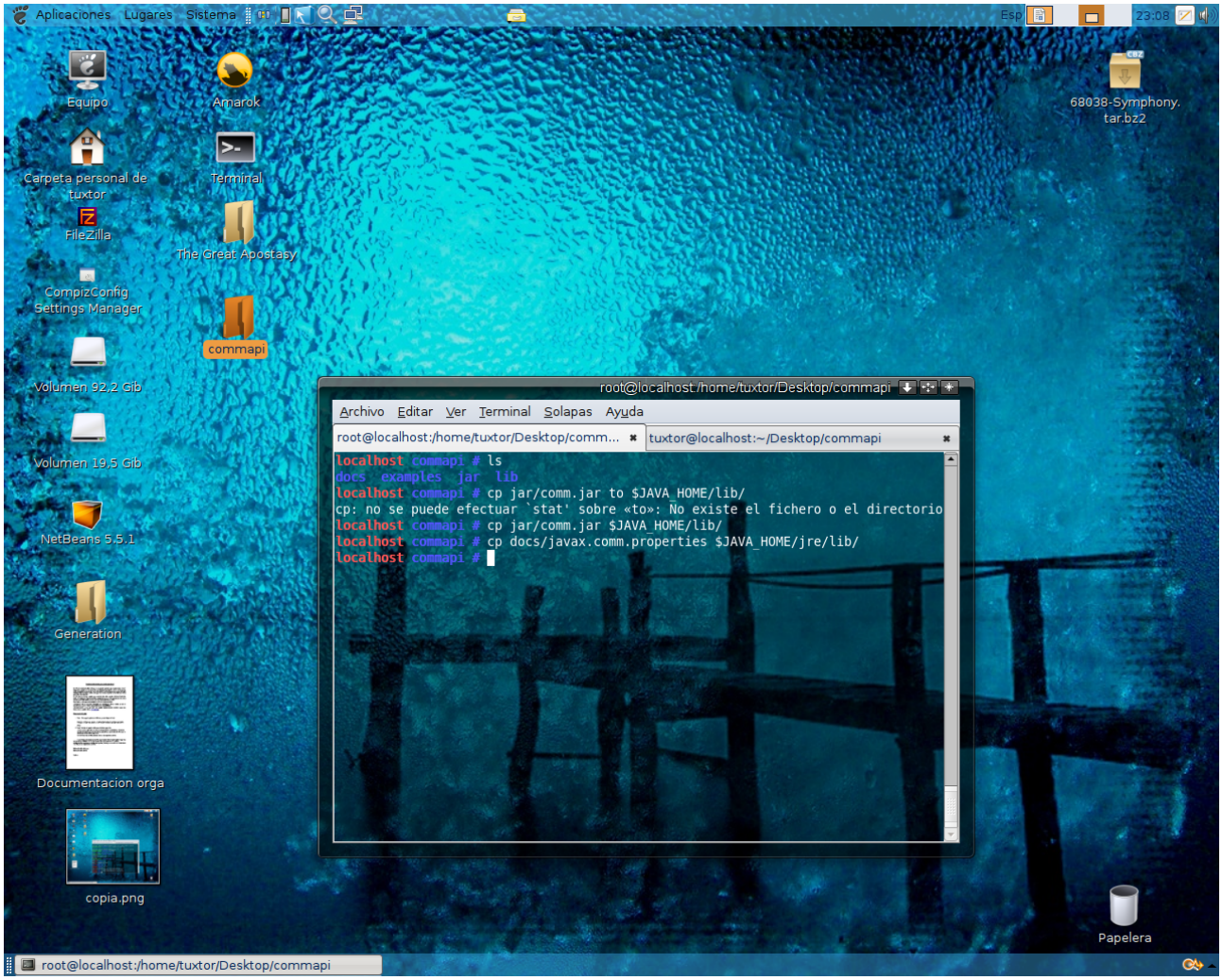
Captura:


Aplicaciones Lugares Sistema
Esp
23:00
Equipo
Amarok
68038-Symphony.tar.bz2
Carpeta personal de tuxtor
Terminal
FileZilla
The Great Apostasy
CompizConfig Settings Manager
commapi
Volumen 92,2 Gib
Volumen 19,5 Gib
NetBeans 5.5.1
Generation
Documentacion orga
Papelera
Archivo Editar Ver Terminal Solapas Ayuda
root@localhost:/home/tuxtor/Desktop/comm...
tuxtor@localhost:~/Desktop/commapi
zip error: Invalid command arguments (use -x or -i after name of zipfile)
tuxtor@localhost ~/Desktop $ zip -x comm3.0_ul_linux.zip
zip error: Invalid command arguments (use -x or -i after name of zipfile)
tuxtor@localhost ~/Desktop $ cd commapi/
tuxtor@localhost ~/Desktop/commapi $ ls
docs examples jar lib
tuxtor@localhost ~/Desktop/commapi $ su
Contraseña:
su: Fallo de autenticación
Disculpe.
tuxtor@localhost ~/Desktop/commapi $ su
Contraseña:
localhost commapi # LS
bash: LS: command not found
localhost commapi # ls
docs examples jar lib
localhost commapi # cd lib
localhost lib # ls
libLinuxSerialParallel_g.so libLinuxSerialParallel.so
localhost lib # cp libLinuxSerialParallel.so /usr/lib
localhost lib # cp libLinuxSerialParallel_g.so /usr/lib
localhost lib #
root@localhost:/home/tuxtor/Desktop/commapi/lib

3. Paso 3: Copia del api y sus propiedades, las propiedades establecen los lineamientos para el driver..

```
cp docs/javax.comm.properties $JAVA_HOME/jre/lib/
cp jar/comm.jar to $JAVA_HOME/lib/
```

Donde $JAVA_HOME es una variable de entorno no necesaria pero que representa, el path completo hacia nuestro directorio de instalación de nuestro JDK.



Hasta aquí ya esta instalada la API en nuestro sistema.

Codigo de deteccion de puertos

Ya que todo esta correctamente instalado, solo nos queda ejecutar un código de prueba en el cual detectaremos los puertos de la pc actual.

package orga;

```
/**
 *
 * @author tuxtor
 * Bajo licencia GPL v3
 */
import javax.comm.*;
import java.util.Enumeration;

public class Main {
  public static void main(String args[]) {
    Enumeration ports =
      CommPortIdentifier.getPortIdentifiers();//Metodo para identificar los
puertos y lo almacena
    //objeto ports de tipo Enumeration
    while (ports.hasMoreElements()) {
      CommPortIdentifier port =
        (CommPortIdentifier)ports.nextElement();
      String type;
      switch (port.getPortType()) {
        case CommPortIdentifier.PORT_PARALLEL:
          type = "Paralelo"; //Se ejecuta si el puerto es paralelo
          break;
        case CommPortIdentifier.PORT_SERIAL:
          type = "Serial";  //Se ejecuta si el puerto es serial
          break;
        default:
          type = "Desconocido/Error"; //No deberia de suceder o el puerto esta
dañado
          break;
      }
      System.out.println("Nombre del puerto"+port.getName() + ": " + type);
    }
  }
}
```

Y si el programa corre con éxito ya podemos empezar el desarrollo de nuestro programa.