ACCES USB DEVICES AND LINUX

This document is intended to help the user set up and use ACCES USB devices in Linux. It is divided into two sections plus an addendum. The first covers setting up Linux so that the device is available for use. The second section covers using the available aiousb.c and aiousb.h files to utilize the device. The addendum addresses security issues associated with using the aiousb library.

SECTION 1: Uploading the firmware.

In order to upload the firmware you must have the *fxload* package installed on your system. To test if you have fxload installed on your system simply type 'fxload -V' at the command line. If fxload is installed on your system you will see version information displayed on your screen. If fxload is not installed on your system you can find more information at http://linux-hotplug.sourceforge.net/ or check the package manager for your Linux distribution.

There are two methods for uploading the firmware to an ACCES USB device. The first involves manually uploading the firmware every time the device is plugged into the computer. When the device is plugged in you need to run the accesloader.pl script with root privileges. This will result in something similar to the following:

```
This script will upload the firmware to any raw ACCES USB
devices that are found on the system If any devices are
programmed the script will pause for 5 seconds before
attempting to make all ACCES USB devices on the system
usable by users other than root. This script should be run
with root privelages
fxload -t fx2 -D /proc/bus/usb/001/009 -I USB-DIO-32.hex
chmod 0666  /proc/bus/usb/001/010
```

In this example 1 USB-DIO-32 had its firmware uploaded.

It is unlikely that this manual method would be used for anything other than simple testing purposes.

The second method involves setting up your system to automatically upload the firmware anytime an ACCES USB device is attached to the system. This method takes only a few minutes to set up and also requires root access to the system.

Step 1: Copy all the .hex files to /usr/share/usb/

Step 2: Copy accesloader.sh to /usr/share/usb/

Once these steps are completed running ls -l on /usr/share/usb should result in something similar to the following:

```
total 64
-rw-r--r-- 1 root root  4026 2004-08-13 11:09 a3load.hex[1]
-rwx------ 1 root root   585 2006-12-14 05:58 accesloader.sh
-rw-r--r-- 1 root root 11609 2006-12-13 00:59 USB-CTR-15.hex
-rw-r--r-- 1 root root  2119 2006-12-13 00:59 USB-DA128-ArevA.hex
-rw-r--r-- 1 root root 12708 2006-12-13 00:59 USB-DIO-32.hex
-rw-r--r-- 1 root root 11216 2006-12-13 00:59 USB-IIRO-16.hex
-rw-r--r-- 1 root root 11233 2006-12-13 00:59 USB-IIRO4-2SM.hex
```

Step 3: copy acces_usb.rules to /etc/udev/rules.d/

Running ls -l on /etc/udev/rules.d/ should result in something similar to the following:

```
total 144
-rw-r--r-- 1 root root   262 2006-05-22 07:25 00-init.rules
-rw-r--r-- 1 root root  2264 2006-05-22 07:25 20-names.rules
-rw-r--r-- 1 root root   190 2006-05-22 07:25 25-iftab.rules
-rw-r--r-- 1 root root  3048 2006-05-22 07:25 40-permissions.rules
-rw-r--r-- 1 root root 47992 2006-05-30 17:55 45-libgphoto2.rules
-rw-r--r-- 1 root root 28262 2006-04-06 00:12 45-libsane.rules
-rw-r--r-- 1 root root  1306 2006-05-22 07:25 60-symlinks.rules
-rw-r--r-- 1 root root  2585 2006-05-22 07:25 65-persistent-disk.rules
-rw-r--r-- 1 root root   385 2006-05-22 07:25 80-programs.rules
-rw-r--r-- 1 root root   171 2006-05-29 05:03 85-alsa.rules
-rw-r--r-- 1 root root   208 2006-05-22 08:09 85-hal.rules
-rw-r--r-- 1 root root    81 2006-01-04 03:13 85-hdparm.rules
-rw-r--r-- 1 root root   126 2006-05-15 18:43 85-hwclock.rules
-rw-r--r-- 1 root root   657 2006-01-30 05:40 85-ifupdown.rules
-rw-r--r-- 1 root root   937 2006-03-23 12:40 85-pcmcia.rules
-rw-r--r-- 1 root root    82 2006-05-22 08:09 90-hal.rules
-rw-r--r-- 1 root root  2534 2006-05-22 07:25 90-modprobe.rules
-rw-r--r-- 1 root root    75 2006-05-22 07:25 99-udevmonitor.rules
-rw-r--r-- 1 root root  1216 2006-12-14 06:55 acces_usb.rules
```

Once these 3 steps are completed plugging an ACCES USB device into the system should result in its firmware being automatically uploaded and made available for use.

1 The a3load.hex is placed here by the fxload package and should be left alone.

Section 2: AIOUSB

The API for ACCES USB devices consists of two files. These are aiousb.c and aiousb.h. It is possible to compile these into a library, but since the source code is provided it's easier to simply include the files in the build. The API depends on libusb being installed on your system. The latest version of libusb can be obtained from http://libusb.sourceforge.net/ or from your distribution's package manager.

If you are writing a program that makes use of the aiousb library with the source file called tester.c you would build it with the following command.

```
cc -o tester tester.c aiousb.c -lusb
```

Information about the functions available through aiousb can be found in the comments of the aiousb.h file.

ADDITIONAL INFORMATION

At the time of this writing a method has not been discovered by ACCES to allow a regular user account to send control messages to a USB device. Unfortunately, ACCES USB devices use control messages almost exclusively for their functionality.

If it is necessary for the final version of your program to be used by someone without superuser permissions use of the setuid bit will be required. A complete discussion of the setuid bit is beyond the scope of this document, but a brief summery is provided here.

When the setuid bit is set on an executable file it means that the access privileges of the user who executes the file will be temporarily changed to the owner of the file. This means that if root owns the executable file 'foo' and it is run by a user the program foo will run with all the same access privileges as root.

The most common example of this is the 'passwd' command. When a user runs passwd their effective privileges for the running of that program become root as only root has permission to change a password. Using the setuid bit should be done with caution in any environment where security is a concern. It is important to be certain that the program only accesses those resources required to accomplish the required tasks.