



## Solid-State Drive Driver Package

### 2.6.32 Kernel-Based Linux

#### 1 Introduction

**1.1** The Solid-State Drive (SSD) Driver Package consists of a Linux block device driver and a lock program.

**1.2** The driver was built and tested on a Linux version 2.6.32-32-generic based Ubuntu 10.04 Lucid Lynx distribution.

**1.3** The driver is for use with WinSystems, Inc. Single Board Computers (SBC) that provide battery-backed SRAM. This driver will allow this memory to appear as a disk. This will allow a user to load the driver module, make a file system, mount it, and create a file.

The following WinSystems, Inc. Products incorporate a battery-backed SRAM:

SBC:           PCM-VDX-512-2, EPX-C380-S, EPX-C380-D

**1.4** This driver is provided on an 'as-is' basis and no warranty as to usability or fitness of purpose is inferred or claimed.

**1.5** WinSystems, Inc. does not provide support for the modification of this driver. Customer application specific queries can be sent to: [support@winsystems.com](mailto:support@winsystems.com).

**1.6** This work is provided under the terms of the GNU General Public License (GPL).

#### 2 Installations and Build

**2.1** The device driver and lock program are provided in source code form as a compressed zipped folder. This document assumes the source code is placed at the directory */usr/src*.

**2.2** It will be necessary to become the root user to build and install the driver and device nodes.

**2.3** The MAJOR number for this device is allocated dynamically. A static number can be assigned by editing the *ssd\_init\_major* variable at the beginning of *ssd.c*.

**2.4** To create the device driver Loadable Kernel Module and the lock program in a command shell, execute: ***make all***. The device driver Loadable Kernel Module *ssd.ko* is created and moved to the appropriate kernel driver directory. The lock program is also built.

*make install* will install the kernel driver to a kernel directory and create dependencies.

*make uninstall* will remove the kernel driver from the kernel directory.

*make lock* will create the lock program.

*make clean* will remove objects created by the build.

*make spotless* will forcibly remove all artifacts of the build.

**2.4** The device driver can be loaded with the provided initialization script *ssd\_load* or manually. In either case modprobe is used to install the driver. The I/O base address assignment should be specified as an argument to modprobe. Executing:

```
modprobe ssd.ko io=0x220
```

This will load the module. The base address that must be used for each SBC is as follows:

PCM-VDX-512-2:	0x220
EPX-C380-X:	0x210

**2.5** Once the driver is built and loaded, a file system must be added to the disk. This can be accomplished by executing:

```
/usr/src/ssd/lock off (this is required to enable write accessibility)
mkfs -t <type> /dev/ssd
```

The argument type is used to specify the type of file system desired (e.g. ext4).

**2.6** To load and mount the disk automatically at boot the following lines can be added to /etc/rc.local file.

```
/usr/src/ssd/ssd_load
/usr/src/ssd/lock off (this is only required if write accessibility is desired)
mount /dev/ssd /mnt
```

### **3 Driver Usage**

**3.1** The device driver format utilized is the block model. This allows the device to transfer randomly accessible data in fixed-size blocks. The driver was designed to allow the battery-backed SRAM to be seen as a disk by the operating system. No application interface is required once the drive is mounted.

## **4 User Programs**

### **4.0 Lock**

The “lock” application is a simple program that allows the write-protect feature of the drive to be switched. The drive defaults to write-protect mode and is read only until this feature is disabled.

To disable the write protect feature, execute the following: `./lock off`

To enable the write protect feature, execute the following: `./lock on`