

## Building Slackware-style Packages (.tar files with scripts)

### Abstract:

Slackware-style installation packages include scripts that can install, remove, and modify files in an IPm controller. In this regard, they are enhanced '.tar' files. Slackware packages can check the version of installed software, perform conditional operations and other advanced functions. Although most Slackware functionality is compatible with an IPm, only the features described in this technical note are supported by Sixnet.

This information is applicable to IPm firmware v1.4 and I/O Tool Kit v2.2 or later.

### Slackware-Style Packages

Please take note that Slackware packages and what we refer to as Slackware-style packages are not necessarily the same thing. Sixnet does not support the slackware package system (pkgtool, installpkg, removepkg, upgradepkg, makepkg, etc), which was created for the Slackware Linux distribution. Since the IPm is an embedded device, it does not maintain an internal database of installed slackware packages. Most slackware packages can, however, be installed successfully using the Sixnet I/O Tool Kit.

This tech note is aimed at trying to help users build packages for the Sixnet IPm product line. It is just a rough guide and is not an all inclusive "how to". It is assumed that the user is familiar with shell scripting and building applications for the Sixnet IPm. The procedure described below is the only supported method of creating packages for the IPm.

Please respect any existing software licenses. It is always a good practice to check licenses to make sure you are in compliance. If the license is a GPL licenses you are required to provide the source files for the package.

### Preparation

Create a directory that will simulate the IPm file system. This directory must be completely empty.

```
# mkdir -p /opt/tmp
```

**Note:** The above command will create the directory /opt/tmp. If the directory already exists, check to see if /opt/tmp contains any files or directories. If /opt/tmp is not empty back up the directory and create a new /opt/tmp.

```
# mv /opt/tmp /opt/tmp.orig  
# mkdir -p /opt/tmp
```

/opt/tmp will represent the root ( / ) file system in the IPm. All files will be installed relative to this directory.

i.e.     /opt/tmp/usr/local/bin on the PC will map to /usr/local/bin on the IPm.

Before we can build the package, we need to pay special attention to the following items:

### Permissions

It is important that the files and directories being packaged have the correct permission and ownership. If a file needs to be of a specific group or owner, you must create that group or user. This can be done manually or by making the necessary entries in the installation script.

### Dependencies

If a package requires libraries that are not part of a standard IPm installation, you will need to create a new package with just those libraries or include them in the package. Creating a new package for the libraries is the better option. If libraries are being added to the station, be sure to edit `/etc/ld.so.conf` and run `'ldconfig'` to create the updated library cache file.

### Installation Location

It is important to keep track of where all files are installed. Some programs need to know the location of certain files or libraries. If you are building a package from source, make sure that all the files are installed to the `/opt/tmp` directory. Sometimes `'make install'` will copy some files to an undesirable location. If this happens, simply copy those files to the correct path in `/opt/tmp`.

## Installation Script

Create the `'install'` directory in `/opt/tmp`.

```
# mkdir /opt/tmp/install
```

Slackware-style packages use an installation script to provide for a more flexible installation. This script is always located in the `'install'` directory. The installation script must be called `'doinst.sh'`.

All slackware-style packages must contain the `'doinst.sh'` file. It is possible that a valid package contains nothing more than the `'doinst.sh'` file. The `'doinst.sh'` script should be used to remove, relocate, edit, or alter permissions on existing files. These files may include files in the package or files which already exist in the station. Special care should be taken, when writing the installation script, so as to not alter pre-existing files in a way that may cause instability in the station.

The script can be very simple and may not be interpreter-specific. However, the script must be set executable! Failure to set the `'doinst.sh'` script executable will cause your install to fail.

More complex scripts will need to use the "shebang" notation (`#!/`). Both `'perl'` and `'sh'` (ash) are supported in the default IPm installation. To specify the interpreter, the shebang will be the very first line in the `doinst.sh` script, and will look something like the following:

```
#!/usr/bin/perl          or          #!/bin/sh
```

**Note:** Be sure to use the complete path to the interpreter.

## Building a Package

By now you should have a clean directory to do your local installation (/opt/tmp).

Copy the files that you wish to install in the station to the proper location, in /opt/tmp. Again if a file is supposed to be installed in /usr/lib, on the IPm, place the file in /opt/tmp/usr/lib, on the PC.

```
i.e.    # cp myfile /opt/tmp/usr/bin/
```

**Note:** It may be possible to skip the copy step, mentioned above, if the package was built from source and the installation prefix is set to the appropriate location in /opt/tmp. In this case, '**make install**' will copy the files to the proper locations and set the necessary permission.

Set the appropriate permissions and ownership on all files to be installed.

```
i.e.    # chmod 755 /opt/tmp/usr/bin/myfile
        # chown johndoe /opt/tmp/usr/bin/myfile
```

The above example sets 'myfile' to be readable and executable by all, but only writeable by owner. The owner is then set to 'johndoe'.

**Note:** If a user does not exist in the IPm, one will need to be created. Keep in mind that user/group IDs on the PC may or may not match the user/group IDs in the IPm. If the owner on the PC is set to johndoe (uid: 11), and janedoe (uid: 11) exists on the IPm, janedoe will be the owner. The same issue exists for groups. Look at **/etc/passwd** and **/etc/group** to verify the correct IDs.

If you have dependencies, and you have not created a separate package for them, be sure to include them. Remember to set the appropriate permissions and ownership.

If /opt/tmp does not contain an 'install' directory, create one.

Create the 'doinst.sh' script, which must reside in /opt/tmp/install/.

Change to the /opt/tmp directory

```
# cd /opt/tmp
```

Tar up (and compress) the package.

```
# tar zcvpf mypackage_1_3.tar.gz *
```

**Note:** It is advised to create packages with version numbers as the IPm does not maintain a database of installed packages. Slackware packages use the '.tgz' extension exclusively. The Sixnet I/O Tool Kit and IPm recognize both the '.tgz' or '.tar.gz' extensions.

Copy the package to a location accessible by the I/O Tool Kit.