

## Creating CVS Projects

This procedure describes how to create a new software project from scratch, how to place the project into the CVS repository and, finally, how to add an alias for your new project to the CVS modules file so that you can manipulate your project code with the shorthand form of the CVS commands. Throughout this note references are made to rfiles as the developer's organization, please substitute the name of your organization where necessary.

### 1.0 Creating New Projects

Creating a new project is quite painless. Here's how:

1) Select a name for your new project. Short descriptive names are best (e.g., io100 for the IO-100 board) since you will more than likely have to type this name many times throughout your career. Make sure that your chosen project name is not already in use by issuing the `incvs` command which checks and reports whether the supplied name is already used in the repository.

```
incvs xxx
```

where **xxx** is the name you have chosen your new project.

2) Use the setup tool<sup>1</sup> to create a "working directory" for your new project. This working directory is not a "sandbox", it is only used to collect files to be initially included in your new project:

```
setup xxx_orig
```

where **xxx** is the name you have chosen your new project. The setup tool will offer to provide template files to start-off your new project. You will have to edit and/or rename some of these template files to tailor them to your specific needs.

3) Try to get the files in the working directory into a functioning state before committing them to the repository. This will minimize the number of superfluous repository file revisions. When you have a clean and operational project in your working directory you may create a CVS repository using the procedure in **2.0** below.

---

<sup>1</sup>See "Managing Embedded System Software with the RFI-ES Development Tools" for a description of the setup tool.

## 2.0 Creating New CVS Repositories

Creating a new CVS repository for your project is a bit more complicated than creating the project itself. Here's how:

1) The files to be placed into the repository should be in a "working directory" that does not have the same name as your project (e.g., io100\_orig when creating the io100 project repository.) Make sure that this working directory contains only the files to be placed into the repository. Only source and project control files are suitable for the repository. The Target, Headers, \*.doc and object files do not go into the repository since they are automatically generated. A 'make clean' should do the job.

2) Make sure that all files in the working directory have owner and group write permissions:

```
chmod 664 *
```

3) Place the files into the CVS repository:

```
cvs import fermi /org/shared/xxx FERMI initial
```

where **org** is your organization's name and **xxx** is your shared library project name, or:

```
cvs import fermi /org/fe/xxx FERMI initial
```

where **org** is your organization's name and **xxx** is your front-end project name, or:

```
cvs import fermi /org/dsp/xxx FERMI initial
```

where **org** is your organization's name and **xxx** is your DSP project name. At this point CVS will put you into an editor so that you can provide a comment as to the nature of your activity. Enter a meaningful message at the top of the editor window like "Creation of new shared project io100." and close the file.

4) At this point a copy of your files have been placed into the repository but your working directory still contains the original files. You can save your working directory contents in case something went wrong during the process, or you can throw them away when you have the nerve.

5) If you have not already done so add an alias for your project to the CVSROOT modules file using the procedure in **3.0** below.

6) To get a working copy of your project in a sandbox directory use the setup tool:

```
setup xxx
```

and follow the prompts, or use the cvs checkout command:

```
cvs checkout xxx
```

to get a project sandbox in your cwd.

### 3.0 Adding a Project Alias to the CVSROOT modules File

The CVS repository exists in a file hierarchy underneath \$CVSROOT/fermi. DSP libraries are in \$CVSROOT/fermi/org/dsp, shared libraries are in \$CVSROOT/fermi/org/shared and front-end projects are in \$CVSROOT/fermi/org/fe. Again, org is your organization's name. When issuing CVS commands it is tedious to type path names like fermi/rfies/shared/io100 so the CVS'ers provided a way of specifying aliases for projects. CVS project aliases are defined in a modules file that is itself within the CVS repository. To add a CVS alias for your project you must get a copy of the CVS modules file in your sandbox, modify it to include your alias specification(s) and put it back into the repository. Here's how:

If you already have a copy of the CVSROOT project in your sandbox you MUST update the project:

```
setup CVSROOT
update
```

and go on to step 4 below. Otherwise you need to get a copy of the CVSROOT project as outlined in steps 1 through 3 below, then continue the procedure at step 4.

1) move to your sandbox directory:

```
cd $USER_SANDBOX_DIR
```

2) check out the CVSROOT project and move into its directory:

```
cvs checkout CVSROOT
cd CVSROOT
```

3) open the modules file with your favorite editor:

```
edit modules
```

4) Add the alias specification(s) for your project(s). Follow the examples set by previous projects already in the modules file. Recall that projects are classified as dsp, shared or fe and should be placed in the appropriate section of the modules file.

5) in case someone else was editing the modules file and committed it while you were working, you need to update your copy to merge in any changes:

```
cvs update
```

6) put the file back into the repository:

```
cvs commit
```

At this point CVS will put you into an editor so that you may provide a comment as to the nature of your changes. Enter a meaningful message at the top of the editor window like "Added alias for shared project io100." and close the file.

7 - optional) you can release and delete your sandbox copy of the CVSROOT project:

```
cd ..
cvs release -d CVSROOT
```

This is not necessary since CVS is non-locking and having a copy in your sandbox will not affect the work of others. In fact, if you are going to add aliases to the modules file often you might want to keep the CVSROOT project in your sandbox. If you do keep the CVSROOT project in your sandbox remember to do an update before you make any future changes to the modules file.

End.

070611

DCV