Creating CVS Projects

This note describes the process of creating new software projects, placing them into the CVS repository and adding a project alias to the CVS repository's modules file so that the project can be easily accessed with the setup tool.
Throughout the note the Low Level RF Group's organization name 'rfies' is used as an example wherever the organization name must be provided, please substitute your organization name 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 drivers.) Make sure that your chosen project name is not already in use by issuing the 'incvs' command. The incvs command will search the repository and report whether the supplied name or any derivative form has already been used.

```
incvs xxx
```

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

2) Use the 'setup' tool to create a working directory for your new project:

```
setup xxx
```

where **xxx** is the name you have chosen your new project. The setup tool will create a project working directory and offer to install template project support files by asking a series of yes–no questions. You may have to edit and/or rename some of these template files to tailor them to your specific needs.

3) Add your project's source code files to the newly created project working directory ~/esd/src/**xxx**. When you have completed preparing your projects initial file content you may create a CVS repository using the procedure in section **2.0** below.

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 with the same name as your project (e.g., io100 when creating the io100 project repository.) Creating new projects with the setup tool as described in section **1.0** above will assure that everything is in canonical form. Make sure that the working directory contains only those files that are to be placed into the repository. For example source and project control files are suitable. Automatically generated files such as the Target and Headers files, any *.doc files and any *.o, *.a and *.out object files should not be placed into the repository. A `'make clean'` should properly clean the working directory.

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

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

```
chmod 664 *
```

4) 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 like "Creation of new Low Level RF shared project io100." and close the file.

5) At this point a copy of your files has been placed into the repository and your working directory still contains the original files. You should save your original working directory under a new name in case something went wrong during the import process, for example:

```
cd ..
mv xxx xxx_orig
```

where **xxx** is your project name.

6) Use the setup tool to get a working copy of the newly created CVS repository for your project:

```
setup xxx
```

Simply follow the setup tools prompts. Setup will create a new sandbox directory and configure it for development. Do a project make to assure that all is well. When you are confident that the repository is correct you may delete the copy of your original files created in step 5 above.

Revised 16020610
DCV

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 CVSers provided a way of specifying shorter aliases for projects. CVS project aliases are defined in a modules file that is located in the CVSROOT CVS repository. To add a CVS alias for your project you must get a copy of the CVSROOT's 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 on with step 4.

Note: the `setup` command will not get a copy of CVSROOT. It must be done manually as follows:

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:

```
nedit modules
```

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

5) in case someone else has edited 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 can provide a comment as to the nature of your changes. Enter a meaningful message 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 anticipate adding aliases in the future you might just want to keep the CVSROOT project in your sandbox. If you do keep the CVSROOT project in your sandbox remember do a CVS update each time before you make any changes to the modules file.

End.

Revised 16020610
DCV