Developing New IFF FORMs and Chunks

by Christian Ludwig - Multimedia Standards Engineer, CATS

IFF has been one of the keys to the Amiga's superiority in multimedia applications, allowing interchange of media elements between packages. The wealth of standard IFF FORMs and chunks gives the Amiga user data-sharing capabilities that are virtually unequaled on other systems. The Amiga's ability to render an image, touch it up, convert it to a different display mode, and load it into in another package is something that is a chore on other platforms, simply because the format of the image file may be different from one application to the next. IFF files lessen the need for ‘‘conversion’’ software, because most Amiga applications can read and write them.

Since its introduction as an open standard in 1985, IFF has widened to encompass data of many sorts--and the need for new IFF types continues to grow. To satisfy these growing needs, developers will continue to define and support new IFF types. **Now that both the console and *iffparse.library* support sharing IFF files through the clipboard in release 2.0, IFF support is more important than ever.**

When developing a new IFF type, there are several steps you should follow:

❏ Discuss needs and specifications within the developer community and with Commodore.

> The most important thing about designing IFF FORMs and chunks is that they meet the data storage and transfer needs of multiple applications. When more than one product uses the same IFF type, the market widens for all products that use that IFF type. Users are not forced to use one product or another, but can buy as many as they need to get the job done, fully utilizing all the features that each product has to offer. This step helps to ensure that a

proposed IFF form or chunk type is flexible and isn't redundant. A good way to start this kind of discussion is to post a message in Commodore's *amiga.dev/iff* conference on the BIX electronic network. Also, feel free to contact me to discuss your IFF needs and issues. You can reach me at (215) 431-9316 or on BIX (cludwig). I can also be reached via UseNet InterNet e-mail at:

chrisl@cbmvax.commodore.com or ...[uunet| rutgers]!cbmvax!chrisl.

❑ Implement the new type and conduct feasibility tests.

Before settling on a format, set up prototype code to test the proposed format. This will help to prove that the idea is sound and can be implemented in software before others try to use it.

❑ Submit specifications to Commodore.

Coming up with a new kind of IFF FORM or chunk is easy--almost too easy. Just about anyone can follow the IFF guidelines and define their own FORM or chunk. If every application used a different IFF FORM, one application would be unable to share data with another because it can't read the other application's IFF FORM[1]. It's like making up a new word for something that everyone sees every day. You may understand what the word means, but when you try to use your new word to communicate with others, they won't understand you. Further, deciding to use a pre-existing FORM or chunk in a new and different way is a lot like making up your own meaning for a pre-existing word. Confusion results when programs try to read FORMs or chunks whose meaning was altered by a non-conforming program.

To avoid the problem of incompatible IFF types, register your new IFF types with CATS. CATS acts as a ''dictionary'' of IFF types. By submitting your proposals for FORM or chunk types to CATS, you help prevent duplication of an existing data type. Also, if you register your new IFF type, it is more likely that it will be adopted as an IFF standard that other applications will use. For example, the ANIM form came from third party developers who proposed and refined the format. Now ANIM is the de facto standard for animation files.

CATS wants this last step to be as easy as possible, so we've included a standard form at the end of this article. Just photocopy the form whenever you need it, and use it as a guide for submitting your FORM or chunk specifications. The registration form should be

[1] Also, according to the IFF guidelines, any program that reads an IFF file with a chunk it does not understand must not write out that chunk when attempting to re-save the IFF file.

accompanied by a disk containing ASCII text files of the IFF specification. If available, include some code examples which demonstrate the use of the IFF type. Please do not place copyright notices in your specifications or examples so that CATS may make them available to other developers.

For an excellent example of a third party FORM specification, see the WORD FORM in the third party specification section of the IFF manual. For an example of chunk descriptions, examine the 8SVX FORM's SEQN and FADE chunks in the same section.

Note that even you don't plan to release the specifications of your FORM or chunk, you must still register the *name* with CATS. This is the only way to prevent name conflicts in IFF files. You should register your FORM and chunk names *before* finalizing your product and its documentation in case there is a name conflict with an existing IFF type.

❏ Distribute final specifications to the developer community.

Once you have registered your FORM or chunk with CATS, you should release the specifications of the chunk to the developer community. Although CATS publishes FORMs and chunks in the IFF Manual and occasionally in Amiga Mail, developers should not rely on these methods to distribute their IFF type specification. One of the most efficient ways to distribute your specification is to include it in your application's documentation. Another good distribution method is to post the final specifications in *amiga.dev/iff* on BIX. Distributing the specification will increase the probability of your form or chunk becoming a standard.

❖