

ENHANCE DESIGNER'S PRODUCTIVITY WITH XSTREAM

UMESH SISODIA

CADENCE DESIGN SYSTEMS

91-120-2562842

usisodia@cadence.com

MANISH BARONIA

mbaronia@cadence.com

RAJAN ARORA

rarora@cadence.com

INTERNATIONAL CADENCE USERS GROUP CONFERENCE

September 13 - 15, 2004

Santa Clara, CA

ABSTRACT

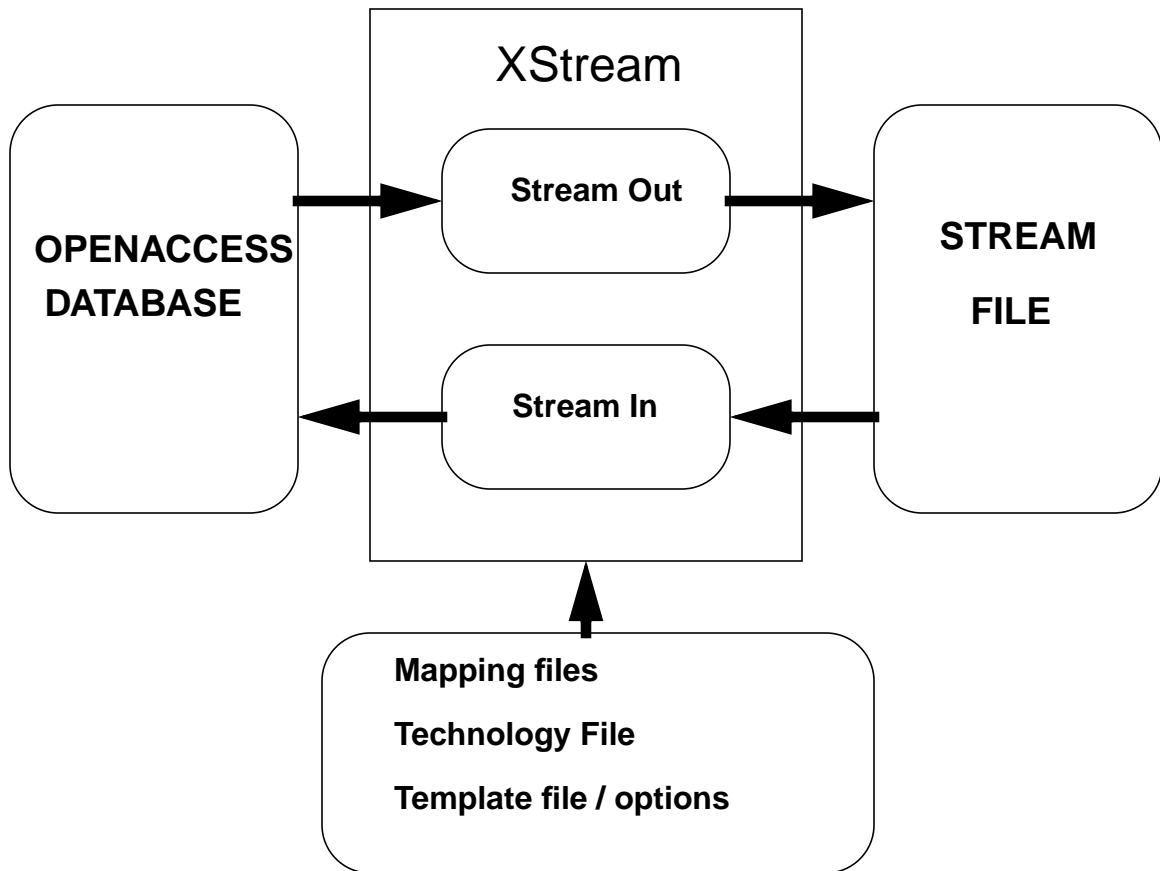
With the increasing complexity of chip designs, the size of Stream/GDSII data is increasing, making performance and capacity of Stream translators critical. XStream is Stream translator that works on DFII on Open Access (OA) environment of Virtuoso platform. This paper describes the performance and capacity, memory prediction and usability features in XStream. These features help designers iterate over layout very fast to meet tapeout schedules. High emphasis has been given on performance and capacity in XStream, which makes it one of the best Stream translators in the market. The paper discusses in detail, features like optimized and compressed data, support for files greater than 2GB on 32bit machines and support for large number of layers. For complex chip designs, Stream translators may exhaust available virtual memory (VM) and abort translation. For a chip design, XStream uses heuristics to detect insufficient memory very early and gives details of required VM to the user. Sometimes, wrong user inputs can lead to corrupt translations, leading to loss of designer's valuable time. To overcome this, XStream supports an interactive mode in which it does a very fast scan of the layout, where it collects useful information. Then it interactively allows user to correct these issues before the actual design translation. Also, many features have been added in XStream GUI to provide convenient interface to users for working with the Stream translator. In this paper, we present in detail the performance, capacity and usability features with the results on benchmark designs.

INTRODUCTION

XStream is the Stream format interface tool with Custom IC Physical Design tools on Open Access, customized in the DFII environment. Stream format (GDSII) is an standard format for representing the layout data. It is used for the following purpose:

- Tape-out purpose (To transfer the final chip layout from design house to the mask shop.)
- To exchange layout data between different tools to complete a design cycle.
- To archive the layout design data.

Following figure shows the use model of XStream.



With the continuously growing design complexity, the size of tape out data represented in Stream format is becoming huge. Normally XStream is used at a stage when design data is in the form the full chip layout ready to be manufactured. Data volumes are huge at this stage and any type of data processing requires lots of designer's time & system resources. The EDA tools used at this stage have to ensure that there are no performance / capacity bottlenecks and there are no usability issues which may force designers to commit mistakes and repeat the time consuming steps. XStream is the next generation Stream translator introduced in DFII on OA. This paper exhibits the following benefits of XStream:

- Performance
- Capacity
- Usability

- Memory prediction.
- Interactive mode.
- Better GUI.
- Other usability features.

PERFORMANCE

XStream translates data to and from Open access database, it automatically gains in terms of performance because of faster data access through OA APIs. Right from the beginning of development phase, focussed efforts were made to ensure that it is free from any performance bottleneck. Performance benchmark was done on some of the large designs against PIPO, and we see huge gain in XStream performance.

Following data shows the performance comparison between XStream (the new Stream translator available in DFII on OA) and PIPO (Stream translator available in DFII on CDBA)

Stream Out performance benchmark (XStream Vs. PIPO)

Testcase	PIPO IC4.4.6 FCS Elapsed time (seconds)	XStream Elapsed time	Performance Gain (%)
Design1	5307	669	7.93 X
Design2	940	159	5.9 X
Design3	1258	470	2.68 X

Stream In performance benchmark (XStream Vs. PIPO)

Testcase	PIPO IC4.4.6 FCS Elapsed time (seconds)	XStream Elapsed time	Performance Gain (%)
Design1	19838	1382	14.35 X
Design2	5497	946	5.81 X
Design3	5153	1251	4.12 X

Customer Wins:

- Was able to win several performance evaluations & competitive benchmarks at customer sites (UMC, SunPlus, ATI, Agere, Intel etc.)
- UMC was able to Stream In a 5 GB design in 50 minutes.
- During VCE evaluation, Intel was able to StreamIn a 11 GB design in 75 minutes.
- ST was impressed with the speed of strmin in VCE, while evaluating DFII on OA for their 65nm flow.

CAPACITY

Open Access is a compact database. Various capacity features of Open Access are automatically inherited by the XStream.

- It uses memory mapping for reading and writing Stream file, hence the files greater than 2 GB can also be accessed using the 32bit binaries. Users will have to use 64bit XStream only if the contents of a single cellview exceeds 4 GB.
- It supports more than 255 layers & purposes.
- Stream In supports Stream files having more than 32767 cellview. In the DFII library structure, cells are represented by subdirectories. Most of the unix system imposes a limit of 32767 on the number of subdirectories in a directory, hence a DFII library cannot have more than 32767 cellviews. If a Stream file has cellviews more than 32767, then strmin automatically breaks it into multiple DFII libraries.

USABILITY

MEMORY PREDICTION

With the designs becoming more and more complex, hitting the system resource limits by the tools that work on entire chip designs is becoming more frequent. The most common resource limit that is hit by these tools on big chip designs, is the total Virtual Memory (VM) available for a process. There have been numerous instances where the designers run EDA tools on a big design and after running for hours, these tools exit because of exhausting all the available VM. This leads to re-run of these tools on higher capacity machines resulting in loss of hours of designer's productivity and delay in tapeout. XStream strmin has an inbuilt memory prediction module which can identify VM limit hits by predicting the peak VM requirement for a design much earlier than the actual run time on the design. This solution can save a lots of runtime iterations for the designers and improve the productivity. During StreamIn, before starting the actual translation, XStream uses some heuristics to calculate the peak virtual memory that might be required during translation. If the memory available to the process on the machine is not sufficient for the successful translation then it gives proper error messages to user for remedial actions, rather than running the whole time consuming process and failing at the end. The time consumed in memory prediction is approximately 1% of total translation time. The memory predictor module has been tested on variety of chip designs.

Strmin translates one cellview at a time, the peak memory requirement during translation will be proportional to the largest cellview in the input Stream file.

The results show that the predicted peak memory falls within 10% of actual memory consumption during the run in most of the cases. Following table shows the results of testing done on different customer chip designs.

S. No	Testcase	Size of biggest cellview in Stream file	Predicted VM	Actual VM consumption	Deviation %
1	F13_DSP_MvHBSP	1429906	480357	520092	7
2	F13_DSPMvU3TP	5055996	1821599	1830530	1
3	chip_spk	429514056	152850679	163166090	6
4	MvU3SP_1640x338	16072152	5892124	6226819	5
5	COLLAR_spkl	197121924	72503435	75984194	4
6	chip	321445736	119888591	116582896	2
7	F13B	71066448	24267339	27104669	10
8	pmmu	88561942	33631196	33631196	3

The maximum memory that a process can allocate depends on the following parameters:

Description	Acronym used in the paper
Total virtual memory available on the system.	totalSystemVM
User can set the limit on the max memory available to a process, by using 'limit' command of unix.	limitProcessMax
32bit process cannot access memory greater than 4 GB. On linux this limit is 3 GB.	hardMemLimit

The lowest of the three parameters will decide the maximum memory that can be allocated to a process.

XStream strmin will make the first pass on the input gds file and predict the maximum memory that will be required during translation. If the predicted memory exceeds the maximum memory that can be allocated or is very close to it then one or more of following messages are displayed before starting the translation.

ERROR (87): Process may require more virtual memory than 32 bit system can handle. You may need to run 64bit version of the product.

Available Memory:%llu MB Predicted Memory:%llu MB.

ERROR (88): Process might require more virtual memory than available on the system.

Available Memory:%llu MB Predicted Memory:%llu MB.

INFO (88): Process might require more virtual memory than available on the system.

Available Memory:%llu MB Predicted Memory:%llu MB.

ERROR (90): Process may require more virtual memory than available on the system for a process. You can increase this limit by using the 'limit' command.

Available Memory:%llu MB Predicted Memory:%llu MB.

INFO (91): Process may require virtual memory near to what it is available on the system for a process. You can increase this limit by using the 'limit' command.

Available Memory:%llu MB Predicted Memory:%llu MB.

INTERACTIVE MODE

When `strmin / strmout` is invoked from command line, by default it works in non-interactive mode. To run it in the interactive mode, `-i` option has to be specified. XStream is a translator, generally users have to provide various options/information that has to be utilized during translation. Example of this information are command line options, layer mapping information, cell mapping information, input Stream file / library etc. Before starting the actual translation these inputs are parsed and information is stored. If there are some discrepancies in the inputs then messages are displayed on the screen, these messages can be FATAL, ERROR, WARNING, INFO depending on the severity. If a FATAL message comes, then translation does not proceed, but if there are only ERROR, WARNING & INFO then translation proceeds.

If XStream is invoked in interactive mode, and there are some messages (ERROR or WARNINGS), then after displaying all the messages, user will be prompted whether to continue or stop the translation.

Following question will appear on the screen:

Do you wish to continue?:

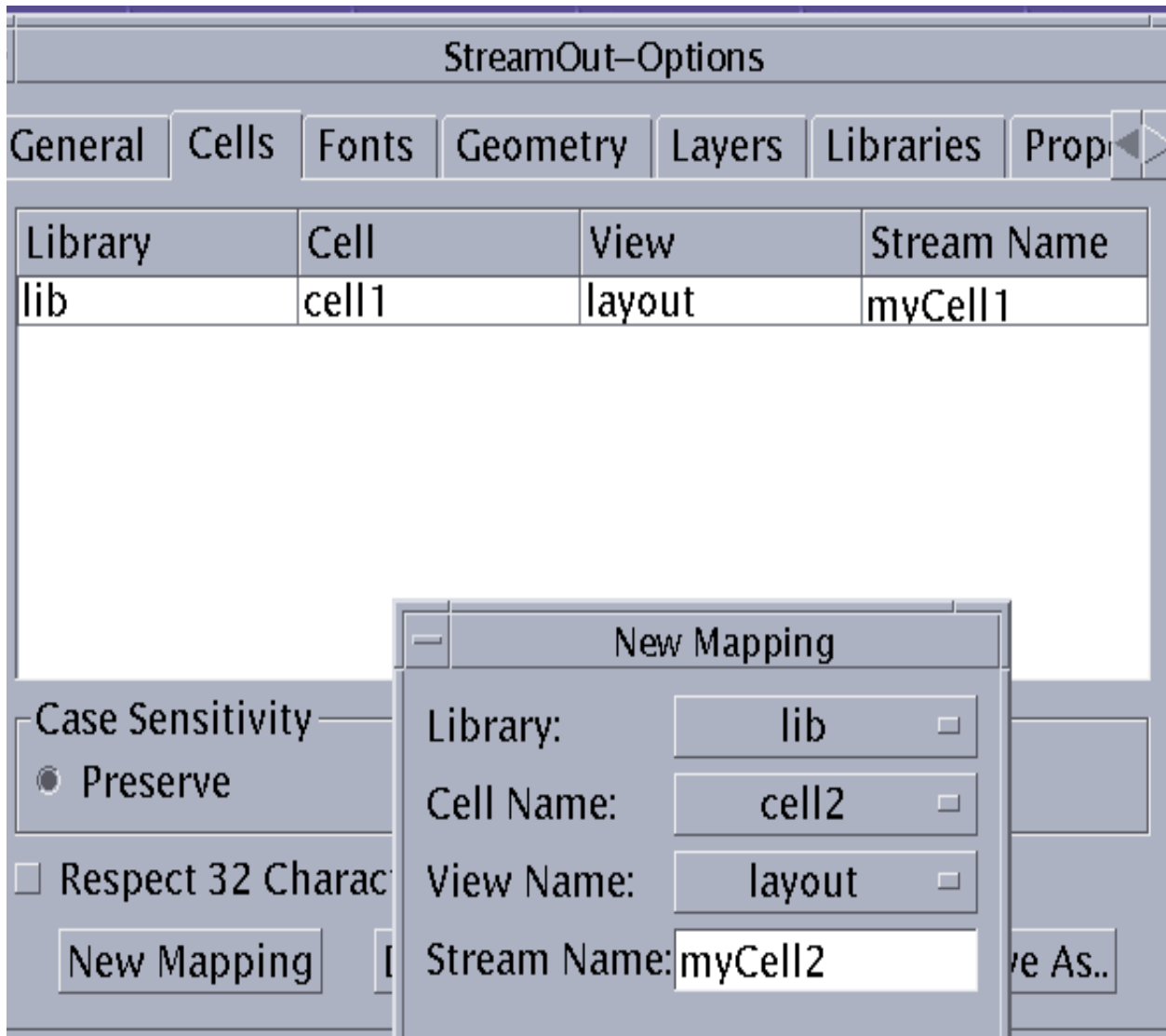
If user press 'y', then it continues else it stops. Many times it have been observed that because of discrepancies in the inputs, users get wrong output which they come to know only in the end after looking in the log file. Then they will have to repeat the time consuming translation steps, which significantly affects productivity. Situation can further worsen if user does not notice the error, warnings in the log file and transfer the output data to next design steps. Following are some of the example where interactive mode helps.

- If any of the mapping file (layerMap file, cellMap file, propMap file, fontMap file, refLib list) have an illegal entry then error message is issued and entry is ignored. So if the user has made some mistake while entering the mapping info, then translated output may be wrong. In interactive mode user will be informed about these mistakes before starting the translation.
- During `strmin`, if layerMap file is provided then only those layer purpose are translated which are specified in the layerMap file. If interactive mode is being used then before starting the actual translation user will be informed about all the layers present in Stream file but not specified in the layerMap file.
- During `StreamIn`, if cellName mapping leads to duplicate names, then warnings will be displayed before starting the actual translation.

BETTER GUI

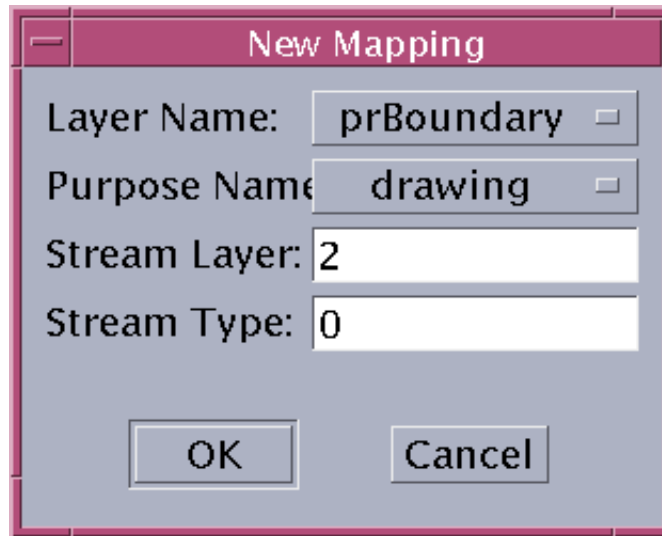
XStream has much better QT based GUI as compared to legacy Stream translator PIPO. Following are some of the GUI features which helps in enhancing the designer's productivity.

- In PIPO if user have to specify the cell name mapping, he will have to create the mapping file by typing the cellview names. In XStream cell name mapping can be created interactively.



- Similarly layer mapping, font mapping, property mapping, refLib list can be created interactively. User can also save these mappings in the text files which can be used while invoking XStream from command line.

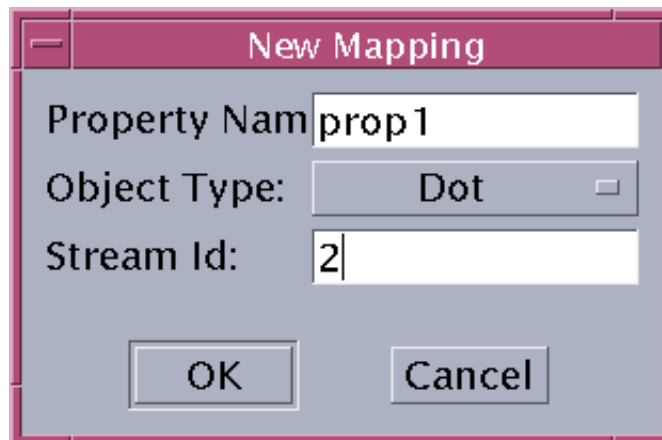
Layer Mapping:



A dialog box titled "New Mapping" with a grey background and a maroon title bar. It contains four input fields: "Layer Name" with a dropdown menu showing "prBoundary", "Purpose Name" with a dropdown menu showing "drawing", "Stream Layer" with a text box containing "2", and "Stream Type" with a text box containing "0". At the bottom are "OK" and "Cancel" buttons.

Layer Name:	prBoundary
Purpose Name:	drawing
Stream Layer:	2
Stream Type:	0

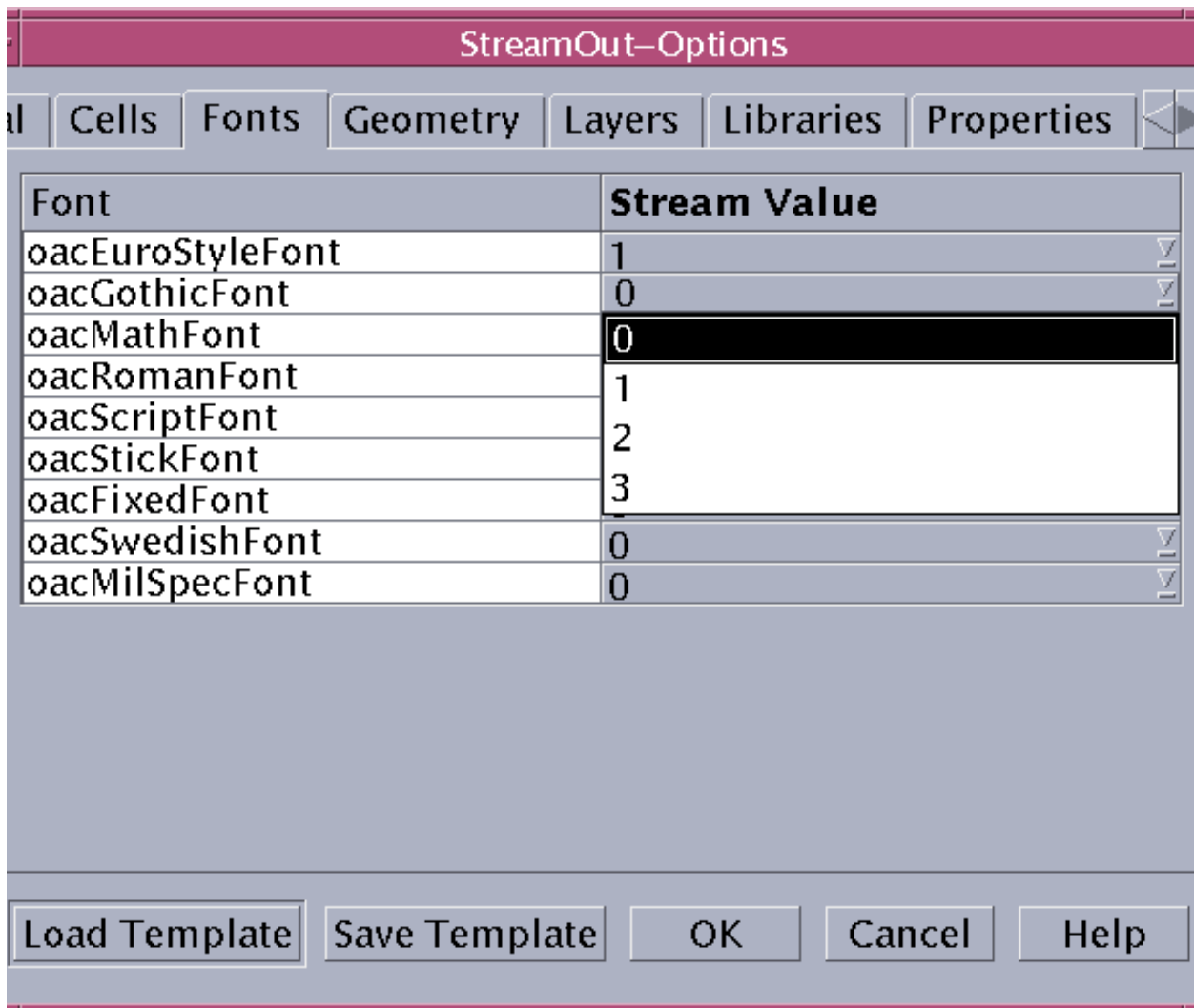
Property Mapping:



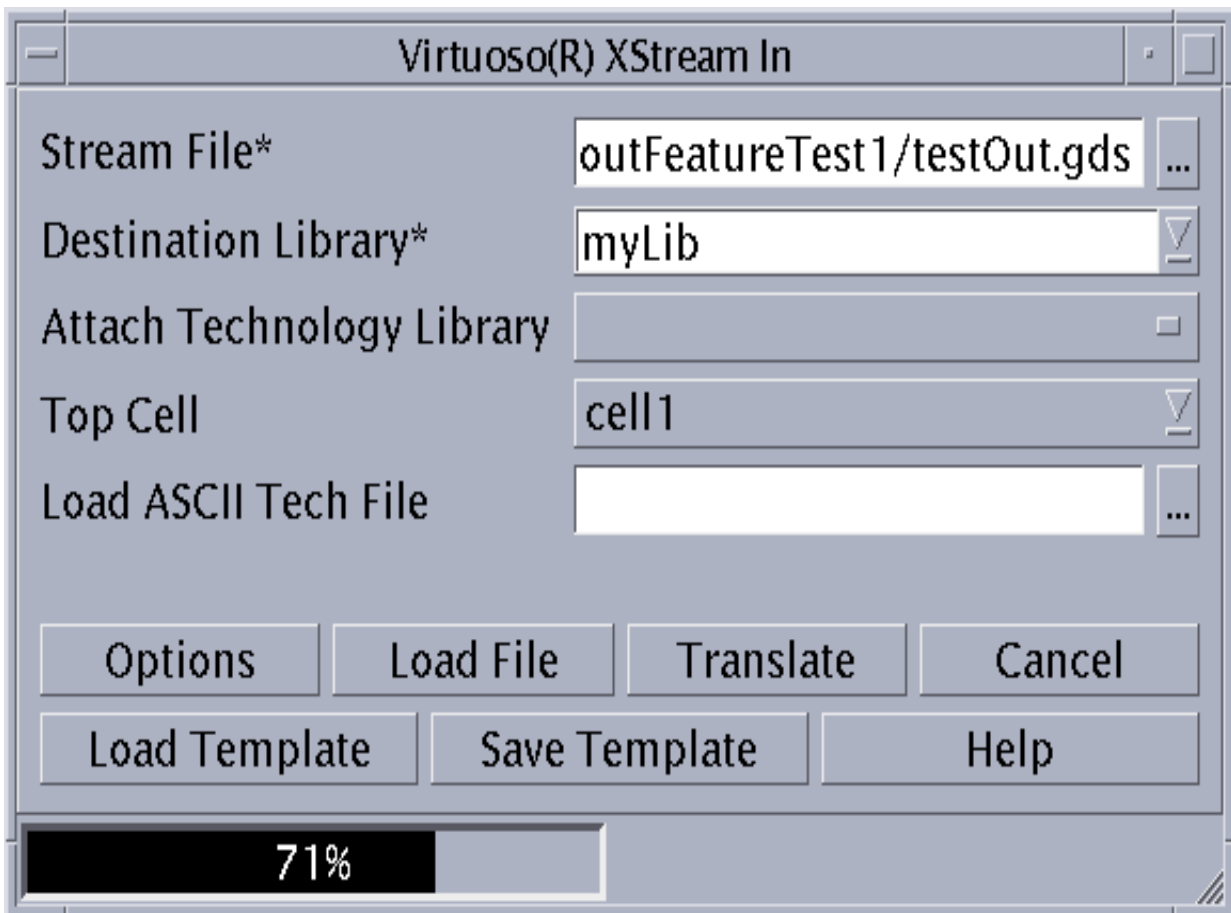
A dialog box titled "New Mapping" with a grey background and a maroon title bar. It contains three input fields: "Property Name" with a text box containing "prop1", "Object Type" with a dropdown menu showing "Dot", and "Stream Id" with a text box containing "2". At the bottom are "OK" and "Cancel" buttons.

Property Name:	prop1
Object Type:	Dot
Stream Id:	2

Font Mapping:



- Progress bar indicates the percentage completion of translation, this can be very useful while translating large designs.



- In StreamIn GUI, there is a button 'Load File', when this button is pressed, then a quick pass over the input Stream file is done and various lists in the GUI are populated with the collected information. User will be able to see following info about the Stream file.
 - Users can see the names of STRUCTURES (cells) defined in the Stream file, and can translate a portion of the hierarchy.
 - User can see the layer, datatypes used in the Stream file and can use them to create the layer mapping.

OTHER USABILITY FEATURES

- Automatic layer mapping: In PIPO, it is necessary for the user to specify the layer mapping during Streamout, so he will have to find out what layer-purpose are used in the design. In XStream, if user does not specify layer mapping, then automatic layer mapping algorithm of XStream will be utilized, in such case XStream will generate a layer mapping file which can be used during StreamIn to recreate the original design.