Lahey Product Information

Overview of LF95 Linux

LF95 Linux v6.1 was released on November 28, 2001. It features a performance enhancement called prefetch for Pentium III and Athlon processors. Almost all benchmark tests have shown a performance improvement with LF95 v6.1 with prefetch. Beta testers and benchmark tests have seen performance improved by about 10%, with a small number of tests showing up to 25% improvement and a small number showing no improvement or slight performance degradation. Try LF95 v6.1 with your code today!

LF95 v6.1 includes new enhancements and features to make programming easier. Lahey's industry-leading compiler and runtime diagnostics now include column position error messages and improved runtime speed when checking subscripts. Moving code between LF95 for Windows and LF95 Linux v6.1 is easy because most Windows, Winsock, GNOI, and Autocone code will port with no changes. Multiple compilers are not required as LF95 supports your Fortran 95, 90, and 77 code. Both PRO and Express CDs include source and instructions for the MPICH message passing library to get your Linux cluster up and running.

Features of LF95 Linux PRO

- Auto-parallelization for multi-processor machines
- OpenMP support
- Optimized BLAS (Basic Linear Algebra Subprograms) and LAPACK (Linear Algebra Package) libraries callable from parallelized Fortran programs.
- Thread-safe SSL2 math library callable from parallelized Fortran programs.
- Win32/NT math library callable from parallelized Fortran programs.
- Win32/NT Starter Kit, W3K, a subset of Win32/NT by Interactive Software Services, Ltd. W3K is a portable source and graphics package for Fortran 95/90 developers.
- Autocode, by Polymedron Software, a mature utility that automatically determines program dependencies.
- Handover manuals.
- Unlimited Free "Call-back" phone support.

Features of both PRO and Express

- NEW! Run your applications on Linux clusters at no extra charge.
- NEW! Improved global compile-time diagnostics.
- NEW! Improved runtime diagnostics.
- NEW! MPICH source code.
- FSB Debugger.
- MPI compatible.
- Supports static linking with Fujitsu C, g77, or egcs.
- Free technical support.
- Compatible with TotalView parallel debugger.
- ANS105-Compliant Fortran 95.
- Supports Fortran 95 and 77.
- Compatible with standard IMSL (IMSL available at additional charge).


Lahey offers competitive floating and site-license pricing. Educational discounts are available.

Prices do not include shipping and handling.

Call today with your order and questions. 800-548-4778 or 775-831-8123 / sales@lahey.com

Buy Online at www.lahey.com!
Dear Fortran Programmers,

As Lahey’s response to the World Trade Center attacks, we have been contacting firms with registered copies of Lahey products in the New York City area to see how best we can help them recover. If we missed your organization, please give us a call.

On our web-site, there’s a tutorial on threading derived-type objects called “Putting Mega-Byte Memory to Work.” Lahey will continue to publish articles on Modern Fortran subjects both in Essential SOURCE and on our web-site www.lahey.com. So far, there are also papers on style and the Bryan Price array expressions articles that appeared in Fortran SOURCE years ago. If you have a topic you’d like to see, send e-mail to tlahe@lahey.com.

Recently, I’ve been working on three small ports. A table below describes some of the before-and-after program characteristics. These programs were written years ago and inherited by scientists who recognized their time would be better spent doing science than programming. Also, no programmer had ever reviewed the programs.

<table>
<thead>
<tr>
<th>Type</th>
<th>Lines</th>
<th>Character Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Livestock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>64</td>
<td>5665</td>
</tr>
<tr>
<td>Ported</td>
<td>60</td>
<td>4381</td>
</tr>
<tr>
<td>Furnaces</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>34</td>
<td>10394</td>
</tr>
<tr>
<td>Ported</td>
<td>34</td>
<td>7495</td>
</tr>
<tr>
<td>Elements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>6</td>
<td>279</td>
</tr>
<tr>
<td>Ported</td>
<td>6</td>
<td>307</td>
</tr>
</tbody>
</table>

Note: When I port a program, I do NOT change the algorithm that solves the problem, only how the algorithm is implemented.

The first program predicts economic parameters for up to six livestock herds.

Given herd parameters (number, sex, age, weight) predict births, deaths, and derivative products, i.e., the economic future of the herd.

The second program has to do with predicting radiation heat transfer and emissions (particles and chemical compounds) from power plant furnaces.

The third program has to do with the chemical elements.

The table below, compares programs before and after porting. Porting services like these are available from Lahey and costs vary widely based on the types of changes required for the code:

- More uniform style.
- Less code to read.

Some of the characteristics of these programs:
1) All had at least questionable programming practices, some of these practices could be classified as bugs.
2) Execution speed was improved.
3) Over the years, more than one scientist had worked on the code.
4) The ported programs were more readable for at least two reasons:
   a) More uniform style.
   b) Less code to read.

Regards,

Thomas M. Lahey

Bio-Mathematical Simulation of Ruminant Production Systems

This paper gives some of the history and rationale for developing and evolving a Fortran program that simulates herds of ruminants: cattle, sheep, and goats. It is difficult to make how complex these animals are: one example:

Some dairy cows can produce 20 gallons of milk, day after day. For each gallon of milk, a cow circulates 350 gallons of blood through her udder.

There is a great genetic diversity in livestock: a wide range of animals adapted to specific environments—from hothot milking machines, to cows that can live virtually on air in hot deserts. Nonetheless, what ultimately struck our numerical analysts derives their economic performance.

Many developing countries are characterized by harsh environmental conditions that adversely affect people and livestock. Scientists working to improve livestock production at low-cost organizational level understand the complex ways these animals interact with their environment. For example, they have to understand the economics of feeding in a poor African country with limited agricultural land resources. Only animals that can survive on crop aftermath and poor pastures, and are capable of traveling long distances to water can be substantially productive under such conditions.

Consider the following situation. The more milk a cow produces, the better she can nurse her calf. This means the calf will grow faster and obtain a better price in the market. However, if nutrition is limited, that cow will suffer a severe depletion of its body reserves. We know that undernourished cows have difficulties conceiving. As a consequence, cows with the higher milk potential may have a calf only once every other year in harsh environments. When both breeds of cows are compared (one with lower milk yield, and lighter but more calves, and one with more milk, and heavier but fewer calves), the economic returns do not become immediately apparent without making calculations across years and considering all animals in a herd. This means the decision that might appear to be obvious—to use higher producing animals—may actually worsen the situation of the farmers.

As you can see, there are complex interactions that are not amenable to paper and pencil. More than 20 years ago, scientists at Texas A&M University, under the leadership of Professor Thomas C. Cartwright, an internationally recognized cattle geneticist, tackled these problems. They developed a methodology for the quantitative description of the complex processes in cattle and other ruminant production systems. This methodology is based on a bio-mathematical model of the physiology of animals, and a model of the population dynamics occurring in herds.

Models are constructed to create a virtual herd, i.e., a computer program simulates all relevant processes and events that would take place in a real-world herd. Comparing program output with real-life data validates the model and simulated data can be analyzed with statistical and econometric methods. These models address decisions ranging from breeding plans (Which animal is best for a given environment?) to management decisions (When should animals be bred? When should an animal be sold? to receive investment decisions? Should feed be purchased? Should pastures be improved?).

Recently, after Dr. Cartwright’s 20 years of development, I decided to completely update the Fortran code he developed. I contacted with Lahey for Tony’s services as lead programmer. Our most important goals are to ensure numerical consistency of the simulation models and to create a program that facilitates the continued updating of mathematical equations based on new research findings on the biology of ruminant animals and how they interact with their environments.

We are using these models for the development of sustainable livestock production systems in Central Asia. In these countries, agriculture is faced with extreme environmental problems. Most of the territory of these countries can only be used for extensive, low-input livestock production. Our technology helps farmers to make better decisions regarding livestock production systems, and to improve the livelihood of farmers. The project is funded by USAID, the agricultural development fund of the United Nations.

Through our project with Lahey Computer Systems we have learned that the most efficient way of using advanced quantitative methods in science is close collaboration with professional software developers. This approach allows us to concentrate on our field, and ensures that we are using professional and dependable software. Based on our success to date, we plan to expand our collaboration with Lahey Computer Systems, Inc.

Wolfgang Pittroff
Assistant Professor of Range Animal Science, University of California, Davis.
E-mail: Wpittroff@ucdavis.edu

Essential LF90 Source Checker Now Available

By replacing the free Essential LF90 compiler Lahey used to give away to students, here’s a good alternative: the FREE ESSENTIAL LF90 Source Checker. While it won’t generate executable code, it will help you write good, clean, maintainable programs.

It’s easy to use. With your web browser, you can:
1) go to www.lahey.com/check.htm,
2) select the source file you want to process,
3) choose “Essential Lahey Fortran 90,” and
4) click submit.

The Essential LF90 Source Checker will check for adherence to the Essential Fortran language (see www.lahey.com/elf.htm and www.lahey.com/elf_spec.htm), detect argument mismatches across files (you can submit more than one file using a .ZIP archive), and generate source and cross-reference listings.

Remember, Essential Fortran is NOT a FORTRAN 77 compiler. It won’t compile standard FORTRAN 77 code. What is it then? It’s a clean, self-contained language with few redundancies, appropriate for teaching programming fundamentals. It’s not just for teaching Fortran, it’s for teaching fundamental programming concepts and for writing maintainable production programs.

Here’s what Essential Fortran does:
- Omits all obfuscating features of Fortran 90
- Example: Arithmetic IF and assigned GO TO
- Eliminates clutter: Example: The optional comma after DO
- Makes selections among redundant variants: Example: Entity-oriented vs. attribute-oriented declarations
- Imposes requirements where the full Fortran 90 allows exceptions: Example: IMPLICIT NONE in every main program or module
- Prevents overloading declarations: Example: COMMON, EQUIVALENCE, and implicit procedure interface

The Essential Fortran language does retain statement labels for the sake of input-output exception handling and DO WHILE in recognition of its popularity in some quarters. You can build an Essential Fortran language-compliant program with any standard-conforming Fortran 90 or 95 language system.

Lahey/PTac Fortran 95 (LF95) and Lahey Fortran 90 (LF90) Source Checkers are still available at www.lahey.com/check.htm.
Q: Why are LF95 Windows executables larger than LF90 executables?
A: LF90 programs depend on system DLLs or DOS interrupts to accomplish many runtime tasks, which means a much smaller executable. LF95 for Windows internalizes much of this runtime code, which is why its executables are relatively large. Were this not the case a third party install could replace a “good” DLL with an incompatible DLL, which could cause the executable to “mysteriously break.”

Q: I have a program that is working correctly when compiled with LF90. Will I have any problems building it with LF95?
A: The chances are overwhelming that you will be able to build your program using LF95 and that it will run without problems. That being said, here are some situations that may cause differences in how the program will execute:

One possibility is that the program uses units 5 and/or 6 for file I/O. Unlike LF90 or EM32, LF95 pre-connects unit 5 for reading from stdin and unit 6 for writing to stdout. These unit numbers should not be re-assigned or reused for any other purpose. Another possibility is that your program has undetected array bounds issues or other memory access problems. LF95 does a much better job of checking for runtime-error conditions than other compilers (including LF90) and is more sensitive to such memory access problems. LF95 does a much better job of checking for runtime-error conditions than other compilers (including LF90) and is more sensitive to such memory access problems.

Q: How do I capture compiler output in a text file?
A: In Windows, output can be captured from any program that sends text to stdout (the console) by using the redirection operator at the end of the command line: “ > file.ext”, where file.ext is the name of the file you wish to contain the compile results. In Linux the same is true with one exception. In Linux, LF95 compiler output is by default written to stderr. To redirect output written to stderr, you need to add the ‘&’ character, i.e.: >file.ext &. In both Windows and Linux, compiler output can be appended to an existing file use ‘>>’ instead of ‘>’.

Q: Where can I find LF95 source code examples?
A: Assuming you installed LF95 v.5.6 in the standard directory, you will find an examples directory at \lf95\examples. There are several directories with examples of mixed language compiling, Wintester/programming, Automake, SN22 programming, C, f90gl, and f90qgl.

Q: What is involved with porting code to Linux (LF95 v.6.1) from Windows?
A: If your code is F77, F90, or F95 standard conforming, it will port to Linux simply by recompiling. If you are using the Wintester or GINO GUI libraries, you can recompile your code and link with the Linux version of these libraries without making any changes. If you are using Automake, the basic structure of the automake.fig configuration file will remain the same. If any code or data contains path information, you will have to change the Windows directory separator ‘\’ to the Unix separator ‘/’ and make sure that pertinent files are in the indicated directories. If you use environment variables, you will need to convert from Windows style “\var1” to Unix style “/var1”. Many non-standard extensions are supported under both the Windows and Linux environments. If an extension is not supported, it will most likely cause an “undefined symbol” error when linking. If your code uses the “system” subroutine, you should consult your LF95 Language Reference. Although the basic form of the command is supported under both systems, optional arguments are not supported on the Linux side.

Rent LF95 and a Beowulf Cluster
Lahey and Tsunami Technologies, Inc. (TTI) are pleased to announce the availability of Lahey/Fujitsu Fortran 95 (LF95) on an eight-node, 1.53 GHz Athlon Beowulf cluster for remote rental. If you’ve ever wanted to try parallel computing or simply want to try LF95 for Linux, now’s the time.

What’s available?
• LF95 v.6.1 compiler, runtime, and debugger for developing commercial or educational applications.
• Eight 1.33 GHz Athlon computers configured as a Beowulf cluster.
• 512 MB DDRAM per node.
• Linux kernel 2.4.5.
• MPICH fully integrated with the Portable Batch System, including security.
• 1 GB of storage.
• Secure work environment.
• And, as always from Lahey and TTI – quality service.

How does it work?
Set up a free account with Tsunami Technologies and upload your source code (or existing parallel or serial LF95 program) to a TTI server. Next, through a secure telnet connection, use LF95 to build your application. Then, submit your application to the queueing system with a simple command. Your job is securely placed on one or more cluster nodes for execution. Finally, the standard output of your program is transferred to your secure personal account directory.

If you’ve never used clustering technology before or are new to Linux, TTI can help. Their staff will set up your account directories, assist with compiling your source code, and help you create the scripts you’ll need to get started. LF95 and MPICH documentation are available on your personal account space.

Why try it?
It’s a great deal!
Where else can you have access to a Linux cluster for $1.95 per CPU per hour?

Evaluate LF95 for Linux
Before you buy, see how LF95 stacks up to your current Linux Fortran solution. Try it on a single node if you don’t want to create a parallel application.

MPICH is ready to go.
No need to configure a message passing interface library. MPICH is installed, built, and ready for use with LF95.

Don’t know if parallel computing will help your applications?
Try it without buying your own cluster.

On a budget?
Avoid the costs of planning, purchasing, building, and maintaining a cluster. TTI does all the system administration and technical support. There’s a fully maintained Beowulf cluster at your fingertips.

Between projects?
Unlike an on-site cluster that incurs costs even when not in use, with TTI’s on-demand services you pay only for the CPU time LF95 and your applications use.

Behind schedule?
TTI can provide the extra computing power you need to meet your deadlines. Their responsive technical support gets you up and running quickly. You get on-demand computing power when you need it.

Already have a cluster?
You can still take advantage of TTI’s low-cost computing and significantly increase your productivity without capital investment. Supplement and integrate your computing power with TTI’s convenient remote access clusters.

What does it cost?
Commercial use is US$3.09 per CPU per hour. Educational use is US$0.79 per CPU per hour. And, it’s FREE for the first 7 days. Your free days begin when you submit your first job to the queue. Use this time to test or benchmark your applications or run our demo programs.

Give it a try today.
LF95 and a Linux Beowulf cluster are only as far as your Internet-connected computer.

Setup an account online at www.onlinecluster.com or through the TTI link at www.lahey.com or call 1-877-492-8027.

TTI specializes in online access to computing clusters for computationally intensive applications. Lahey, of course, specializes in Fortran. TTI can be reached at www.onlinecluster.com, Lahey, at www.lahey.com.
News Briefs

Code Repository

Some of our key technologies and tools are now available to a broader audience. Our new web site offers free download of the latest version of GINOMENU Studio. It includes the data access type and the semantic information about the data that will be used repeatedly in the immediate future. The second type, temporal data, contains information that is particularly important for applications that share critical data, such as financial transactions or medical records. Spatial data can be made up of either temporal or non-temporal data types. Depending on the data type, different optimization techniques will be used to reduce data storage and improve performance. For example, data types that are stored in cache can be accessed faster than those stored in RAM. The performance of a program can be improved by using the prefetch optimization.

What is Prefetch?

Editor’s note: Dave Berry is the technical manager of our Linux products. He wrote this article in conjunction with the release of our new Linux product (see page 1), which is the first Lahey product to contain the prefetch optimization.

The performance of many programs can be improved considerably if certain data can be loaded into a cache before it is needed by the processor. Prefetch instructions allow this to happen and reduce the time required to process the data already loaded into the cache. The cache memory is a memory region that can access faster than the CPU waits for data to move from RAM to the cache. The prefetch instructions do not change the user-visible semantics of a program. They merely provide information to the hardware on how to use memory more efficiently.

The relevant information the prefetch instruction provides includes the data access type and the specific cache level. There are three data access types, and each data type has a specific purpose:

- Temporal data contains information that is accessed frequently within the immediate future. The second type, temporal data, contains information that is particularly important for applications that share critical data, such as financial transactions or medical records. Spatial data can be made up of either temporal or non-temporal data types. Depending on the data type, different optimization techniques will be used to reduce data storage and improve performance.

- Prefetch instructions can reduce the overhead of memory manipulations by using the cache and memory efficiently and by reducing the amount of data that needs to be transferred between cache and RAM. This is particularly important for applications that share critical system resources, such as the memory bus. For some source code, the compiler automatically provides a prefetch instruction with advance information about the data to be prefetched. In general, prefetch instructions can improve the performance of a program if the program:

  - Reads data in a predictable manner
  - Contains memory flow bottlenecks

Prefetch instructions can reduce the overhead of memory manipulations by using the cache and memory efficiently and by reducing the amount of data that needs to be transferred between cache and RAM. This is particularly important for applications that share critical system resources, such as the memory bus. For some source code, the compiler automatically provides a prefetch instruction with advance information about the data to be prefetched. In general, prefetch instructions can improve the performance of a program if the program:

- Reads data in a predictable manner
- Contains memory flow bottlenecks

Get the latest information on GINOMENU Studio v3.0 and other Lahey products by subscribing to our information e-mail list. We will send regular updates about GINOMENU Studio v3.0 to all subscribers who have either a warm memory or a war story.

GINOMENU Studio v3.0 is now available. Please visit our web site for more information. Currently, we are featuring a preview of new features in GINOMENU Studio v3.0. We look forward to promised .Net-catalyzed efforts at improvement.

By Kevin Bradly, Bradly Associates

Starting in January, you can learn more about GINOMENU Studio v3.0 by calling 1-800-548-4778 to get your copy. Our staff will recommend changes for clarity, speed, or for other reasons.

Please visit our web site for more information. Currently, we are featuring a preview of new features in GINOMENU Studio v3.0. We look forward to promised .Net-catalyzed efforts at improvement.

By Kevin Bradly, Bradly Associates
News Briefs

Code Repository

Some of our leading software engineers have worked with a FORTRAN code repository on our web site. The code repository is a new area on our web site where users can share FORTRAN code and programming techniques. Through collaboration with customers we designed the code repository with the following features in mind:

Features:
- Code will be reviewed by our staff before it is posted.
- Our staff will recommend changes for clarity, speed, or for other reasons.
- Code in the repository can be viewed by anyone visiting the web-site.

Conditions of Use

• All donations become public domain.
• Author will get any comments Lahey gets.
• Need test driver/(a/e) to validate.
• Need documentation as a header.
• Lahey reserves right to modify – hopefully with the intent of improving.

Host & use the repository

• Code in the repository can be viewed at www.lahey.com.
• Submit code by e-mailing it to code@lahey.com.

WinterCate v4.0 is Now Available

WinterCate is a modern GUI toolkit for the Fortran 90/95 programming language. It consists of various visual development tools and a substantial subroutine library.

What’s New in v4.0?

• New features include:
- A new combined-resource editor lies at the core of WinterCate v4.0. The new editor combines the functionality of the four previously separate dialog, menu, icon/user, and toolbar editors. The new editor greatly streamlines GUI design and management of the associated resources. Navigation of your dialogs and menus is now much easier with the entire contents of the program resource files accessible via a single treewview. There is no need to contain your resource files into multiple tools. The combined editor also provides greater consistency of behavior when editing different resource types and features numerous minor enhancements.

• Graphics Text
WinterCate’s graphic text handling has undergone a major redesign in v4.0, providing a much more concise calling interface and additional functionality. While the old calling interfaces remained supported, most programs will benefit from the new interface. New features include access to any Windows font in GD graphics output to screen, bitmap, printer, or metafile. By reducing the amount of default text values is now supported, as is direct output of numeric values.

• Graphics Export/Import
WinterCate’s graphics output and import capabilities have been further expanded in v4.0. The CGM importer has been upgraded substantially to read a much wider range of third party metatiles. BMP, PCX, and PICT bit image files loaded via ReadImage are now reproduced in graphics output via the Print Manager. Windows, metafile, and X windows image header files are also supported. Version 4.0 generates smaller CGM and DXF files too.

Operating System Interface

New operating system interface features include the following: generation of temporary file/directory names, entities to set or get

Mix Visual Basic and Fortran easily with j90VB!
j90VB is a set of four Fortran libraries that provide the functionality necessary for your Fortran programs to gain access to COM and ActiveX objects. In addition, you can use j90VB to call Visual Basic DLLs from your Fortran applications, or to create Fortran DLLs that are able to handle standard Visual Basic data types (VB strings, arrays and variants). j90VB also includes TLView, a professional quality TLB browser that can be used as a tool to explore COM and ActiveX objects you use from your Fortran applications.

j90VB includes five main components:
- j90AutomationLibrary: lets you handle COM, ActiveX and Visual Basic’s string types from Fortran.
- j90SaferyGenLibrary: gives you access to COM, ActiveX and Visual Basic’s array structures and functions.
- j90VariantsLibrary: Allows you to manipulate COM, ActiveX and Visual Basic’s variants from Fortran.
- j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic.
- j90AutomationLibrary: lets you build user interfaces for your Fortran programs.

For more information about this innovative Microsoft initiative and it includes links to sites with even more information. Currently, we are featuring a technology preview for Fortran.NET including examples of new .NET programming, Visual Studio integration, new object-oriented features for Fortran and more.

What is it?

• j90VB allows you to create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

• j90AutomationLibrary: lets you create Automation controllers from Fortran. Fully manipulate Excel spreadsheets or Internet Explorer from your Fortran applications as easily as with Visual Basic. By using j90VB you can create applications that fully interact with Visual Basic DLLs – you can use Visual Basic from your Fortran program.

GIFNEMU Studio v3.0 to be Released

By Kevin Bradly, Bradly Associates

GIFNEMU Studio v3.0 is now available. GIFNEMU Studio v3.0 is a serious Rapid Application Development tool for the Fortran developer and includes a whole host of new facilities such as:

• New look and feel
• Automatically generate C++ code as well as F77 & F90
• $399 for educational use. Visit www.lahey.com or give Lahey a call at 1-800-548-4778 to get your copy.

Fortran.NET

• What is it?
• Why does it matter?

.NET is Microsoft’s new infrastructure solution for programming the Internet and for sharing information via networked systems. It is designed to help solve many common programming problems including:
- Inter-language programming
- Hardware upgrades
- Operating system organization
- Application upgrades
- Memory management
- Data sharing between networked applications

Lahey and Fujitsu have partnered with Microsoft to bring .NET to Fortran users all over the world, and we want to keep you informed about our progress. In order to do so, we have set up an information e-mail list. We will send regular updates about Fortran.NET, our schedules for beta and release, NET seminars and more, and you can join by visiting our web-site or by sending an e-mail as follows:

Send e-mail to interfer@lahey.com, with only the words, “subscribe,fortan.net@ unserenmeer” on the first line of the body.

Peter Coffee’s eWEEK Magazine made the following statements about .NET:

“Just call me a kid in a candy store. Nothing gets my brain revved up like learning a new way of thinking in a novel programming language, and Microsoft’s .NET platform is revising research and implementation efforts in both legacy and leading-edge languages. I have almost any language you might name; I have either a warm memory or a war story – and a reason to look forward to promised net catalyzed efforts at improvement... Lahey and Fujitsu are now collaborating to bring FORTRAN as well to .NET.”

Peter’s entire article is available at http://techupdate.zdnet.com/techupdate/stories/mn/01/14/792789800.html.

Lahey’s website now includes a section dedicated to .NET and Fortran.NET. This site is updated regularly with new information about this innovative Microsoft initiative and it includes links to sites with even more information. Currently, we are featuring a technology preview for Fortran.NET including examples of new .NET programming, Visual Studio integration, new object-oriented features for Fortran and more.

We look forward to hearing what our customers think about .NET and how we can best serve your Fortran needs.

GIFNEMU Studio v3.0 is now available. GIFNEMU Studio v3.0 is a serious Rapid Application Development tool for the Fortran developer and includes a whole host of new facilities such as:

• New look and feel
• Automatically generate C++ code as well as F77 & F90
• $399 for educational use. Visit www.lahey.com or give Lahey a call at 1-800-548-4778 to get your copy.
Q: Why are LF95 Windows executables larger than LF90 executables?
A: LF95 programs depend on system DLLs or DOS interrupts to accomplish many runtime tasks, which means a much smaller executable. LF95 for Windows internalizes much of this runtime code, which is why its executables are relatively large. Were this not the case a third party install could replace a “good” DLL with an incompatible DLL, which could cause the executable to “mysteriously break.”
Q: I have a program that is working correctly when compiled with LF90. Will I have any problems building it with LF95 compiler output by is default written to stdout? To redirect output written to stderr, you need to add the ‘&’ character, i.e., “>file.ext”.
A: The chances are overwhelming that you will be able to build your program using LF95 and that it will run without problems. That being said, there are some situations that may cause differences in how the program will execute.

One possibility is that the program uses units 5 and/or 6 for file I/O. Unlike LF90 or EM32, LF95 pre-connects units 5 for reading from stdin and unit 6 for writing to stdout. These unit numbers should not be re-used or realigned for any other purpose. Another possibility is that your program has undetected array bounds issues or other memory access errors. LF95 does a much better job of checking for runtime-error conditions than other compilers (including LF90) and is more sensitive to such problems. Programs with problems like these might run with apparent success for years on a given platform/compiler combination, but fail miserably when moved to a different compiler or computing platform. Such problems may get ‘illegal operation’ messages when run after compiling with LF95. Compiling your code with the -chk or -chkglobal option will allow you to quickly find where the problem occurs and give you the opportunity to flush these bugs out of your code once and for all.

Q: How do I do a compile captured output in a text file?
A: In Windows, output can be captured from any program that sends text to stdout (the console) by using the redirection operator at the end of the command line: “> file.ext”, where file.ext is the name of the file you wish to contain the compile results. In Linux the same is true with one exception. In Linux, LF95 compiler output is by default written to stdout. To redirect output written to stderr, you need to add the ‘&’ character, i.e., “>file.ext”. In both Windows and Linux, compiler output can be appended to an existing file (using “>>”) instead of “>”.

Q: Where can I find LF95 source code examples?
A: Assuming you installed LF95 v5.6 in the standard directory, you will find an examples directory at \examples. There are several directories with examples of mixed language compiling, Wndowstpc programming, Automake, UNIX2 programming, C, F90GL, and F90QA.

Q: What is involved with porting code to Linux (LF95 v6.1) from Windows?
A: If your code is F77, F90, or F95 standard conforming, it will port to Linux simply by recompiling. If you are using the Wndowstpc or GNOI GUI libraries, you can recompile your code and link with the Linux version of these libraries without making any changes. If you are using Automake, the basic structure of the automake fig configuration file will remain the same. If any code or data contains path information, you will have to change the Windows directory separator “\” to the Unix separator “/”. Make sure that pertinent files are in the indicated directories. If you use environment variables, you will need to convert from Windows style “%var%” to Unix style “$var”.

Many non-standard extensions are supported under both the Windows and Linux environments. If an extension is not supported, it will most likely cause an “undefined symbol” error when linking. If your code uses the “system” subroutine, you should consult your LF95 Language Reference. Although the basic form of the command is supported under both systems, optional arguments are not supported on the Linux side.

Rent LF95 and a Beowulf Cluster
Lahey and Tsunami Technologies, Inc. (TTI) are pleased to announce the availability of Lahey/Fujitsu Fortran 95 (LF95) on an eight-node, 1.53 GHz Athlon Beowulf cluster for remote rental. If you’ve ever wanted to try parallel computing or simply want to try LF95 for Linux, now’s the time.

What’s available?
• LF95 v6.1 compiler, runtime, and debugger for developing commercial or educational applications.
• Eight 1.53 GHz Athlon computers configured as a Beowulf cluster.
• 512 MB DDRAM per node.
• Linux kernel 2.4.5.
• MPICH fully integrated with the Portable Batch System, including security.
• 1 GB of storage.
• Secure work environment.
• And, as always from Lahey and TTI – quality service.

How does it work?
Set up a free account with Tsunami Technologies and upload your source code (or existing parallel or serial LF95 program) to a TTI server. Next, through a secure telnet connection, use LF95 to build your application. Then, submit your application to the queueing system with a simple command. Your job is securely placed on one or more cluster nodes for execution. Finally, the standard output from your program is transferred to your secure personal account directory.

If you’ve never used clustering technology before or are new to Linux, TTI can help. Their staff will set up your account directories, assist with compiling your source code, and help you create the scripts you’ll need to get started. LF95 and MPICH documentation is available as man pages on the cluster. Complete LF95 documentation is available at www.lahey.com.

Why try it?
It’s a great deal!
Where else can you have access to a Linux cluster for $1.09 per CPU per hour?

Evaluate LF95 for Linux
Before you buy, see how LF95 stacks up to your current Linux Fortran solution. Try it on a single node if you don’t want to create a parallel application.

MPICH is ready to go.
No need to configure a message passing interface library. MPICH is installed, built, and ready for use with LF95.

Don’t know if parallel computing will help your application?
Try it without buying your own cluster.

On a budget?
Avoid the costs of planning, purchasing, building, and maintaining a cluster. TTI does all the system administration and technical support. There’s a fully maintained Beowulf cluster at your fingertips.

Between projects?
Unlike an on-site cluster that incurs costs even when not in use, with TTI’s on-demand services you pay only for the CPU time LF95 and your applications use.

Behind schedule?
TTI can provide the extra computing power you need to meet your deadlines. Their responsive technical support gets you up and running quickly. You get on-demand computing power when you need it.

Already have a cluster?
You can still take advantage of TTI’s low-cost computing and significantly increase your productivity without capital investment. Supplement and integrate your computing power with TTI’s convenient remote access clusters.

What does it cost?
Commercial use is US$0.09 per CPU per hour. Educational use is US$0.79 per CPU per hour. And, it’s FREE for the first 7 days. Your free days begin when you submit your first job to the queue. Use this time to test or benchmark your applications or run your demo programs.

Give it a try today.
LF95 and a Linux Beowulf cluster are only as far as your Internet-connected computer.

Set up an account online at www.onlinecluster.com or through the TTI link at www.lahey.com or call 1-877-492-8027.

TTI specializes in online access to computing clusters for computationally intensive applications. Lahey, of course, specializes in Fortran. TTI can be reached at www.onlinecluster.com, Lahey, or www.lahey.com.
Dear Fortran Programmers,

As Lahey’s response to the World Trade Center attacks, we have been contacting firms with registered copies of Lahey products in the New York City area to see how best we can help them recover. If we missed your organization, please give us a call.

On our web-site, there’s a tutorial on threading derived-type objects called “Putting Mega-Byte Memory to Work.” Lahey will continue to publish articles on Modern Fortran subjects both in Fortran SOURCE and on our web-site www.lahey.com. So far, there are also pages on style and the Flynn Price array-expressions articles that appeared in Fortran SOURCE years ago. If you have a topic you’d like to see, send email to tlahe@lahey.com.

Recently, I’ve been working on three small ports. A table below describes some of the before-and-after program characteristics. These programs were written years ago and inherited by scientists who recognized their time would be better spent doing science than programming. Also, no programmer had ever reviewed the programs.

**Note:** When I port a program, I do NOT change the algorithm that solves the problem, only how the algorithm is implemented. The first program predicts economic parameters for up to six livestock herds.

<table>
<thead>
<tr>
<th>Program Units</th>
<th>Lines</th>
<th>Character Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Livestock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>64</td>
<td>5,665</td>
</tr>
<tr>
<td>Ported</td>
<td>60</td>
<td>4,361</td>
</tr>
<tr>
<td>Furnaces</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>34</td>
<td>10,094</td>
</tr>
<tr>
<td>Ported</td>
<td>34</td>
<td>7,495</td>
</tr>
<tr>
<td>Elements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>6</td>
<td>279</td>
</tr>
<tr>
<td>Ported</td>
<td>6</td>
<td>307</td>
</tr>
</tbody>
</table>

Ported 6 307 8,999

Ported 34 7,495 284,023

Livestock 34  10,094  284,436

Original 64  5,665  214,240

<table>
<thead>
<tr>
<th>Program Units</th>
<th>Lines</th>
<th>Character Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Livestock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original</td>
<td>34</td>
<td>258,436</td>
</tr>
<tr>
<td>Ported</td>
<td>34</td>
<td>200,323</td>
</tr>
</tbody>
</table>

Porridge 10 307 200,000

Porridge 34 7,495 284,023

For each line of milk, a cow circulates 500 gallons of blood through her udder.

Bio-Mathematical Simulation of Ruminant Production Systems

This paper gives some of the history and rationale for developing and evolving a Fortran program that simulates herds of ruminants: cattle, sheep, and goats. It is difficult to make how complex these animals are: one example.

Some dairy cows can produce 20 gallons of milk, day after day. For each barrel of milk, a cow circulates 500 gallons of blood through her udder.

There is a great genetic diversity in livestock: a wide range of animals adapted to specific environments—from leech milking machines, to cows that can live virtually on air in hot deserts. Note, however, that ultimately these nutritional regimes determine their economic performance.

Many developing countries are characterized by harsh environmental conditions that adversely affect people and livestock. Scientists working to improve livestock production have modeled the complex ways these animals interact with their environment. For example, they have to understand the economics of feeding in a poor African country with limited agricultural land resources. Only animals that can survive on crop aftermath and poor pastures, and are capable of traveling long distances to water can be substantially productive under such conditions.

Consider the following situation. The more milk a cow produces, the better she can nurse her calf. This means the calf will grow faster and obtain a better price in the market. However, if nutrition is limited, that cow will suffer a severe depletion of its body reserves. We know that undernourished cows have difficulties conceiving. As a consequence, cows with the higher milk potential may have a calf only once every other year in harsh environments. When two breeds of cows are compared (one with lower milk yield, and lighter but more calves, and one with more milk, and heavier but fewer calves), the economic returns do not become immediately apparent without making calculations across various years and considering all animals in a herd. This means the decision that might appear to be obvious – to use higher producing animals – may actually worsen the situation of the farmers.

As you can see, there are complex interactions that are not amenable to paper and pencil. More than 20 years ago, scientists at Texas A&M University, under the leadership of Professor Thomas C. Cartwright, an internationally recognized cattle geneticist, tackled these problems. They developed a methodology for the quantitative description of the economic processes in cattle and other ruminant production systems. This methodology is based on a bio-mathematical model of the physiology of animals, and a model of the population dynamics occurring in herds.

Models are constructed to create a virtual herd, i.e., a computer program simulates all relevant processes and events that would take place in a real-world herd. Comparing program output with real-life data validates the model and simulated data can be analyzed with statistical and econometric methods. These models address decisions ranging from basic plans (Which animal is best for a given environment?) to management decisions (When should animals be bred? When should an animal be sold to finance investment decisions? Should feed be purchased? Should partners be improved?).

Recently, two years after Dr. Cartwright’s 20 years of development, I decided to completely update the Fortran code he developed. I contacted with Lahey for Tom’s services as lead programmer. Our most important goals are to ensure numerical consistency of the simulation models and to create a program that facilitates the continued updating of mathematical equations based on new research findings on the biology of ruminant animals and how they interact with their environment.

We are using these models for the development of sustainable livestock production systems in Central Asia. In these countries, agriculture is faced with extreme environmental problems. Most of the territory of these countries can only be used for extensive, low-input livestock production. Our technology helps find solutions that minimize environmental impact of livestock production while improving the livelihood of farmers. The project is funded by USAID, the agricultural development fund of the United Nations.

Through our project with Lahey Computer Systems we have learned that the most efficient way of using advanced mathematical techniques is in close cooperation with professional software developers. This approach allows us to concentrate on our field, and ensures that we are using professional and dependable software. Based on our success to date, we plan to expand our collaboration with Lahey Computer Systems, Inc.

Wolfgang Pittroff
Assistant Professor of Range Animal Science, University of California, Davis.
E-mail: Wpittroff@ucdavis.edu

Regards,

Thomas M. Lahey

Some of the characteristics of these programs:

1) All had at least questionable programming practices; some of these practices could be classified as bugs.
2) Execution speed was improved.
3) Over the years, more than one scientist had worked on the code.
4) The ported programs were more readable for at least two reasons:
   a) More uniform style.
   b) Less code to read.

Essential LF90 Source Checker is Now Available

Being missing that free Essential LF90 compiler? Lahey used to give away! So have others. Here’s a good alternative: the FREE Essential LF90 Source Checker. While it won’t generate executable code, it will help you write good, clean, maintainable programs.

It’s easy to use. With your web browser,

1. go to www.lahey.com/check.htm,
2. select the source file you want to process,
3. choose “Essential Lahey Fortran 90,” and
4. click submit.

The Essential LF90 Source Checker will check for adherence to the Essential Fortran language (see www.lahey.com/ef.htm and www.lahey.com/ef_source.htm). detect argument mismatches across files (you can submit more than one file using a ZIP archive) and generate source and cross-reference listings.

Remember, Essential Fortran is NOT a FORTRAN 77 compiler. It won’t compile standard FORTRAN 77 code. What is it then? It’s a clean, self-contained language with few redundancies, appropriate both for teaching programming fundamentals. It’s not just for teaching Fortran, it’s teaching fundamental programming concepts and for writing maintainable production programs.

Here’s what Essential Fortran does:

- Omits all obfuscating features of Fortran 90
- Example: Arithmetic IF and assigned GO TO
- Optimizes clarity
- Example: The optional comma after DO
- Makes selections among redundant variants
- Example: Entity-oriented vs. attribute-oriented declarations
- Imposes requirements where the full Fortran 90 language
- Example: IMPOR
t NONE in every main program or module
- Provides storage association
- Example: COMMON, EQUIVALENCE, and implicit procedure interface

The Essential Fortran language does retain statement labels for the sake of input-output exception handling and DO WHILE in recognition of its popularity in some quarters. You can build an Essential Fortran language-compliant program with any standard conforming Fortran 90 or 95 language system.

Lahey/Tomcat Fortran 95 (LPFS) and Lahey Fortran 90 (LF90) Source Checkers are still available at www.lahey.com/check.htm.
LF95 Linux v6.1 Released!

Features of LF95 Linux PRO
- Auto-parallelization for multi-processor machines.
- OpenMP support.
- Optimized BLAS (Basic Linear Algebra Subprograms).
- LAPACK (Linear Algebra PACKage) libraries callable from parallelized FORTRAN programs.
- Thread-safe SSL2 math library callable from parallelized Fortran programs.
- Windsornet Starter Kit, WSK, a subset of Windsornet by Interactive Software Services, Ltd. WSK is a portable non-commercial interface and graphics package for Fortran 90/95 developers.
- Automake, by Polyhedron Software, a multi-utility that automatically determines program dependencies.
- Hardware manuals.
- Unlimited Unix ‘Callback’ phone support.

Features of both PRO and Express
- NEW! Prefetch optimization for Pentium III and Athlon processors.
- NEW! Run your applications on Linux clusters at no extra charge.

Lahey offers competitive floating and site-license pricing.
}

LP95 Linux v6.1 was released on November 28, 2001. It features a performance enhancement called prefetch for Pentium III and Athlon processors. Almost all benchmark tests have shown a performance improvement with LF95 v6.1 with prefetch. Beta testers and benchmark tests have seen performance improvements by about 10%, with a small number of tests showing up to 25% improvement and a small number showing no improvement or slight performance degradation. Try LF95 v6.1 with your code today.

LP95 v6.1 also includes numerous enhancements and features to make programming easier. Lahey’s industry-leading compiler and runtime diagnostics now include column position error messages and improved runtime speed when checking subsrgrams. Moving code between LP95 for Windows and Linux and LF95 Linux v6.1 is easy because most Windows, Workstation, GINO, and Automake code will port with no changes. Multiple compilers are not required as LF95 supports your Fortran 95, 90, and 77 code.

Both PRO and Express CDs include source and instructions for the MPICH message passing library to get your Linux cluster up and running.

Call today with your order and questions. 800-546-4778 or visit www.lahey.com.

FREE technical support.

Lahey’s Fortran SOURCE newsletter – Your source for the latest news from Lahey.

FD95 Linux v6.1 Released! PRO and Express Editions Available

Features of FD95 Linux PRO
- Auto-parallelization for multi-processor machines.
- OpenMP support.
- Optimized BLAS (Basic Linear Algebra Subprograms) and LAPACK (Linear Algebra PACKage) libraries callable from parallelized FORTRAN programs.
- Supporting SGI’s math library callable from parallelized FORTRAN programs.
- Windsornet Starter Kit, WSK, a subset of Windsornet by Interactive Software Services, Ltd. WSK is a portable non-commercial interface and graphics package for Fortran 90/95 developers.
- Automake, by Polyhedron Software, a multi-utility that automatically determines program dependencies.
- Hardware manuals.
- Unlimited Unix ‘Callback’ phone support.

Features of both PRO and Express
- NEW! Prefetch optimization for Pentium III and Athlon processors.
- NEW! Run your applications on Linux clusters at no extra charge.

Lahey offers competitive floating and site-license pricing.

To learn more about our Linux, Windows, and other Fortran products, please call Lahey Sales at 800-546-4778 or 775-831-2500 or visit www.lahey.com.

Lahey’s Fortran SOURCE newsletter – Your source for the latest news from Lahey.