About a FOSS for the Teaching of FORTRAN Programming Language Course

Afaq Ahmad[1], Tariq Jamil[1], Saad Ahmad[2] and Amer Arshad Abdulghani[1]

***Abstract*— The FORTRAN programming language has its revival phase. Majority of the science colleges of the world universities are teaching FORTRAN programming as a primary language. At Sultan Qaboos University, FORTRAN programming language is taught using the Free Open Source Software (GFORTRAN compiler with Geany editor) since 2009. We share our knowledge about this Free Open Source Software for the benefits of scientists and engineers.**

***Index Terms*— FORTRAN, Geany, GFORTRAN, High Level Language, Scientific Language, Support file-types, Plugins.**

LII. INTRODUCTION

FREE Open Source Software (FOSS) drive a free software is program that provides users a certain levels of essential freedoms. Those freedoms are of kinds as:

- The freedom to run the program,
- The freedom to study how the program works and
- The freedom to redistribute copies of the programs as it is or in the modified forms.

Typical free software licenses such as the GPL, LGPL, Mozilla, Apache, MIT-style and BSD-style licenses grant these above described freedoms [1].

If a program fails to serve with these freedoms is termed as non-free program. Non-free program approach is unethical because it makes such program an instrument of unjust power. The issue of Free Software is not just a technical issue; it is an ethical, social, and political issue. It is related to an indirect question of the human rights that the users of software ought to have [1]. It is also to be reminded wit ourselves that freedom and cooperation are essential values of Free Software [1]. The freedom provided by Free Open Software Source plays a fundamental role in education in the forms of enhancement of learning processes and in obtaining fruitful research results. Hence to disseminate human knowledge and to prepare students to be good members of their community we should use and teach Free Software in educational institutions of all levels. The Free Software is the only software that allows them to accomplish their essential missions of achieving goals of education. The source code and the methods of Free Software are part of human knowledge. On the contrary, proprietary software is secret, restricted knowledge, which is the opposite of the mission of educational institutions. Free Software supports education, proprietary software forbids education [1] – [2].

To this direction we moved towards FOSS and developed many laboratory experiments [3] – [22] along with teaching of FORTRAN language programming.

LIII. FORTRAN LANGUAGE HISTORICAL VIEW

FORTRAN (Formula Translation) was originally developed in the 1950s, is one of the most popular languages in the field of numerical and scientific computation, and especially in High Performance Computing (HPC). Among the earliest High Level Languages (HLL) FORTRAN played a key role in computation, and is still in widespread use today. FORTRAN 66 released in 1966 was the first of all programming language standards. Hence, after the release of the first standard the following rapid developments came in existence.

- Second standard was released in 1978 as FORTRAN 77 and
- Third standard was released in 1991 as FORTRAN 90, by adding new, modern features such as structured constructs, array syntax and ADT.

FORTRAN 90 was extended and revised in 1997 and commercialized as FORTRAN 95. The release version of FORTRAN 95 was further, extended with published report in 2004, and marked as FORTRAN 2003, existed as 4th standard of FORTRAN language. Major revision, incorporates TRs, adds many new features like Object Oriented (OO) programming. The fifth standard of FORTRAN was released in 2010 and available as FORTRAN 2008. For more information about several versions of the FORTRAN language, interested readers can see reference [1], [23] – [25].

Although many versions of FORTRAN but FORTRAN 90 was a significant revision to the original standard. Newer versions do exist (FORTRAN 95/2003/2008), but they generally build upon the FORTRAN 90 core.

LIV. REVIVAL OF FORTRAN LANGUAGE

Considered the Latin of the computer programming world, FORTRAN, a workhorse of the last century, is today considered to be a forgotten language which is almost dead. An attempt made in 2003 to revive this workhorse by approving FORTRAN 2003 standard which incorporates features of object-oriented programming, IEEE 754 standard for floating point arithmetic, and inter-operation with the C

[1]Department of Electrical and Computer Engineering, College of Engineering, Sultan Qaboos University, P.O. 33, Al-Khodh, Muscat -123, Sultanate of Oman; (phone: 968-24141327; fax: 968-24413454; e-mail: afaq@squ.edu.om)

[2]Department of Electrical and Computer Engineering, Cockrell School of Engineering, The University of Texas at Austin, Austin, TX 78712, United States; (e-mail: ahmad8@utexas.edu)

programming language, has given new life to this beast. It is being used today as a tool of choice for carrying out very large numerical computations, such as simulating atomic bomb explosions, weather prediction, flight simulations, DNA proteins simulations, and electron-orbiting simulations. Programmers call FORTRAN a horrific language but, in the face of no other language having optimized compilers to suit computational tasks, the availability of vast sub-routines library of this language containing optimized code makes it attractive to the scientists [26], [27]. A table detailing the features of available FORTRAN compilers can be found at [28]. A more modern version of FORTRAN, approved in 2008, includes features for parallel programming and new capabilities are being added constantly to keep it up-to-date with the features found in other popular high-level programming languages and as demanded by the programmers worldwide [29].

Despite the fact that it is journey of over 63 years and serving professionally over more than 40 years, FORTRAN is still taught at many Universities to students of Physics, Chemistry, Engineering and more as their first ever formal introduction to programming in spite of existence of much better alternatives, such as C++, Java, Python that would serve a physical science undergraduate much better. Yes, it is true that at a stage when FORTRAN 77 was in use and the C++ programming has evolved, naturally, there was no comparison of C++ for its excellence with respect to FORTRAN 77. The initial research projects which were started during FORTRAN 77 are still continuing with upgraded FORTRAN 90/95/2003 and 2008. Some of such global projects are in computational physics, earth sciences, and agriculture for clients which include Departments of Energy and Departments of Defense laboratories as well as commercial clients as diverse as hazardous waste companies and grain companies. Programs for numerical methods and codes based on them for modeling contaminant flow in porous media, modeling the structural mechanics of resonant sonic drilling rigs, modeling bi-static ground penetrator radar propagation in partially saturated soils, modeling solid dynamics at high strain rates, modeling the phonological development of corn and soy beans, modeling solar insolation in the photo synthetically active spectrum based on first principle atmospheric physics, and developed neural networks for the detection of dust clouds from satellite imagery [30].

FORTRAN has strict aliasing semantics compared to C++ and has been aggressively tuned for numerical performance for decades. Algorithms that uses the CPU to work with arrays of data often have the potential to benefit from a FORTRAN implementation [30].

LV. GFORTRAN COMPILER AND GEANY EDITOR

The FORTRAN compiler (gFortran) has gained support for the IEEE intrinsic modules specified by the FORTRAN 2003 and FORTRAN 2008 standards. The code was contributed by François-Xavier Coudert of CNRS [30].

Amongst many available Fortran 90 compilers, perhaps the most common is gFortran (part of the GNU Compiler Collection). It is Free and Open Source Software. Using Geany, an Integrated Development Environment (IDE) for writing and compiling codes is a good combination with gFortran and provides many nice features.

The gFortran installer can be downloaded from the GFortran page [ ]. Geany, which is a text editor using the GTK2 toolkit with basic features of an integrated development environment can be downloaded from the Geany website [30].

The main programming languages that are included with the Raspberry Pi are Scratch and Python. Those who want to learn how to use other programming languages can do so by using a code editor. A useful code editor to have is Geany. Once the Geany program has installed it will appear in the menu options under programming. To look at the programming that can be accomplished on the Geany Text Editor launch the code editor from the menu. Next, go to the Document menu. The drop down menu has option for set file type. Click on this option on the menu you will see four active options.

- Programming Languages - this includes assembler, Pascal, Fortran, Java, C, C++
- Scripting Languages - Python, JavaScript, PHP, Perl, Ruby being the most commonly used in "real" programming
- Markup Languages - HTML, XML as examples
- Miscellaneous - SQL file types, YAML are available among others

The list above is an indication of the languages that Geany is able to help. To program in any of the languages involves typing the program in the text-editing window. By saving the program using the appropriate file extension such as in the case of Python .py, Geany will run the program as a python program.

Geany will not teach you programming but will allow you to experiment with Python and other languages all in the same basic text-editing environment. As indicated above you can also use Geany to program in Python. The IDE (Integrated Development Environment), which is what Geany and IDLE (on your Raspberry Pi) are, is a step up from using the LX Terminal and Leaf pad. Geany is more sophisticated than IDLE in what can be done in the same code editor.

A good example of how to use the Geany code editor for Python can be found in [30]. The book describes about using the Geany code editor and tries to impart some basic programming skills as well.

Geany is useful to have on the Raspberry Pi for the reason that it opens up some of the more common languages used in producing commercial software. The cross-platform nature of Geany allows you to use the Raspberry Pi as a development tool for say Windows 8 apps that combine HTML, CSS and JavaScript. Google is Python Powered. The Google App Engine for building courses Course builder is a good example of Python in action. Quite along blog post today about the merits of Geany. The Raspberry Pi has limits only determined by your imagination (ok, a little bit to with the hardware). If your program runs well on a Raspberry Pi it is bound to run well when you go cross platform.

Geany is currently developed by the following people: Colomban Wendling, Nick Treleaven, Matthew Brush, Enrico Tröger, Frank Lanitz and many other contributors, translators and patch writers. The regular contributors are Lex Trotman and Eugene Arshinov [30].

Geany is a small and lightweight Integrated Development Environment. It was developed to provide a small and fast IDE, which has only a few dependencies from other

packages. Another goal was to be as independent as possible from a special Desktop Environment like KDE or GNOME - Geany only requires the GTK2 runtime libraries. Some basic features of Geany are as below [30] – [32]:

- Syntax highlighting
- Code folding
- Symbol name auto-completion
- Construct completion/snippets
- Auto-closing of XML and HTML tags
- Call tips
- Symbol lists
- Code navigation
- Build system to compile and execute your code
- Simple project management
- Plugin interface

Geany supports file-types various kinds of programming languages. Table 1 provides the list of such programming languages.

Table 1: Geany support file-types

Abaqus	F77	Pascal
Abc	Ferite	Perl
ActionScript	Forth	PHP
Ada	Fortran	Po
Asciidoc	FreeBasic	PowerShell
ASM	Genie	Python
Batch	GLSL	R
C	Go	reStructuredText
C#	Graphviz	Ruby
C++	Haskell	Rust
CAML	Haxe	Scala
Clojure	HTML	Sh
CMake	Java	SQL
COBOL	Javascrip	Tcl
Conf	LaTeX	Txt2tags
CSS	Lisp	Vala
CUDA	Lua	Verilog
Cython	Make	VHDL
D	Markdown	XML
Diff	Matlab	YAML
Docbook	NSIS	
Erlang	Objective-C	

As for as plugins are concerned, the Geany provides followings:

- Classbuilder: Creates source files for new class types
- Export: Exports the current file into different formats
- Filebrowser: Adds a file browser tab to the sidebar
- HTML Characters: Inserts HTML character entities like '&'
- Save Actions: Provides different actions related to saving files (autosave, instantsave, backupcopy)
- Split Window: Splits the editor view into two windows

In general, it is said that Geany should run on every platform, which is supported, by the GTK libraries. Geany comfortably runs in environments of various operating systems such as:

- Linux,
- FreeBSD,
- NetBSD,
- OpenBSD,
- MacOS X,
- Solaris Express and
- Windows

Here, it is to be noted that only the Windows port of Geany is missing some features such as AIX v5.3.

The code is licensed under the terms of the GNU General Public License with following available versions.

- Ubuntu 13.10
- Ubuntu 13.04
- Ubuntu 12.10
- Ubuntu 12.04
- Ubuntu 11.10
- Ubuntu 11.04
- Ubuntu 10.04

LVI. CONCLUSION

FORTRAN 77 was a terribly outdated language due to this reason C++ has attracted the attention it has in the scientific community. FORTRAN 90 has every feature in C that is important to scientific programming and most of the features of an object-oriented language. FORTRAN 90 lacks only inheritance and that was resolved by FORTRAN 2000.

FORTRAN 90 is designed to generate executable codes that are highly optimized and thus run extremely fast unlike C and C++. An example is pointers. Pointers are integral to C and C++ programming and because the compiler cannot determine whether a pointer is aliased, it is impossible for it to determine inter procedural dependencies. The result is a significant degradation in optimization and extremely slow execution speeds for most scientific codes with C and C++.

FORTRAN 90 pointers are designed to give the functionality of pointers, but with restrictions that eliminate issues such as aliasing. From a programming perspective however, an even more important point is that FORTRAN 90 has more natural ways of expressing the functionality that C and C++ require pointers to express. Because of this, FORTRAN 90 90 is a more natural language to program in and the time required for debugging codes is a fraction of that required by C and C++ (C++ is much worse than C).

FORTRAN 90 has another major advantage over C or C++ toward the programming parallel computers, which is in terms of linear memory model, only FORTRAN 90, has addressed this problem and providing standardized language support for parallelism. This support includes array syntax and many intrinsic functions for doing array operations varying from reduction operations such as array sums to matrix operations. With the use of FORTRAN 90 operator overloading and polymorphism, one can significantly extend the number of operations that avoid any reliance on the linear memory model.

We authors think that it is reasonable that students must learn C++ before they graduate, though even more important is that they learn how to program MATLAB. However, the issue is what freshmen should learn as their first language and for that, we would like to recommend FORTRAN 90 hands down. This is because it is a better language for scientific programming and is both easier to learn and use than the alternatives.

ACKNOWLEDGMENT

The authors are thankful to Sultan Qaboos University, Sultanate of Oman for providing research facilities, technical supports and research environment. The authors would like to express their great appreciations and gratitude for this to the university.

REFERENCES

[1] http://www.gnu.org/education/education.html
[2] Cambridge online learning community cambolc.co.uk
[3] Ahmad, A., Rizvi, M. A. K., Al-Lawati, A., Al-Abri, D. and Awadalla, M., "Design of a Probabilistic Based Software Tool for evaluating Controllability, Observability and Testability Models of Digital Systems", Indian Journal of Science and Technology (IJST), vol. 7, no. 10, pp. 1525- 1537, 2014.
[4] Samir A. Al-Busaidi, "Complementing Digital Logic Design with Logisim", TJER Vol. 11, No. 1, pp. 69-76, 2014.
[5] Ahmad, A., Ahmad, S., Al-Habsi, A. and Al-Hinai, M., (2014), "Understanding Universal Product Code – A Study and Simulation Experiments", Indian Journal of Industrial and Applied Mathematics (Taylor & Francis) (IJIAM), vol. 5, no. 1, pp. 25 – 34, 2014.
[6] Samir A. Al-Busaidi, "Complementing Digital Logic Design with Logisim", Free and Open Source Software Conference" (FOSSC-13), held at Muscat, Oman, February 18-19, 2013, pp. 4-9.
[7] Ahmad, A., Al-Abri, D. and Bait-Shiginah, F., (2013) "Software Testing Education – A Required Knowledge for Software Systems' Developers and Managers" Free and Open Source Software Conference" (FOSSC-13), held at Muscat, Oman, February 18-19, 2013, pp. 29-33.
[8] Ahmad, A. and Ruelens, D. and S. Ahmad, (2013), "Development of verification tool for minimal Boolean equation", IEEE Technology and Engineering Education (ITEE), vol. 8, no. 4, pp. 1-6, 2013.
[9] Afaq Ahmad, Sayyid Samir Al-Busaidi, Mufeed Juma Al-Musharafi, (2013), "On Properties of PN Sequences generated by LFSR – a Generalized Study and Simulation Modeling", Indian Journal of Science and Technology (IJST), vol. 6, no. 10, pp. 5351-5358, 2013.
[10] Ahmad, A. and Ruelens, D., (2013),"Development of digital logic design teaching tool using MATLAB & SIMULINK", IEEE Technology and Engineering Education (ITEE), vol. 8, no. 1, pp. 7-11, 2013.
[11] Ahmad, A., Al-Abri, D. and Al-Busaidi. S. S., (2012), "Adding Pseudo-Random Test Sequence Generator in the Test Simulator for DFT Approach, Computer Technology and Applications", vol. 3, no. 7, pp. 463-470, 2012.
[12] Ahmad, A. and Al-Abri, D., (2012), "Design of a Pseudo-Random Binary Code Generator via a Developed Simulation Model", International Journal on Information Technology (ACEEE - Journal), vol. 2, no. 1(March Issue), pp. 33-36, 2012.
[13] Ahmad, A., and Bait-Shiginah, F., (2012) "A Nonconventional Approach to Generating Efficient Binary Gray Code Sequences", IEEE Potentials, vol. 31, no. 3, pp. 16-19, 2012.
[14] Ahmad, A., Dawood Al-Abri, (2010), "Design of an Optimal Test Simulator for Built-In Self-Test Environment", Journal of Engineering Research, vol. 7, no. 2, pp. 69 – 79, 2010.
[15] Ahmad, A., Al-Mashari, A., Al-Lawati, A., (2010), "Development of a Computer Based Syndrome Diagnostic Testing Tool to Help in Children's Learning Process", International Journal of Education and Development using ICT (IJEDICT), vol. 6, no.1, pp. 76 - 87, 2010.
[16] Ahmad, A., and Mohammed M. Bait Suwailam, (2009), "A Less Complex Algorithmic Procedure for Computing Gray Codes", The Journal of Engineering Research, vol. 6, no. 2, , pp. 12 -19,2009.
[17] Ahmad, A., (2007), "Another Perspective in Generation and using of Gray Code-words", Journal of Electrical Engineering, IEEE Malaysia, (ELEKTRIKA), vol. 9, no. 2, 2007, pp. 49 – 55.
[18] Ali Al-Lawati and Ahmad, A., (2004), "Realization of a simplified controllability computation procedure – A MATLAB-SIMULINK based tool", Sultan Qaboos University Journal for Scientific Research - Science and Technology, Oman, vol. 8, pp. 131 – 143, 2004.
[19] Ahmad A., Al-Lawati, A. M. J. and Ahmed M. Al-Naamany, (2004), "Identification of test point insertion location via comprehensive knowledge of digital system's nodal controllability through a simulated tool", Asian Journal of Information Technology (AJIT), vol. 3, no. 3, pp. 142 – 147, 2004.
[20] Ahmad A., (2003), - An Invited Paper - "Secrets of error detection scheme of ISBN system," Caledonian Tech-News – A Journal: Caledonian Journal of Engineering (CJE), vol. 1, no. 1, pp. 10 – 14, 2003.
[21] Ahmad A., Al-Musharafi, M. J., and Al-Busaidi, S., (2002), "Study and implementation of properties of m-sequences in MATLAB-SIMULINK – A pass / fail test tool for designs of random generators", S.Q.U. Journal of Scientific Research – Science and Technology, vol. 7, part 1, pp. 147 – 156, 2002.
[22] Ahmad A. and Elabdalla A. M., (1997), "An efficient method to determine linear feedback connections in shift registers that generate maximal length pseudo-random up and down binary sequences", Computer & Electrical Engineering - An Int'l Journal (USA), vol. 23, no. 1, pp. 33-39, 1997.
[23] http://www.cmap.polytechnique.fr/~barbier/Guide_fortran/append-d.html
[24] Wikipedia Fortran article
[25] http://en.wikipedia.org/wiki/Fortran
[26] http://blog.profitbricks.com/top-integrated-developer-environments-ides/
[27] http://c2.com/cgi/wiki?DeadLanguageFortran
[28] http://stackoverflow.com/questions/146159/is-fortran-faster-than-c
[29] http://www.polyhedron.com/fortran-compiler-comparisons/compilerfeatures
[30] http://www.hpcwire.com/2011/09/20/why_fortran_still_matters/
[31] http://www.geany.org/
[32] http://www.geany.org/manual/reference/
[33] http://plugins.geany.org/