AT&T F2C Converter for Transputers
Version 1.1
Installation Notes

P5404 GP MIMD

Our Ref: GPMIMD:005/F2C/INSTALL

Revision 1.1

Mark Debbage and Mark B. Hill

September 11, 1992

1 Introduction

This document accompanies a port of the AT&T Bell Labs Fortran-to-C converter (`f2c`) for Inmos 32-bit transputers. The package requires the prior installation of the Inmos ANSI C Toolset (Inmos part number `Dx214`, where `x` depends on the host machine).

The `f2c` converter and its support libraries were acquired by an e-mail distribution from the *netlib* server accessed through `netlib@reseach.att.com`. The original sources were obtained on the 27th June 1992 and are distributed unchanged. A small number of modifications to the source were made to enable compilation on a transputer. These changes are distributed as textual difference files which can be applied to the original source using a patch utility. More information about the `f2c` package and details of the latest file updates can be obtained through e-mail:

```
echo 'send index from f2c' | mail netlib@research.att.com
```

The copyright notice from the AT&T `f2c` sources is repeated here for clarification of the distribution license:

Copyright 1990, 1991, 1992 by AT&T Bell Laboratories and Bellcore.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that the copyright notice and this permission notice and warranty disclaimer appear in supporting documentation, and that the names of AT&T Bell Laboratories or Bellcore or any of their entities not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

AT&T and Bellcore disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness. In no event shall AT&T or Bellcore be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

The components of this release produced at the University of Southampton are delivered with our standard license:

> This software and the accompanying documentation are licenced free of charge with absolutely no warranty. Copyright of the software and documentation is retained by the University of Southampton.
>
> Further copies of this software system may be made and distributed at will, provided that it is distributed in full with no modifications and that no charge is made other than media costs. Parties wishing to distribute modified versions of the package, including commercial products utilising components of the package, should contact the University for further clarification. Any derivative work must retain an acknowledgement to the University of Southampton and ESPRIT.
>
> Since this software is licenced free of charge it is supplied without warranty of any kind, either expressed or implied. The entire risk as to the quality and performance of this software is with the user.

The port of `f2c` has been funded by ESPRIT Project P5404, General Purpose MIMD.

2 Installation Procedure

This release is distributed as a tar file on a raw $3\frac{1}{2}$" floppy disk and is suitable for installation on UNIX systems. This can be typically recovered using:

```
tar xvf /dev/fd0
```

to produce a file `f2c.tar.Z` in the current working directory. This should be uncompressed and then untared to produce a directory `f2c` at the required point of installation. The following changes to the shell environment variables need to be made:

- `$F2C` should be directed towards the f2c installation.
- `$PATH` should contain `$F2C/bin/$ARCH`.
- `$ISEARCH` should contain `$F2C/libs/`.

where `$ARCH` indicates the host machine architecture. The distribution contains only a `sun4` setup, consisting of five scripts and two executables. The scripts are pertinent to UNIX systems featuring a `/bin/sh`. The `mergecmn` binary can be generated for a new architecture using the supplied makefile. Generation of a host `f2c` binary requires compilation of the `f2c` system (see Section 4), but the script `f2c` will invoke the transputer version if a host one cannot be found. The distribution contains a pre-compiled version for `sun4`s and compatibles. Other setups, using the transputer `f2c` binary, can be simply derived. For example, on a `sun3`:

```
cd $F2C/bin/sun4
mkdir ../sun3
cp makefile t8f77 taf77 t8fbtl tafbtl f2c ../sun3
cd ../sun3
make
```

An example code is provided to confirm correct installation. This can be tested by:

```
cd $F2C/example
make
iserver -sb fact.btl
```

The makefile is, by default, configured for transputer type `TA` and the program should therefore run on any transputer that is capable of compiling it.

3 Usage of Transputer

Generic usage of `f2c` is described in the `f2c` manual which accompanies the distribution; this is a postscript file `$F2C/docs/f2c.ps`. For the transputer port, scripts are provided for generating a bootable from a single-file Fortran program or for just compiling a single module from a multi-file Fortran program. Two transputer target types are supported, `TA` and `T8`:

`tafbtl program` compiles, links and collects a single-file Fortran program `program.f` into a bootable file `program.btl` for a `TA` transputer target.

`t8fbtl program` compiles, links and collects a single-file Fortran program `program.f` into a bootable file `program.btl` for a `T8` transputer target.

`taf77 program` compiles a Fortran unit `program.f` into an object file `program.o` for a `TA` transputer target.

`t8f77 program` compiles a Fortran unit `program.f` into an object file `program.o` for a `T8` transputer target.

The first two of these are sufficient to produce a bootable for a single-file Fortran program. If the program is multi-file then the compilation is more complex since the commons of the different units need to be amalgamated. This is accomplished using the second two scripts with the extra option `-e2c` which, in addition to translating the Fortran, also generates common block information. When `f2c` is applied to a code unit `filename.f` with this option asserted, then a file `filename.cmn` is generated containing struct/union declarations of all the common blocks defined in the file. After translation of all source files, the `.cmn` files need to be passed as command-line arguments to the tool `mergecmn` which generates a stub C unit reserving state for the commons. This C unit is then compiled and linked with the rest of the application. The example distributed with `f2c` illustrates the exact usage of this mechanism and includes a suitable makefile structure.

Code units produced by `f2c` are conventional C code units and these may therefore be mixed in multi-language transputer programs just as easily as usual C code units. There is no support however, for reduced code units. Full host capability must be provided to all processors that are to run Fortran code through this mechanism. The VCR package, also available from the authors, is ideal for this task.

4 Recompiling For Transputers

This section is only of interest if it is desired to recompile the `f2c` system. In normal use this will not be required.

The source for the `f2c` system comprises three major parts:

- The source for the converter itself (found in `$F2C/original/source`).
- The source for the in-built mathematical libraries (`$F2C/original/libf77`).
- The source for the host libraries (`$F2C/original/libi77`).

Each component is delivered as a tar file of original AT&T source and a set of patch files to recover the Southampton changes. The first step is to expand the tar files to produce the original file set. Assuming a suitable patch utility is available these patches can be applied by running the script `PATCH` in each of the three directories.

For the `f2c` translator source set, use:

```
make -f tp.mak
```

to generate a transputer bootable `f2c.btl`, or

```
make
```

to generate a UNIX host bootable `f2c`. These files need to be installed to the relevant directory in the binary hierarchy; this is best accomplished by symbolic links in the binary directories pointing back to the files in the build directories. Note that the file `gram.c` is automatically generated by `yacc` in the UNIX makefile. For the transputer however, the first line of `gram.c` needs to be deleted or commented out since it is ANSI-incompatible. A modified `gram.c` is hence distributed along with the difference files.

The run-time libraries can be rebuilt for the transputer by invoking `make` in the relevant directories. The output files (`libI77.lib` and `libF77.lib`) need to be installed to the library directory (as `libi77.lib` and `libf77.lib` respectively, please note the single character case change); this is best accomplished by symbolic links in the library directories pointing back to the files in the build directories.

In addition to the `f2c` libraries, a further file `iccfix.lib` is required to resolve some calls made by `f2c` that are not supported by the Inmos C system. This can be regenerated by the commands:

```
cd $F2C/iccfix
imakef -c -i iccfix.lib
make -f iccfix.mak
```

The target file `iccfix.lib` needs to be installed to the library directory; this is best accomplished by a symbolic link from the library directory pointing back to this file.

5 Changes From Original Sources

5.1 F2C Translator

`$F2C/source/main.c.diff` and `$F2C/source/names.c.diff`
: A new option `-e2c` has been added to support a compilation strategy which guarantees that when several modules define the same common block with different storage requirements that the largest space needed gets allocated.

`$F2C/source/sysdep.c.diff` and `$F2C/source/sysdep.h.diff`
: Minor changes to coerce to Inmos C Toolset header files and available library functions.

`$F2C/source/xsum0.out.diff`
: Recomputed checksums for ported version.

5.2 libf77 Run-time Library

`$F2C/source/libf77/makefile`
: The makefile has been adapted to echo all object files to a linker indirect file since there is a command line length limit of about 500 bytes on transputer commands.

5.3 libi77 Run-time Library

`$F2C/source/libf77/makefile`
: The makefile has been adapted to echo all object files to a linker indirect file since there is a command line length limit of about 500 bytes on transputer commands.

`$F2C/source/libf77/open.c` and `$F2C/source/libf77/err.c`
: Modified since Inmos C does not support the `fdopen` primitive

`$F2C/source/libf77/inquire.c`, `$F2C/source/libf77/util.c`, `$F2C/source/libf77/endfile.c`, `$F2C/source/libf77/fio.h`, `$F2C/source/libf77/fmt.h`, `$F2C/source/libf77/wrtfmt.c` and `$F2C/source/libf77/wsfe.c`
: Minor changes to coerce to Inmos C Toolset header files and available library functions.

6 Bug Reports

As always, problems should be reduced to the smallest code set that demonstates the fault.

Any problems with the generic `f2c` converter should be reported to `dmg@research.att.com` or `uunet!research!dmg`. Please quote that your `f2c` distribution is based on sources recovered from `netlib` on the 27th June 1992.

Any problems with the transputer port should be reported to `md@uk.ac.soton.ecs` quoting the version number on the front of this document.