Inria Esterel Compiler

The Inria Esterel compiler is the reference implementation of the Esterel V5 language. It can produce either automata code or netlist code.

=Documentation = For copyright reasons we are not allowed to distribute the documentation via HTTP. Therefore those files are only locally available via FILE links into the directory /home/esterel/v5_92/doc. You may browse the official Esterel Web for documents released to the public.

The manual pages for Esterel v5_92 are here located in

> */home/esterel/v5_92/man*

Use the Option -M /home/esterel/v5_92/man for the man command to display Esterel manual pages. Or extend the environment variable MANPATH by that directory (bash style):

> *export MANPATH=$MANPATH:/home/esterel/v5_92/man*

HTML-Versions of these manual pages are available in /home/esterel/v5_92/doc/html/.

Two other documents worth reading are:

- primer.pdf: The Esterel v5 Language Primer This paper gives an in depth view of the synchronous programming language Esterel.

- manual.pdf: The Esterel v5_91 System Manual This document describes the use of the Esterel compiler, the interfacing of generated code, and the simulation of Esterel programs.

Examples

There are two example directories prepared for use as templates for your own projects:

- /home/esterel/v5_92/example/ Native compilation and execution on the host (Unix/Linux) architecture.
- /home/esterel/v5_92/example-legOS/ Crosscompilation for the LEGO Mindstorms system with the alternative operating system legOS.

Each directory holds four files:

> */home/esterel/v5_92/example:*

- Makefile: Input file for the make command.
- abro.h: C header file with interface declarations.
- abro.strl: Esterel file.
- abro_data.c: C code with functions for data processing and OS interfaces.

> */home/esterel/v5_92/example-legOS:*

- Makefile: Input file for the make command.
- abro-rcx.h: C header file with interface declarations.
- abro-rcx.strl: Esterel file with the modified ABRO System.
- abro-rcx_data.c: C code with functions for data processing and legOS calls.

Both examples implement the standard ABRO example, which is named after the signals used: A and B as input signals, R as a reset signal and O as output. Details of this standard example can be found in the Esterel v5 Language Primer.

The implementation in /home/esterel/v5_92/example reads the keyboad keys A, B, and R as input signals A, B, and R . The output signal O is bound to a printf() call.

The legOS implementation uses the RCX inputs 1 and 2 as inputs A and B. The panel button Prgm is the source for the R signal. The output O is implemented as a beep on the RCX for each emitted signal.

The directory /home/esterel/v5_92/example-legOS contains another Esterel example: drive.strl and respective C files drive.h and drive_data.c. It is a simple example of the exec statement of Esterel.

To be able to modify or compile these files, copy them into a directory in your home account.

> *mkdir esterel cp /home/esterel/v5_92/example-legOS/* esterel cd esterel*

If you want to use them as templates for another project, then simply rename the files and modify them. When renaming, do not forget the line

> *#include "abro-rcx.h"*

in (formerly) "abro-rcx_data.c". The "abro-rcx" part must be changed to match your file names. Compilation Details Basically the compilation of an Esterel program for an Unix host works as follows: The file with Esterel code is compiled into C code with the Esterel compiler:

> *esterel abro.strl*

The resulting C code is compiled with a C compiler into object code:

> *gcc -c abro.c -o abro.o*

and with the object code of the *_data.c interface file:

> *gcc -c abro_data.c -o abro_data.o*

linked into one executable binary:

> *gcc abro.o abro_data.o -o abro*

Makefiles

These steps are conveniently implemented into the Makefile for each example directory: Makefile for the Unix target The Makefile in /home/esterel/v5_92/example compiles esterel programs for the Unix-Target.

The following targets are available (replace abro with the base name of your project):

- make abro compiles and links the Esterel file abro.strl along with abro_data.c into an executable file abro.
- make abro.c If you are only interested in the C code generated from the Esterel file you can use this target. It just calls the Esterel compiler to produce the C code abro.c from abro.strl.
- make abro.xes is a special case. The resulting file is not linked with abro_data.c but with the XES graphical debugger library from Esterel. It features a graphical interface to single step through the Esterel code with full control over all signals. The interface code in abro_data.c is not needed here. There are two files produced: the big abro.exe and the small abro.xes. abro.exe is the actual binary and abro.xes a shell script to set up an environment variable before starting the binary. Therefore please use abro.xes to start the Esterel simulation.

Makefile for the legOS target The Makefile in /home/esterel/v5_92/example-legOS compiles esterel programs for the legOS-Target. It is similar to the one for Unix with some additional targets for kernel and program upload into the RCX.

The following targets are available (replace abro-rcx with the base name of your project):

- make abro-rcx.lx compiles and links the Esterel file abro-rcx.strl along with abro-rcx_data.c into a binary image abro-rcx.lx. This image can be uploaded into the RCX to be executed.
- make abro-rcx.c If you are only interested in the C code generated from the Esterel file you can use this target. It just calls the Esterel compiler to produce the C code abro.c from abro.strl.
- make abro-rcx.dll uploads the binary image abro-rcx.lx into the RCX. If abro-rcx.lx is not existant or not up to date, then it is rebuilt.
- make kernel just uploads the kernel into the RCX (e.g. after a crash)
- make abro-rcx.xes is a special case. The resulting file is NOT executable on the RCX, instead it is linked with the XES graphical debugger library from Esterel. It is executable on the host and features a graphical interface to single step through the Esterel code with full control over all signals. The interface code in abro-rcx_data.c is not used here. There are two files produced: the big abro-rcx.exe and the small abro-rcx.xes. abro-rcx.exe is the actual binary and abro-rcx.xes a shell script to set up an environment variable before starting the binary. Therefore please use abro-rcx.xes to start the Esterel simulation.