gliss是一个ISS（Instruction Set Simulator指令集模拟器），它在SimNML（一种ADL - 架构描述语言）中输入一个（或者一组）文件，并生成一个包含模块的库，用于对给定指令集中的二进制程序进行解码，模拟和反汇编。

GLISS is an ISS (Instruction Set Simulator). It takes as input a file(or a collection of files) in SimNML (an ADL -- Architecture Description Language) and produces a library containing modules to decode,simulate and disassemble a binary program in the given instruction set.

GLISS v2 has the following dependencies:

\* OCAML

\* Perl,（是广泛应用的脚本编程语，perl初始是Larry为了格式化处理文本而创建的）

\* GNU Make.（GUN is a recursive acronym for “GNU's Not Unix!”GNU和Linux的组合是****GNU / Linux操作系统****，现在被数百万的使用，有时被简单的称为“Linux”。Gnu make是Linux环境下用来构建和管理工程的命令工具。单独的make命令是无法工作的，它需要一个Makefile文件。这个文件描述了整个工程的编译、连接规则，Makefile有自己的书写格式、命令、关键字。make读取Makefile，然后对这些规则解释执行，以完成工程管理。）

===== Directory Organization =====

\* ''gep'' -- generator libraries and commands,

\* ''irg'' -- core module to support language (parsing and intermediate

representation)

\* ''templates'' -- template file used for the C generation

of ISS library, simulator and disassembler.

\* ''lib'' -- other C files included in the ISS library but with

less customization.

\* ''doc'' -- user documentation.

\* ''autodoc'' (automatically generation) -- automatic documentation.

\* ''test'' -- test files.

- parsing (PF) -- produces an IRG representation (mostly bijective with the sources), instruction stays in their source original form;

- instantiation (IF) -- the actual whole set of instructions (WSI) is generated, mode and operation parameters are developed to the leaf actual parameters;

- generation (GF) -- GEP works mainly as a templater system taking as input the IRG and the WSI to generate files for the simulator.

The generation phase involves lot of transformation and analysis of the IRG and WSI:

\* building the fetch tree (to find back quickly the instruction identifier),

\* building the decoding functions (that extracts parameters from the instruction in binary form) and may involve complex parameter decoding,

\* simulation code generation that translate SimNML action to C code requiring support of bit operations, functional expressions, aliasing support, type equivalence, type operation translation, etc.

- 解析（PF） - 产生一个IRG表示，指令以原始形式存在;

- 实例化（IF） - 生成实际的整套指令（WSI），模式和操作参数被开发到实际参数;;

 - 生成（GF） - GEP主要作为一个模板系统，以输入IRG和WSI为模拟器生成文件。

生成阶段涉及IRG和WSI的大量转型和分析：

   \*构建提取树（快速查找指令标识符），

   \*构建解码功能（从二进制形式的指令中提取参数），可能涉及复杂的参数解码，

   \*模拟代码生成，将SimNML动作转换为需要支持位操作，功能表达式，别名支持，类型等价，类型操作转换的C代码。

Several instruction sets are already supported by GLISS v2:

\* PowerPC (ppc2)

\* ARM v5 (armv5t)

\* ARM v7 (armv7t)

\* Sparc (sparc)

\* TriCore (tricore)