William C. McSpadden <bill@riscv.org>; Martin Berger <contact@martinfriedrichberger.net>

== List of programming examples (in increasing complexity)

The main purpose of this document, is to give the user a set of programming examples for working on the RISC-V Sail model (often referred to as the RISC-V Golden Model). The examples will show the user how to change or extend the model. And it will also show the user how to write a RISC-V program (in both assembler and C) and then run it on the Golden Model.

You should read and utilize this document after you have a good handle on the Sail programming language.

[platform-configuration]

== Introduction

== How to contribute (Bill)

=== Coding and indentation style

=== Brevity

Program examples should be short, both in terms of number-of-lines and in terms of execution time. Each example should focus on one simple item. And the execution of the example item should be clear. The example should be short, standalone and easy to maintain.

=== Maintainership (when something breaks)

We would also ask that if you contribute a code example, that you would maintain it.

== Sail installation

TBD

=== Ubuntu (Bill Mc.)

TBD

=== MacOS (Martin)

TBD

=== Docker

Docker is used as a ....

=== Windows

=== Windows: Cygwin (Bill Mc., low priority)

=== Other?

== Basic description === What Sail is Sail is a programming language that is targetted for specifying an ISA. Once specified, a set of instructions (usually found in a .elf file) can then be executed on the "model" and the results observed.

The model is a sequential model only; at this time, there are no semantics allowing for any type of parallel execution.

=== What sail is not Sail is not an RTL (Register Transfer Language). There is no direct support for timing (as in clock timing) and there is no support for parallel execution, all things that an RTL contains.

=== version management and what to expect TBD

== Platform Configuration example (Bill)

== FAQs (Frequently Asked Questions)

Following are a set of FAQs that were generated via set of questions to the Sail developers.

=== Frequently Asked Questions about the Sail RISC-V Golden Model

[q_is_there_support_for_multi_hart_multi_core_simulation]

[q_what_are_ml_files__what_are_their_purpose]

[q__is_there_any_support_for_MTIMER]

[q*is_the*main_loop__coded_in_Sail]

[q-can-gdb-attach-to-the-riscv-golden-model-to-debug-riscv-code]

[q__why_two_executables]

[q___is_there_support_in_the_model_for_misaligned_memory_accesses]

[q-what-is-the-meaning-of-life-the-universe-and-everything]

[q-what-does-the-answer-to-what-is-the-meaning-of-life-the-universe-and-everything-mean]

==== Q: Is there support for multi-HART or multi-Core simulation?

A: There is no inherent support for multi-HART or multi-Core within the existing RISC-V Sail model. There are future plans for adding this kind of simulation. It is needed in order to simulate (in a meaningful way) the atomic memory operations and to evaluate memory consistency and coherency.

The model isn't directly about testing. Testing is a separate activity. The point of the model is to be as clear as possible. and we should keep testing and the model separate.

==== Q: What are .ml files? What are their purpose?

A: These are OCaml files. They are to the ocaml emulator what the .c files are to the c emulator. I question the need for an OCaml emulator ,see also https://github.com/riscv/sail-riscv/issues/

138

==== Q: Is there any support for MTIMER?

A: Yes. MTIMER functionality lives in riscv_platform.sail. At this date (2022-05-27) it lives at a fixed MMIO space as specified by the MCONFIG CSR. In the future, once the Golden Model supports the RISCV_config YAML structure, the MTIMER can be assigned any address.

==== Q: Is the "main loop" coded in Sail?

A: The initial answer to this question ("The main execution loop can be found in `main.sail`.") is incorrect. main.sail is not executed in the RISC-V model, even though it is compiled into the model.

The main loop is actually found on the C side in the file `c_emulator/riscv_sim.c` in the function `run_sail()`. In this function, the Sail function, `zstep()`, is called (which is the Sail function, `step()`)

==== Q: Can gdb attach to the RISCV Golden Model to debug RISCV code?

A: Not at this time (2022-05-27). It is being looked at as an enhancement.

==== Q: There are two C executables built: riscv_sim_RV32 and riscv_sim_RV64. Is there a reason why we need two executables? Can't XLEN be treated as a run-time setting rather than a compile time setting?

A: (Response from Martin Berger) I think this would require a redesign of the Sail code because of the way Sail's liquid types work. Currently xlen is a global type constant, that is used, directly or indirectly, everywhere. As a type-constant it is used during type checking. The typing system might (note the subjunctive) be flexible enough to turn this into a type-parameter, but probably not without major code surgery. I think we should ask the Cambridge team why they decided on the current approach.

==== Q: Is there support in the model for misaligned memory accesses?

A: (Response from Martin Berger) Short answer: I don't know. Alignment stuff is distributed all over the code base. riscv_platform.sail has some configuration options for this. Maybe that's a place to start looking?

==== Q: What is the meaning of life, the universe and everything?

A: 42

==== Q: What does the answer to "What is the meaning of life, the universe and everything" mean?

A: One must construct an experimental, organic computer to compute the meaning. Project 'Earth' is one such computer. Timeframe for an expected answer is... soon.