Errata and updates to “Generating Trustworthy I^{2}C Stacks Across Software and Hardware”

Zikai Liu
Systems Group, ETH Zurich

September 4, 2025

Missing prior work on “calls and continuations” The way the C backend transforms the `talk` and `read` operations into function calls and continuations is in fact a coroutine programming technique pioneered by the previous work [3, 4]. Specifically, this is the “stack-based coroutines” described by Tatham [5]. In Tatham’s terms, “continuation” is “return and continue”. The thesis (and the foundational work [2]) did not discuss and cite them properly.

The validity of the C backend The thesis claimed that whether the properties verified by the Promela verifiers transfer to the C implementation remains a question. With the new insight into the coroutine programming [5], I would claim the properties do transfer. Informally, the Promela verifiers verify *all* possible traces of the coroutines (processes), while the code generated by the C backend executes *one* of the traces.

The C backend can handle “calling graphs” The thesis claimed that the C backend can only handle “calling trees” but not generic “calling network” with the current implementation. Specifically, one layer can make function calls to multiple layers, but can only return to one.

With the new insight into the coroutine programming [5], I believe the claim was wrong. The layer (callee) can indeed return to different layers. The destination is not visible from the code (it is just a return) but the return address is stored on the stack. The missing output parameters are not a problem either since the callee can select the correct output channel. Therefore, CallingTree can be extended to CallingGraph. The implementation and evaluation remain to be done.

Miscellaneous Figure 4.11 had a formatting issue that the code on the left side overflowed and overlapped with the right part.

The total LUTs and FFs on the UltraScale+ MPSoC are missing citations [1].

References

[1] AMD. *Zynq UltraScale+ MPSoC Data Sheet: Overview (DS891)*, November 2022.

[2] Lukas Humbel, Daniel Schwyn, Nora Hossle, Roni Haecki, Melissa Licciardello, Jan Schaer, David Cock, Michael Giardino, and Timothy Roscoe. A Model-Checked I2C Specification. In Alfons Laarman and Ana Sokolova, editors, *Model Checking Software*, Lecture Notes in Computer Science, pages 177–193, Cham, 2021. Springer International Publishing.

[3] Donald E. Knuth. *The art of computer programming, volume 1 (3rd ed.): fundamental algorithms*. Addison Wesley Longman Publishing Co., Inc., USA, 1997.

[4] Christopher D. Marlin. *Coroutines*, volume 95 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, 1980.

[5] Simon Tatham. Coroutines in c, 2000. Accessed: 2024-05-17.