Porting the Steel Bank Common Lisp Compiler and Runtime to the Nintendo Switch NX Platform

Charles Zhang

Yukari "Shinmera" Hafner

charleszhang99@yahoo.com

shinmera@tymoon.eu

Shirakumo.org

Zürich, Switzerland

ABSTRACT

The Nintendo Switch (NX) is a 64-bit ARM-based platform for video games with a proprietary micro-kernel operating system. Notably this system does not give programs the ability to mark pages as executable or give access to thread signal handlers, both of which present a significant hurdle to SBCL's intended bootstrap process and runtime operation. We present our efforts to adapt the SBCL runtime and compiler to deploy applications onto the NX platform.

CCS CONCEPTS

• **Software and its engineering** → **Runtime environments**; **Dynamic compilers**; *Garbage collection*; Software creation and management.

KEYWORDS

Common Lisp, SBCL, porting, ARM, aarch64, NX, Experience Report

ACM Reference Format:

Charles Zhang and Yukari "Shinmera" Hafner. 2025. Porting the Steel Bank Common Lisp Compiler and Runtime to the Nintendo Switch NX Platform. In *Proceedings of the 18th European Lisp Symposium (ELS'25)*. ACM, New York, NY, USA, 1 page.

1 INTRODUCTION

2 RELATED WORK

Rhodes[2] outlines the methodology behind the SBCL bootstrapping process.

A pure Common Lisp bootstrapping process as Durand et al.[1] outline would however not notably improve our process, as all our challenges arise from not being able to bootstrap on the desired target platform in the first place, and needing to handle the low-level system construction processes to be amenable for the NX' restrictions.

Citing information on the NX' operating system is difficult as it is a closed-source platform with all usual information placed under NDA. All publicly available information is from security research such as by Roussel-Tarbouriech et al.[3].

Particularly, we are unaware of any publication about the porting of other runtime environments to the NX, such as C#, JavaScript, Lua, etc.

3 BUILD SYSTEM

4 RELOCATION

5 GARBAGE COLLECTION

6 CONCLUSION

7 FURTHER WORK

8 ACKNOWLEDGEMENTS

We would like to thank Douglas Katzmann for his help and advice for various parts of the porting effort, as well as the rest of the SBCL maintenance team for their continuous improvements to the SBCL platform.

REFERENCES

[1] Irène A Durand and Robert Strandh. Bootstrapping common lisp using common lisp. In *EUROPEAN LISP SYMPOSIUM*, 2019.

[2] Christophe Rhodes. Sbcl: A sanely-bootstrappable common lisp. In *Workshop on Self-sustaining Systems*, pages 74–86. Springer, 2008.

[3] Gauvain Tanguy Henri Gabriel Roussel-Tarbouriech, Noel Menard, Tyler True, Tini Vi, et al. Methodically defeating nintendo switch security. *arXiv preprint arXiv:1905.07643*, 2019.

ELS'25, May 19–20 2025, Zürich, Switzerland

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM