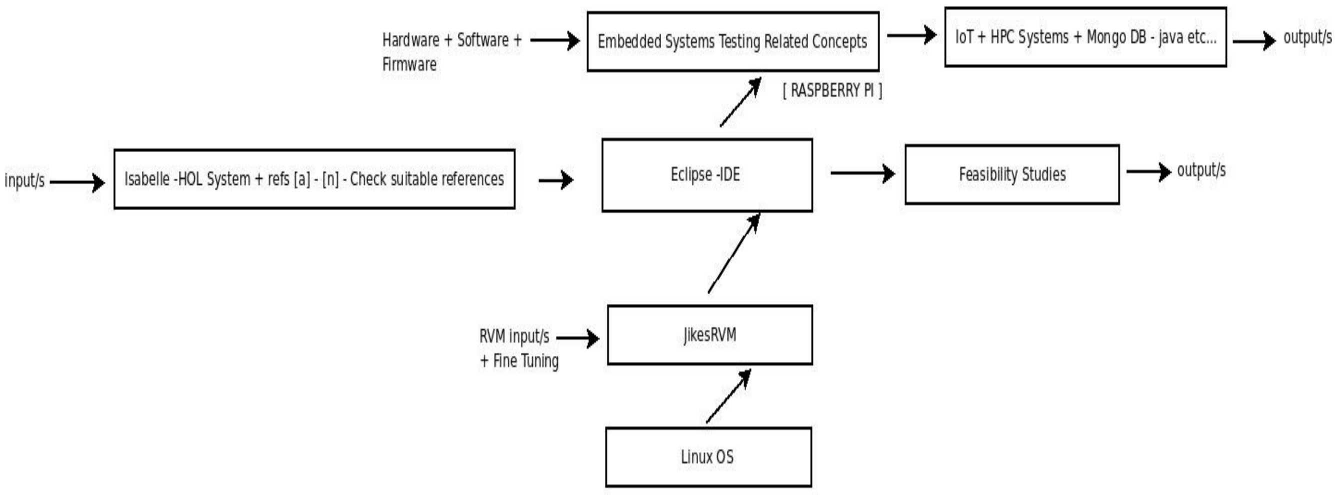
Understanding Microjava Theory/HOL-Isabelle + Testing .mj Compiler/Gradle Implementation For Further Advanced Sensor Informatics R&D.

Nirmal – Informatics R&D USA/UK/Israel/BRICS Group of Nations.
Current Member ante Inst UTD Dallas TX USA.
Contact_info hmfg2014@gmail.com

[I] Idea + Inspiration + Introduction :

Sensor Informatics Using JVM Languages + HOL/Isabelle based Microjava Theory – A Simple Suggestion on Exploring Microjava/.mj Compiler for Smart Devices + IoT + HPC Systems Research.

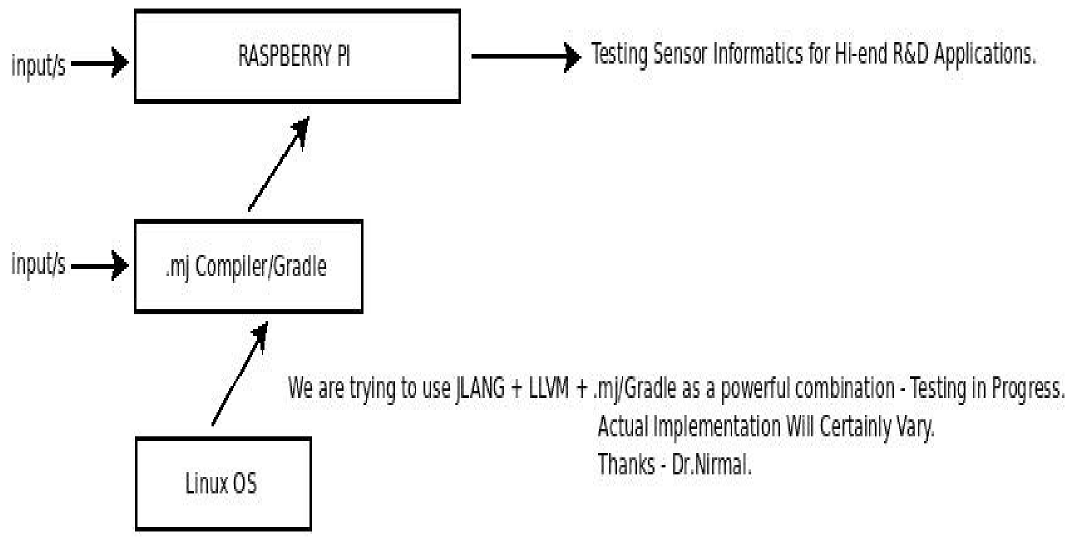
[II] JVM Languages based Sensor-informatics R&D Framework :


Hardware + Software + Firmware
Embedded Systems Testing Related Concepts
IoT + HPC Systems + Mongo DB - java etc...
output/s
[RASPBERRY PI]
input/s
Isabelle -HOL System + refs [a] - [n] - Check suitable references
Eclipse -IDE
Feasibility Studies
output/s
RVM input/s + Fine Tuning
JikesRVM
Linux OS

Simple Algorithm I - To Test Simple Ideas Using Isabelle - HOL System & its related Tools on MicroJava
Testing in Progress.Actual Implementation Will Certainly Vary.
Thanks for Understanding - Dr.Nirmal.

JX OS + mjvmk → Seem to be promsing tools in our exploration of Embedded Systems - Please Check our references
Very Useful in Understanding .mj Compiler/Gradle implementation and Testing on RASPBERRY PI for example.
We are Testing on different Systems -> Bosch XDK -IoT ToolKIT /SOLID RUN from Israel etc...
However we are not recommending any product/s here for any specific purpose.
Just to informa you about the possibilities.
Thanks.

[Figure I – Algorithm I]


input/s
RASPBERRY PI
Testing Sensor Informatics for Hi-end R&D Applications.
input/s
.mj Compiler/Gradle
We are trying to use JLANG + LLVM + .mj/Gradle as a powerful combination - Testing in Progress.
Actual Implementation Will Certainly Vary.
Thanks - Dr.Nirmal.
Linux OS

[Figure II- Algorithm II]

[III] Important References :

[a] https://isabelle.in.tum.de/library/HOL/HOL-MicroJava/index.html – MicroJava

[b] https://github.com/DanijelAskov/microjava-compiler - .mj compiler + Gradle

[c] https://github.com/markic/microjava-compiler -.mj compiler

[d] https://github.com/tejdnk-2019-ShortNotes

[e] https://gradle.org/features/ && (https://github.com/slegge) - Embedded Gradle.

[f] https://www.jikesrvm.org/

[g] http://rok.strnisa.com/lj/ - Light Weight Java

[h] https://doi.org/10.1145/3293880.3294104 - JINJA

[i] http://doi.acm.org/10.1145/604131.604148

[j] http://dx.doi.org/10.1007/11924661_24 – Bytecode Logic(JML)

[k] http://doi.acm.org/10.1145/503502.503505 – Feather Weight Java

[l] https://hal.inria.fr/hal-02427360 - A Generic Framework for Verified Compilers Using {I}sabelle/{HOL}'s Locales.

[m] JX OS - http://www4.informatik.uni-erlangen.de/Projects/JX/Download

[n] mjvmk - https://seancfoley.github.io/mjvmk/) - Micro Java Virtual Machine Kernel

[o] https://polyglot-compiler.github.io/JLang/ - Jlang + LLVM

[p] https://snapcraft.io/install/gradle/raspbian – RASPBERRY PI + GRADLE

[IV] Acknowledgment/s :

Sincere Thanks to all WHO made this happen in my LIFE. Non-Profit R&D.

Inspiring Others is GOOD Always.

[V] Main References :

[1] G. Klein and T. Nipkow. Verified lightweight bytecode verification. In S. Drossopoulou, S. Eisenbach, B. Jacobs, G. T. Leavens, P. Müller, and A. Poetzsch-Heffter, editors, Formal Techniques for Java Programs. Technical Report 269, 5/2000, Fernuniversität Hagen, 2000. ECOOP2000 Workshop proceedings available from http://www.informatik.fernuni-hagen. de/pi5/publications.html.

[2] G. Klein and T. Nipow. Verified lightweight bytecode verification. Concurrency and Computation: Practice and Experience, 13(13):1133–1151, 2001. Invited contribution to special issue on Formal Techniques for Java.

[3] T. Nipkow. Verified bytecode verifiers. In F. Honsell, editor, Foundations of Software Science and Computation Structures (FOSSACS 2001), volume 2030, pages 347–363, 2001.

[4] T. Nipkow, D. v. Oheimb, and C. Pusch. µJava: Embedding a programming language in a theorem prover. In F. L. Bauer and R. Steinbrüggen, editors, Foundations of Secure Computation, volume 175 of NATO Science Series F: Computer and Systems Sciences, pages 117–144. IOS Press, 2000.

[5] D. von Oheimb. Axiomatic semantics for Java `ight in Isabelle/HOL. In S. Drossopoulou, S. Eisenbach, B. Jacobs, G. T. Leavens, P. Müller, and A. Poetzsch-Heffter, editors, Formal Techniques for Java Programs. Technical Report 269, 5/2000, Fernuniversität Hagen, 2000. ECOOP2000 Workshop proceedings available from http://www.informatik.fernuni-hagen. de/pi5/publications.html.

[THE END]