Porting Micropython to LEON platforms (activities at ESTEC)

TEC/SWE: David Sanchez de la Llana
ESTEC
10/10/2016

Contents

- Testing at ESTEC on real targets
- Intended activities

Testing during the TRP (remainder)

- Micropython has a ‘benchmark’ common to all the ports
  - The full benchmark was executed successfully (384 tests)
    - Using the ESOC EMU LEON 2 emulator
- Specific tests (32 tests) created for the LEON port:
  - Test language features with heap allocation blocked
  - RTEMS C API
    - Tasking
    - Queues
    - Semaphores
  - VM control
  - Robustness
  - Performances

Testing at ESTEC

- The MicroPython ‘benchmark’ has been improved so can be used with gdb/grmon on real targets:
  - Minimum changes required to the test bench, it is quite flexible
- Tested in 2 different targets:
  - LEON2 (AT-697E) at 80 MHz
  - LEON3 (GR-712RC) at 80 MHz

- RTEMS version used: RTEMS 4.8 (OAR)
  - Compilation and testing with other RTEMS versions in progress (see last slides)
    - RTEMS OAR 4.10
    - RTEMS OAR 4.11

Results on LEON3 (GR-712RC)

- GR-712 at 80 MHz, RAM waitstates = 2, only one core used
- Benchmark results:
  - All tests executed. No failures
- Pystone (leon-ex-pystone)

	64BIT_NAN_BOXING= FALSE (32 bits objects pointer)	64BIT_NAN_BOXING= TRUE (64 bits objects pointer)
CFLAGS = '-Os' (size)	Code size (**289K**) Pystone(1.2) time for 1000 passes = 1.17253 This machine benchmarks at **852.854** pystones/second	Code size (**324K**) Pystone(1.2) time for 1000 passes = 2.75392 This machine benchmarks at **363.119** pystones/second
CFLAGS = '-O3' (speed)	Code size (**328K**) Pystone(1.2) time for 1000 passes = 1.17357 This machine benchmarks at **852.099** pystones/second	Code size (**338K**) Pystone(1.2) time for 1000 passes = 2.75248 This machine benchmarks at **363.309** pystones/second

Results on LEON3 (GR-712RC) Comments

- Influence of 32/64 bits pointer is the most important:
  - 32 bit object pointers => general improvements of performances
  - But… no heap determinism (each flop operation creates objects)
  - If heavy floating point use, decrease of performances by use of 64-bits object pointers compensated by faster floating point (no overload of creating object)
- Influence of –O not important for speed, important for size
  - Explanation: the VM interpreter is very optimized
- A lot of options available in file “py/config.h” (general), “leon-xx/mpconfigport.h”
  - E.g. activate or not GC, stack check, language features…
  - Mainly affects the size of VM (can be decreased by deactivating features)

Preliminary results with RTEMS 4.10, 4.11

- With RTEMS 4.10:
  - Needs relax of –Werror (warning pointer type in queue module)
  - Some tests performed passed successfully, work in progress
- With RTEMS 4.11:
  - _Thread_Executing replaced by function _Thread_Executing() due to multicore issues
  - “rtems_clock_time type” obsolete. Changed (easy)
  - RTEMS “notepad” API depreciated. Needs replacement
  - Some tests performed passed successfully, work in progress

Intended activities

- Maintenance of MicroPython (TEC framework contract). List of possible activities (to be decided according to budget, priority, etc.)
  - Improvements (support of RTEMS rate monotonic API, etc)
  - Complete port and testing in top of RTEMS 4.10, RTEMS 4.11
  - Integration into the TASTE ecosystem
  - Bare porting to XstratuM
- Qualification of MicroPython
  - Activity submitted to IPC
  - Target, RTEMS 4.8 Edisoft, CAT-C
- MicroPython is identified as a building block in the on-going Payload Execution Platform Software (PEPS) ITT (AO8658/16.132.02)

Contact

- Contacts:
  - George Robotics LTD:
    - dpgeorge@georgerobotics.co.uk
  - ESA (TEC-SWE):
    - David.Sanchez.de.la.Llana@esa.int
  - European Space Software Repository
    - https://essr.esa.int/

QUESTIONS?