Towards ROS 2 micro-controller meta cross-compilation

ROSCON 2018, September 30th, Madrid

Iñigo Muguruza Goenaga, Juan Muñoz Flores, Víctor Mayoral Vilches (Erle Robotics)
Loïc Dauphin, Emmanuel Baccelli, Cedric Adjih (INRIA)

Discussed topics

- Previous work mention with μCs … and new approaches
- Meta cross-compilation approach
- Examples
- Call for collaboration

Previous Work

- ***ROSSerial*** *(ROS 1)*
    - Uses a proxy to integrate µC pub/subs in ROS network
    - Based on Arduino-like µCs,
    - Based on serial/USB based and limited to 32 bytes data
- ***ros2_embededded_nuttx*** (ROS2)
    - Required additional external RAM
    - Complex to port complete DDS (Tinq)
- **FreeRTPS** (ROS2) => baremetal, small RTPS implementation, µC in mind
- ***uROSnode*** *(ROS1)*

Novel Work

New approaches have arisen, confirming the interest of community to support ROS 2.0 in µCs.

OFERA project

- **Complete ROS 2.0 stack based for µCs (XRCE-DDS oriented)**
- ROSSerial for ROS 2.0*

*new approach being considered

Issues

Support for µC in ROS 2.0 difficult because:

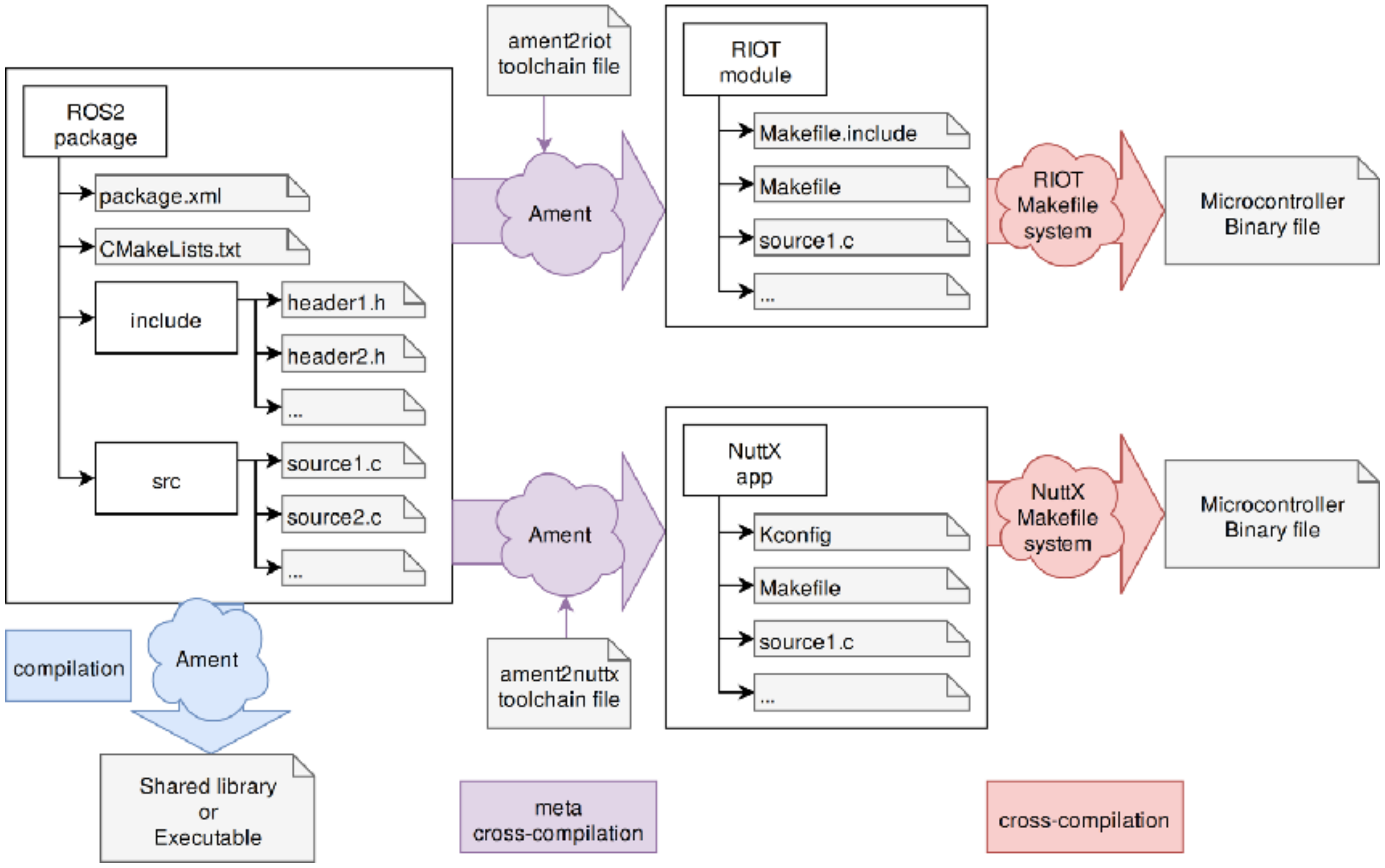
- RAM !
- C++ 11/14 => Not supported for µC toolchains
- Toolchain are different: CMake vs Makefile

Brainstorming outcome

Approaches to support µCs:

1. Add support CMake to each RTOS/platform
2. Support RTOS-dependant compilation process in ROS 2.0
3. “meta cross-compile” => Convert ROS 2.0 code into RTOS-centric code

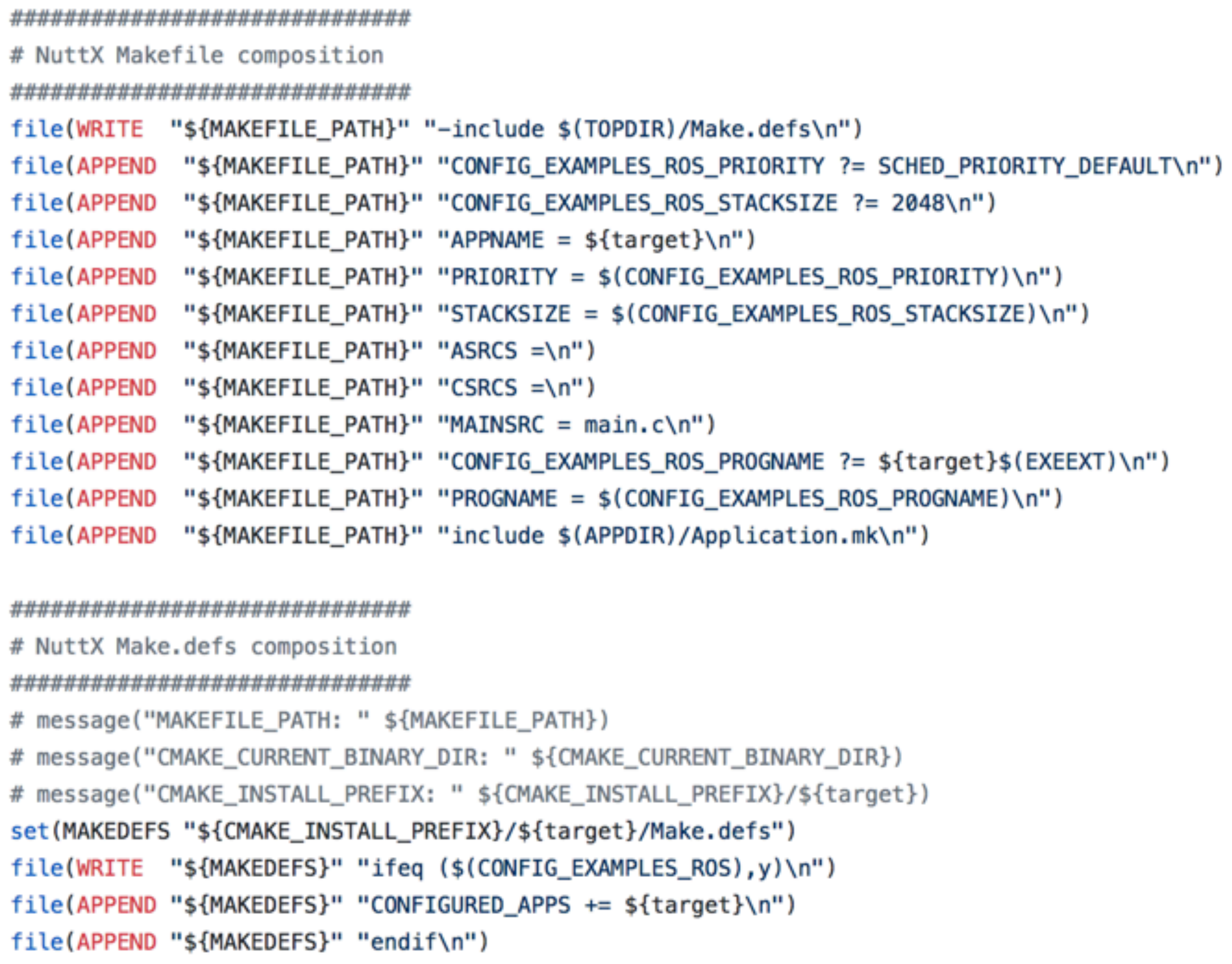
Meta Cross-Compilation



Erle Robotics

inria inventors for the digital world

Meta Cross-Compilation



```
##############################
# NuttX Makefile composition
##############################
file(WRITE  "${MAKEFILE_PATH}" "-include $(TOPDIR)/Make.defs\n")
file(APPEND  "${MAKEFILE_PATH}" "CONFIG_EXAMPLES_ROS_PRIORITY ?= SCHED_PRIORITY_DEFAULT\n")
file(APPEND  "${MAKEFILE_PATH}" "CONFIG_EXAMPLES_ROS_STACKSIZE ?= 2048\n")
file(APPEND  "${MAKEFILE_PATH}" "APPNAME = ${target}\n")
file(APPEND  "${MAKEFILE_PATH}" "PRIORITY = $(CONFIG_EXAMPLES_ROS_PRIORITY)\n")
file(APPEND  "${MAKEFILE_PATH}" "STACKSIZE = $(CONFIG_EXAMPLES_ROS_STACKSIZE)\n")
file(APPEND  "${MAKEFILE_PATH}" "ASRCS =\n")
file(APPEND  "${MAKEFILE_PATH}" "CSRCS =\n")
file(APPEND  "${MAKEFILE_PATH}" "MAINSRC = main.c\n")
file(APPEND  "${MAKEFILE_PATH}" "CONFIG_EXAMPLES_ROS_PROGNAME ?= ${target}$(EXEEXT)\n")
file(APPEND  "${MAKEFILE_PATH}" "PROGNAME = $(CONFIG_EXAMPLES_ROS_PROGNAME)\n")
file(APPEND  "${MAKEFILE_PATH}" "include $(APPDIR)/Application.mk\n")

##############################
# NuttX Make.defs composition
##############################
# message("MAKEFILE_PATH: " ${MAKEFILE_PATH})
# message("CMAKE_CURRENT_BINARY_DIR: " ${CMAKE_CURRENT_BINARY_DIR})
# message("CMAKE_INSTALL_PREFIX: " ${CMAKE_INSTALL_PREFIX}/${target})
set(MAKEDEFS "${CMAKE_INSTALL_PREFIX}/${target}/Make.defs")
file(WRITE  "${MAKEDEFS}" "ifeq ($(CONFIG_EXAMPLES_ROS),y)\n")
file(APPEND "${MAKEDEFS}" "CONFIGURED_APPS += ${target}\n")
file(APPEND "${MAKEDEFS}" "endif\n")
```

Erle Robotics

https://github.com/erlerobot/riot-ros2/blob/nuttx/ament2nuttx.cmake

inventors for the digital world

Meta Cross-Compilation

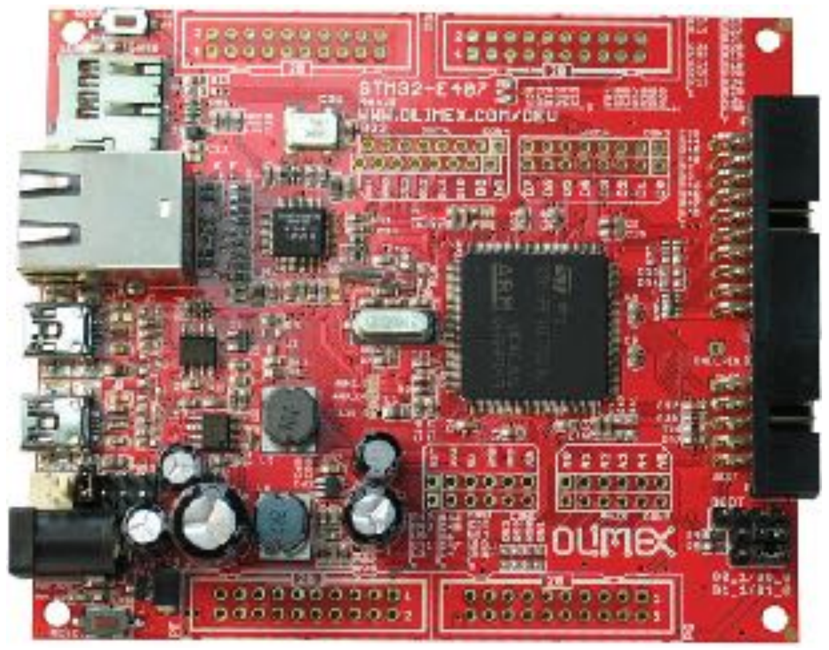
```
################################
# NuttX Kconfig composition
################################
set(KCONFIG "${CMAKE_INSTALL_PREFIX}/${target}/Kconfig")
file(WRITE  "${KCONFIG}" "config EXAMPLES_ROS\n")
# file(APPEND "${KCONFIG}" "config EXAMPLES_ROS\n")
file(APPEND "${KCONFIG}" " bool \"\\\"Hello, micro-ROS!\\\" example\"\n")
file(APPEND "${KCONFIG}" " default n\n")
file(APPEND "${KCONFIG}" " ---help---\n")
file(APPEND "${KCONFIG}" "         Enable the \\\"Hello, micro-ROS!\\\" example\n")
file(APPEND "${KCONFIG}" "if EXAMPLES_ROS\n")
file(APPEND "${KCONFIG}" "config EXAMPLES_ROS_PROGNAME\n")
file(APPEND "${KCONFIG}" " string \"Program name\"\n")
file(APPEND "${KCONFIG}" " default \"hello\"\n")
file(APPEND "${KCONFIG}" " depends on BUILD_KERNEL\n")
file(APPEND "${KCONFIG}" " ---help---\n")
file(APPEND "${KCONFIG}" "         This is the name of the program that will be use when the NSH ELF\n")
file(APPEND "${KCONFIG}" "         program is installed.\n")
file(APPEND "${KCONFIG}" "config EXAMPLES_ROS_PRIORITY\n")
file(APPEND "${KCONFIG}" " int \"Hello task priority\"\n")
file(APPEND "${KCONFIG}" " default 100\n")
file(APPEND "${KCONFIG}" "config EXAMPLES_ROS_STACKSIZE\n")
file(APPEND "${KCONFIG}" " int \"Hello stack size\"\n")
file(APPEND "${KCONFIG}" " default 2048\n")
file(APPEND "${KCONFIG}" "endif\n")
```

Erle Robotics

https://github.com/erlerobot/riot-ros2/blob/nuttx/ament2nuttx.cmake

inventors for the digital world

Proof of concepts



- ARM Cortex-M4F, 256KB RAM and 1MB Flash, Atmel ATSAMR21G18A

https://github.com/erlerobot/riot-ros2/tree/nuttx

Erle Robotics

- ARM Cortex-M0+, 32KB RAM and 256KB Flash, Atmel ATSAMR21G18A
- Using RCLC, RCL, RMW_NDN, NDN, RIOT => 78 KB Flash (~30%) and 10 KB RAM (~31%)

https://github.com/astralien3000/riot-ros2

Inria
inventors for the digital world

Next steps & call for collaboration

- The interest of using ROS in µC is more alive than ever!
- Interested? Contact us! (inigo@erlerobotics.com)