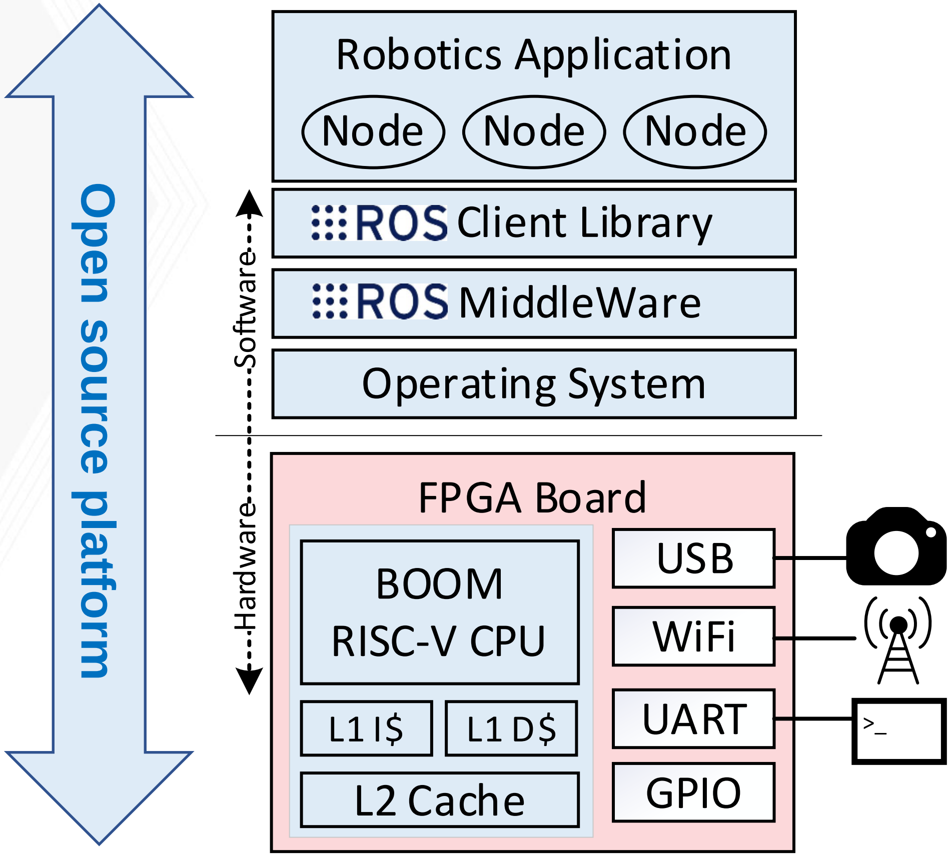
RISC-V FPGA Platform toward ROS-based Robotics Application

Jaewon Lee, Hanning Chen,
Jeffrey Young, Hyesoon Kim

Georgia Institute of Technology

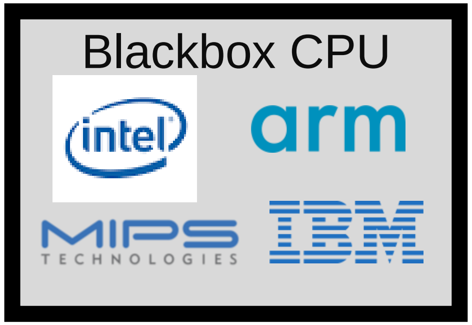
Goal

- To build an end-to-end open source platform for remote robot control application



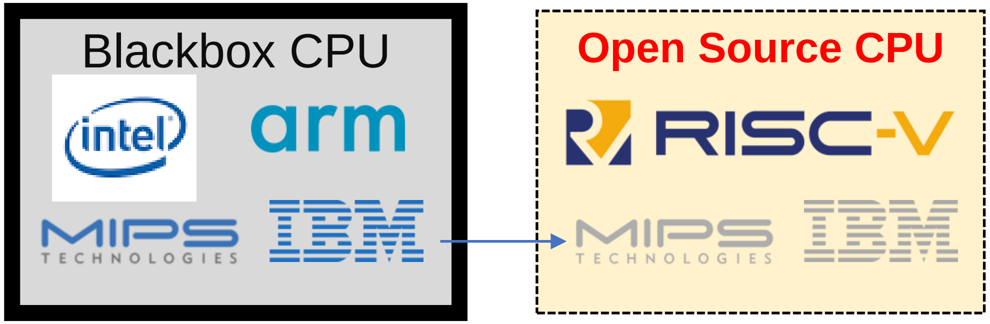
Open source software is everywhere! But how about hardware?

- Why has open source hardware not grown like the software open source community?
  - High barrier to entry - hardware development requires deep knowledge and requires effort of many experts from various areas
    - Digital Circuit design
    - Compiler
    - Operating system
- Only a few big vendors can provide the CPU hardware and compatible software ecosystem



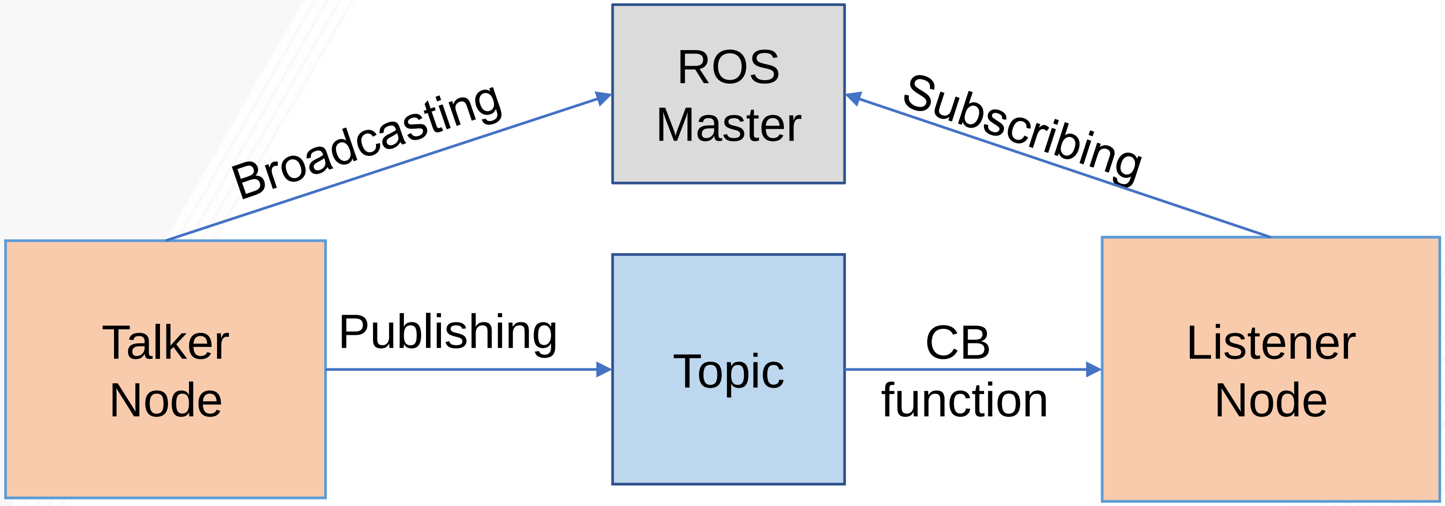
Open standard ISA: RISC-V

- RISC-V (2010~)
  - Royalty-free ISA
  - Supported by both hardware and software open source communities
    - Chisel provides high-level synthesis to HDL
    - Supported by mainstream Linux kernels
    - Clang/LLVM toolchain for compilation
    - Verilator provides an open source HW simulator
    - Has various open source reference platforms (Si-Five Freedom, PulPino)
- Now, anyone can easily build and support a CPU
  - Even students can build modern CPU system with affordable FPGAs
  - RISC-V also has drawn in open source efforts from other vendors like MIPS and IBM



…+ Open source Robotics platform!

- ROS (Robot Operating System)
  - Robotics middleware software
  - Software framework to control robotics components (nodes) and to support communication among each node
  - Of course, ROS is also an open source project!

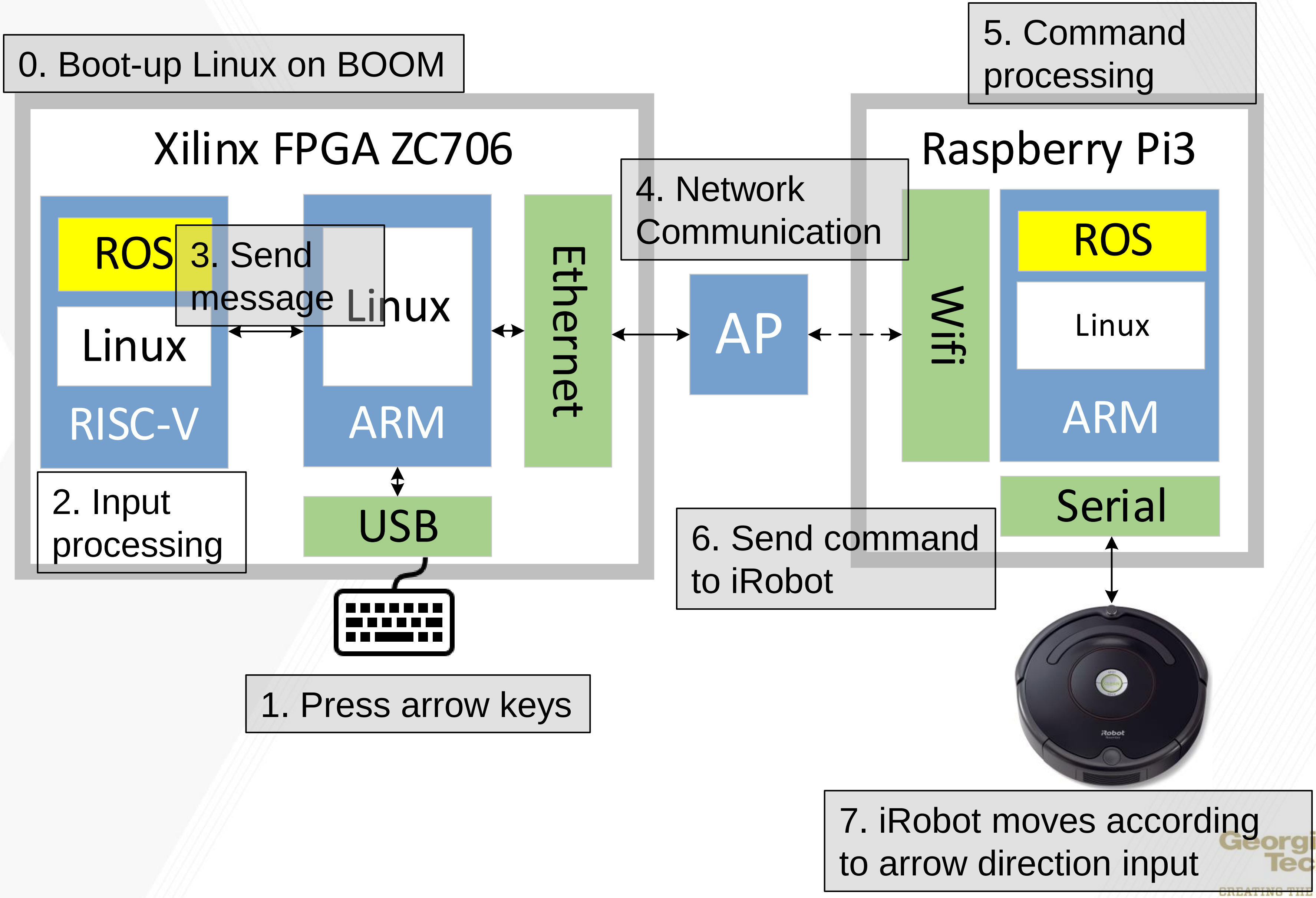


Example of ROS system configuration

Demo Setup

- Xilinx Zynq-7000 SoC FPGA board (zc706)
  - RISC-V BOOM core in tethered mode
  - ARM core supports network communication
  - Shared memory communication between two systems
- Raspberry Pi 3
  - Controlled by the Zynq board.
  - Connection setup
    - Wi-Fi to Access Point (AP)
    - Serial port to iRobot vacuum cleaner
- iRobot vacuum cleaner
  - Receiving commands from the Raspberry Pi

Demo Scenario



0. Boot-up Linux on BOOM

1. Press arrow keys

2. Input processing

3. Send message

4. Network Communication

5. Command processing

6. Send command to iRobot

7. iRobot moves according to arrow direction input

Challenges We Faced

For now, we can run Python on RISC-V BOOM and include all the dependency components of ROS. However the tethered version of the BOOM RISC-V core in the FPGA requires using a ramdisk instead of a normal flash-based disk

This Leads to a Critical Problem:

- GCC, Python and ROS take up too much space in SDRAM, so there is not enough memory space to run ROS.

Proposed Solution:

- We have looked at reconfiguring the Processing System (PS) kernel to allocate more memory for the Programmable Logic but this was not effective.
- We believe that we need to realize an untethered version of the RISC-V core by getting rid of the front-end Server (fesvr) mode.