

Machinekit & ROS-Industrial (2015-1-12)

Charles Steinkuehler

Machinekit HAL

- Virtual electronics lab-bench with real-world physical I/O
 - Lots of stock components (PID, stepgen, debounce, filters, etc)
 - Virtual wires (atomic values in shared memory)
 - Test equipment (HALScope, HALMeter)
- Dynamic environment
 - Components and signals can be added and deleted at run-time
 - Python and C bindings to create and interact with HAL objects
- Real-world connectivity to physical signals
 - Step/direction, Encoders, PWM, GPIO, and other I/O available
 - x86 support for LPT and “smart” FPGA I/O cards
 - Open-source VHDL code available!
 - BeagleBone supports GPIO driven by PRU for fine-grained timings

Connecting ROS to Machinekit

- ROS Message \leftrightarrow Machinekit HAL
 - Directly connect signals (atomic values)
 - Ring and triple buffers for signal groups
 - Protobuf messages
- Where to tie into ROS?
 - Value updates are easy (both directions)
 - “Middleware” is more complicated:
 - Joint homing & enforcing limits
 - Small time-scale path planning and closing servo loops
 - Can be in ROS or Machinekit or both
 - Reuse existing ROS code (ros_control?)

Example (Simple) Applications

- Physical stepper tracking JointTrajectory messages from ROS:

<https://youtu.be/b4O2KU2bLWE>

- Three stepper motors tracking 3D mouse:

<https://youtu.be/m0OeaTcWTZA>

- Machinekit HAL values → ROS Message:

https://github.com/mhaberler/ros_hello_machinekit