**Guide to Configuring Indistinct USB/UART Conflicts**

For many large-scale manufactured devices, the low-cost computing chips inside are often bought and integrated from another manufacturer. This can result in labeling and ID conflicts, especially when dictating a computer’s USB communications with multiple devices that have the same chip. For assigning device IDs, the computer cannot distinguish between identical chips (where all distinguishable information is attained), and the devices will either not function at all or only sporadically.

Our solution to this common issue is long-lasting and robust, where we utilize the chip’s software platform to modify a particular setting to differentiate between two devices, an RPLIDAR A1M8 with a Decawave DM1001 board. This unique setting can be implemented as the modified device’s ID, which can be configured to route the USB assignment, resulting in a functioning and unique computer system.

1. Download Simplicity Studios (version 5) from Silicon Labs onto a machine running Windows
   1. Windows is required, as Ubuntu build isn’t fully set-up, and several crucial functions for modifying USB/UART device IDs aren’t available as a result
   2. Both version 4 and 5 are available, however, only version 5 offers the necessary ID information access
2. After setting up an account on Simplicity Studios, connect the device through USB that’s going to be modified
   1. It should show up in “Connected Devices” with its identifying information (including serial number, which we will be changing)
   2. Press Start -> Create New Project -> Xpress Configuration Project
3. The new project, which consists of the device’s current information, has unset fields for IDs including: vendor ID, product ID, manufacturer string, product string, and serial string
   1. Changing the VID (vendor ID) or PID (product ID) is discouraged, so it’s best to change the serial string for dependable identification
      1. The serial string is listed as “an optional string that is used by the host to distinguish between devices with the same VID and PID combination”
4. Change the serial string to a random yet memorable number that wouldn’t be used by any Silicon Lab devices (eg. 0001, 0002 wouldn’t be safe IDs but 1111,2222,3333,4567 etc. would be)
5. Enter the string, press save, and program the new ID to the device
   1. This process should take under a minute
   2. It’s worth noting that some devices can only be programmed to once. Check if the device/chip are a one-time programmable set-up or multi-time programmable, as this affects the number of attempted modifications available
6. Now, the device’s serial ID is modified to a unique value that won’t conflict with the other, non-unique Silicon Labs devices
   1. Repeat this process n-1 times for n devices with the exact same chip
7. Transfer these edits to the corresponding device’s rules files with a new attribute: ATTRS{serial}==”updated serial number of device”
   1. The new rules files are:
      1. KERNEL==”ttyUSB\*”, ATTRS{idVendor}==”10c4”, ATTRS{idProduct}==”ea60”, ATTRS{serial}==”updated serial number of rplidar”, MODE:=0777, SYMLINK+=”rplidar
      2. KERNEL==”ttyUSB\*”, ATTRS{idVendor}==”10c4”, ATTRS{idProduct}==”ea60”, ATTRS{serial}==”0001”, MODE:=”0777”, SYMLINK+=”uwb”
      3. Repeat these edits with the required format for each device
8. Save and transfer these edits with the command “sudo udevadm control --reload-rules && udevadm trigger”
9. Unplug all of the devices, then plug them back in again
   1. To verify the process, the UWB and RPLiDAR symlinks /dev/usb and /dev/rplidar should both be visible with “ls /dev/uwb /dev/rplidar”
   2. UWBs can occasionally be defective - if the UWB symlink is missing, make sure the USB port can see the device with udevadm info --attribute-walk --path=/sys/bus/usb-serial/devices/ttyUSBn where n is bus 0,1, or 2
10. Replace all of the /dev/ttyUSB0 paths with /dev/rplidar for corresponding rplidar files (these were default paths initialized with the devices)
    1. All of the files to be edited are in ~/catkin\_ws/src/rplidar\_ros/launch
    2. Access all files for easy editing with “code .”, which opens vscode editor
11. Save these edits and recompile the system by running “cmake .. ; make -j” in ~/catkin\_ws/src/rplidar\_ros/launch
12. The UWB/RPLiDAR configurations can be verified by running roslaunch rplidar\_ros rplidar.launch and roslaunch uwb\_slam\_ros uwb\_node.launch