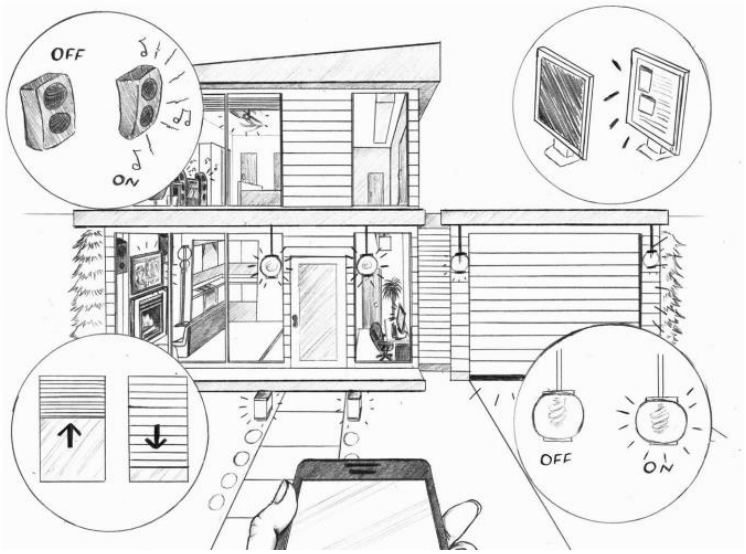



6LoWPAN

IPv6-based Low-power
Wireless Personal Area Networks

RIOT

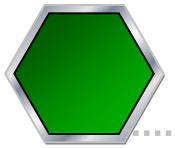
# Towards an Open Framework for Home Automation Development

ACOMP 2015

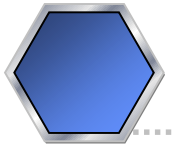
Ho Chi Minh City, Vietnam

**Pham Huu Dang Nhat**
Department of Computer Engineering
Ho Chi Minh City University of Technology
23 November 2015

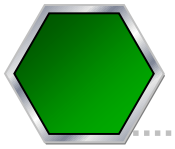
# Outline



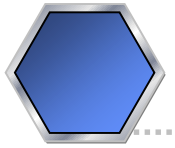
Introduction



Conventional Approaches



Proposed Framework



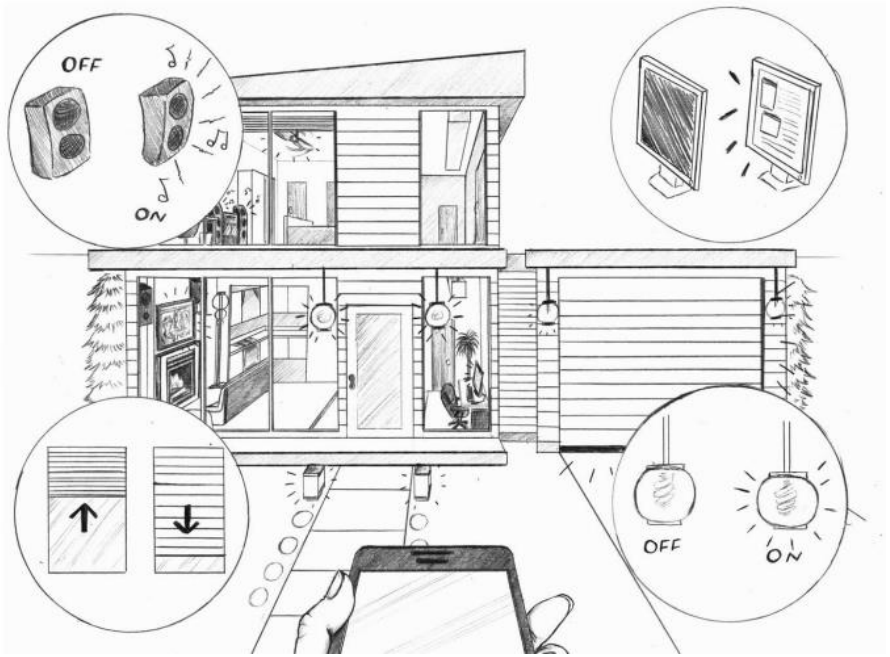
Prototype

Demo

# Introduction

General purpose computing platforms

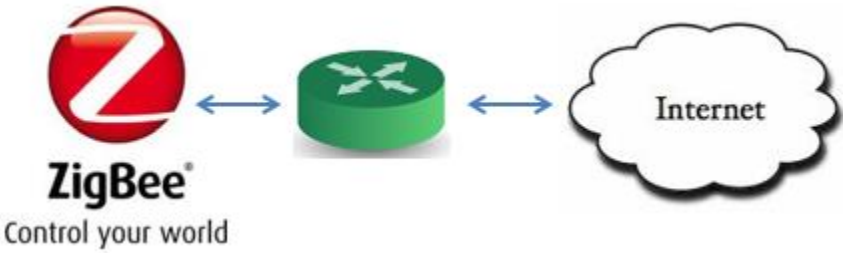
Sensors & Actuators




**Current problems**

Zigbee, Z-Wave, etc.
Inconvenient in connecting to the Internet

Support only a specific set of end devices

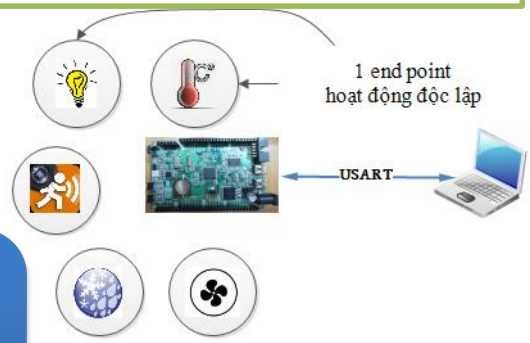



# Conventional Approaches

Automatic operations
Only system administrator/maintainer could configure

Based on 6LoWPAN (IPv6)

General framework for device drivers layer
Support various kinds of end-devices & **runtime configuration**.



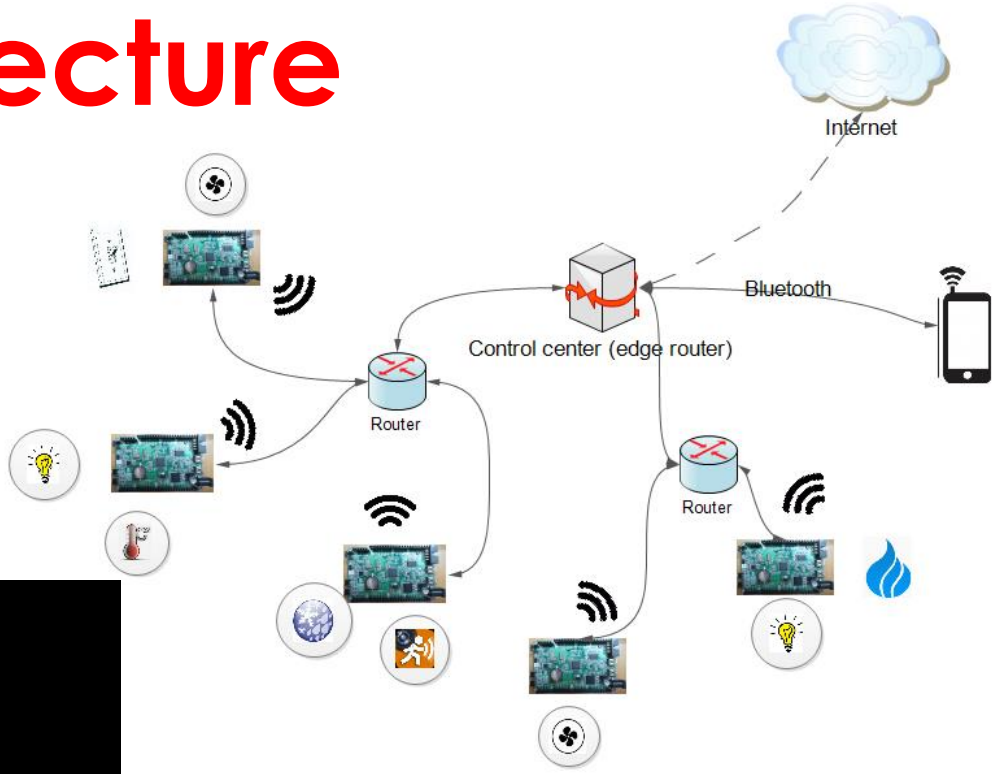

# Our Approaches

Both system administrators and users could configure scenarios for automatic operations

Remotely control end devices & setup scenarios.

# System Architecture

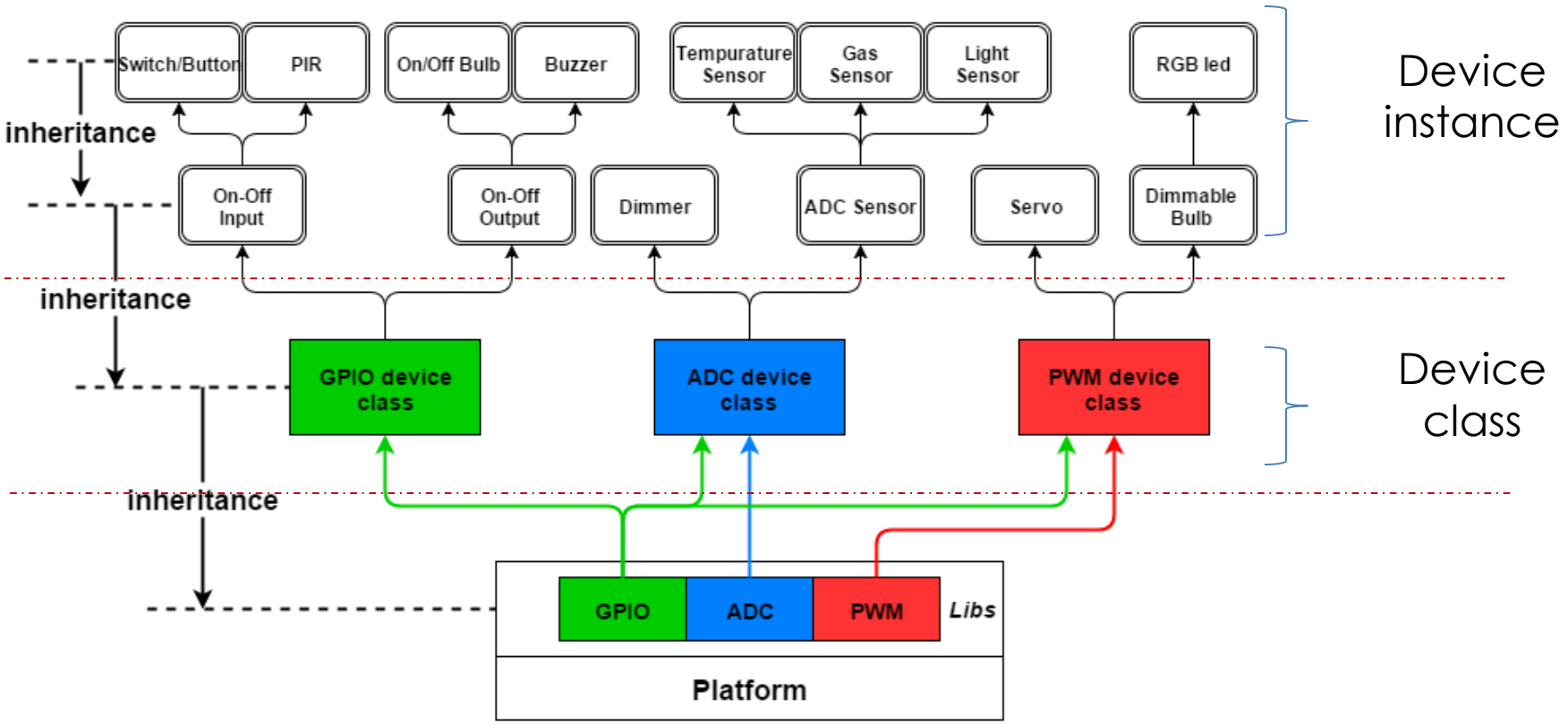
- Control Center (CC)
- Host Board (HB)




```
> mkdir new_dir
> ls
drw-    0       2014 03 22 09:17:27     SYSTEM~1
-rws    0       2014 03 26 20:31:01     DEV_LST
-rws    72      2014 03 26 08:03:09     SLP_CONF
drws    0       2014 03 26 20:31:03     NEW_DIR
> cd new_dir
> pwd
/NEW_DIR
```

- All data and control decisions are centralized → CC.
- Each HB could hosts many end points (OS's threads).
- End point → any supported end device. (Hot plugging)
- Shell (CLI) via USART → communicate with CC, HB.

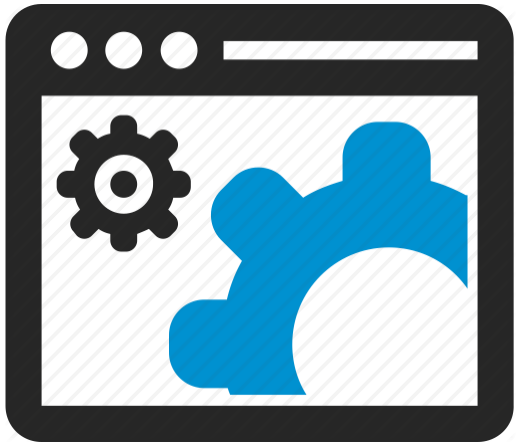
# Device Drivers Layer




- Adopt OOP & Inheritance model to achieve
  - Support many different kinds of end devices.
  - Runtime configuration
- Higher layers don't access directly to hardware. → **reusability**.
- C++ on Cortex-M microcontrollers.

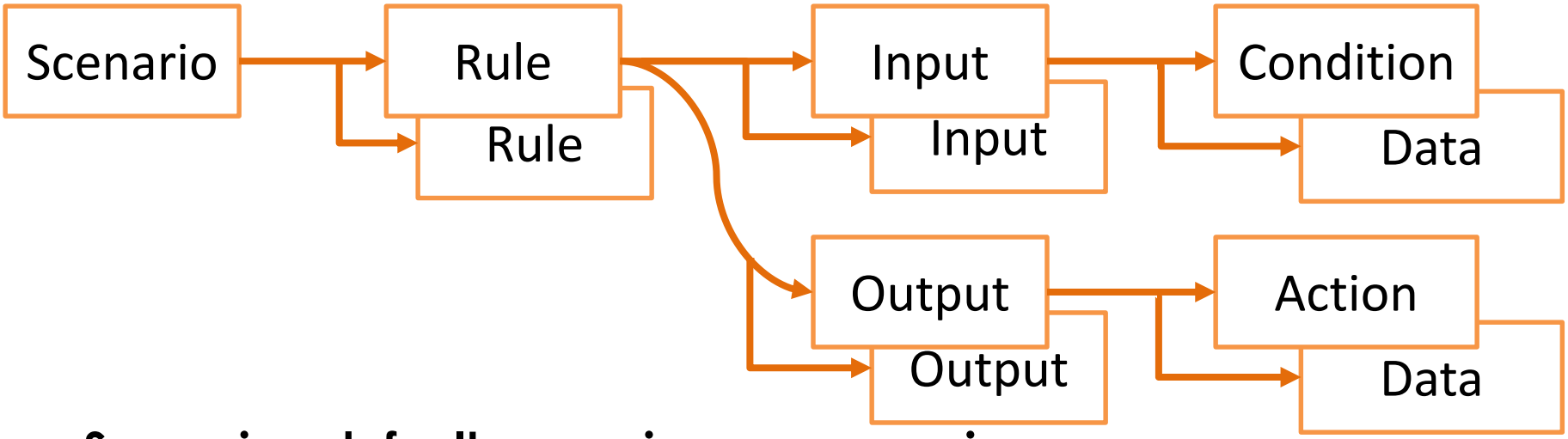
# Runtime Configuration

- Hot plugging new end device.
- Start running new end device using Shell commands.
  - What is the newly plugged in end device?
  - How does it work?
  - Where does it placed?
    - Which GPIOs?
    - Which ADC/TIM/… channels? etc.



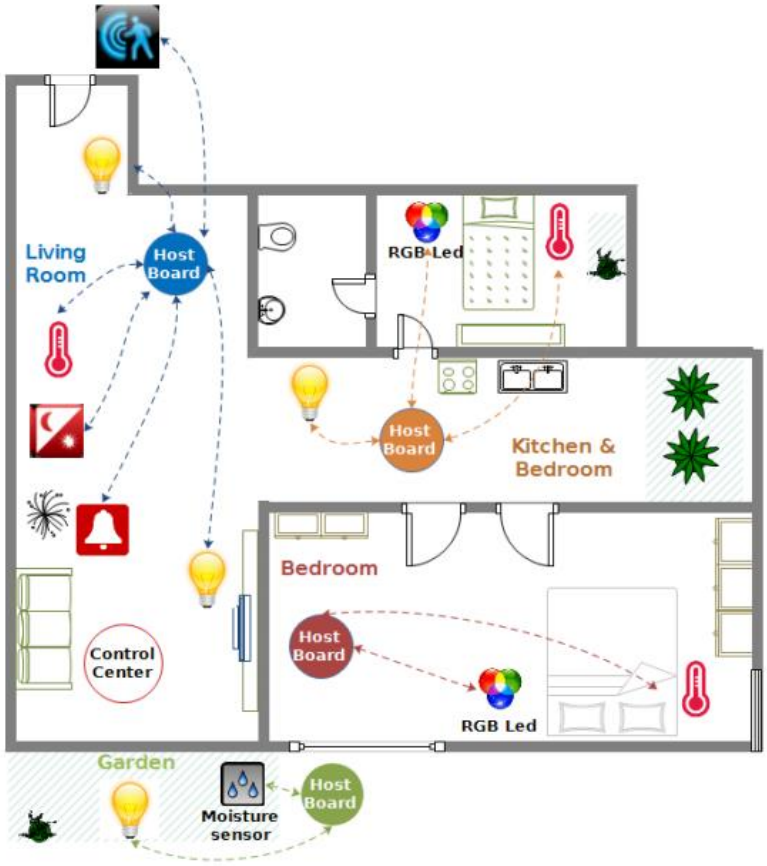
| Command | Options |
| --- | --- |
| senadc | -e [EP_id] -s [sensing condition]<br>-p [port] -n [pin] -a [adc] -c [channel]<br>-t [equations] -P [parameters]<br>-f [noise filtering value]<br>-u [alert under bound] -o [alert upper bound] |

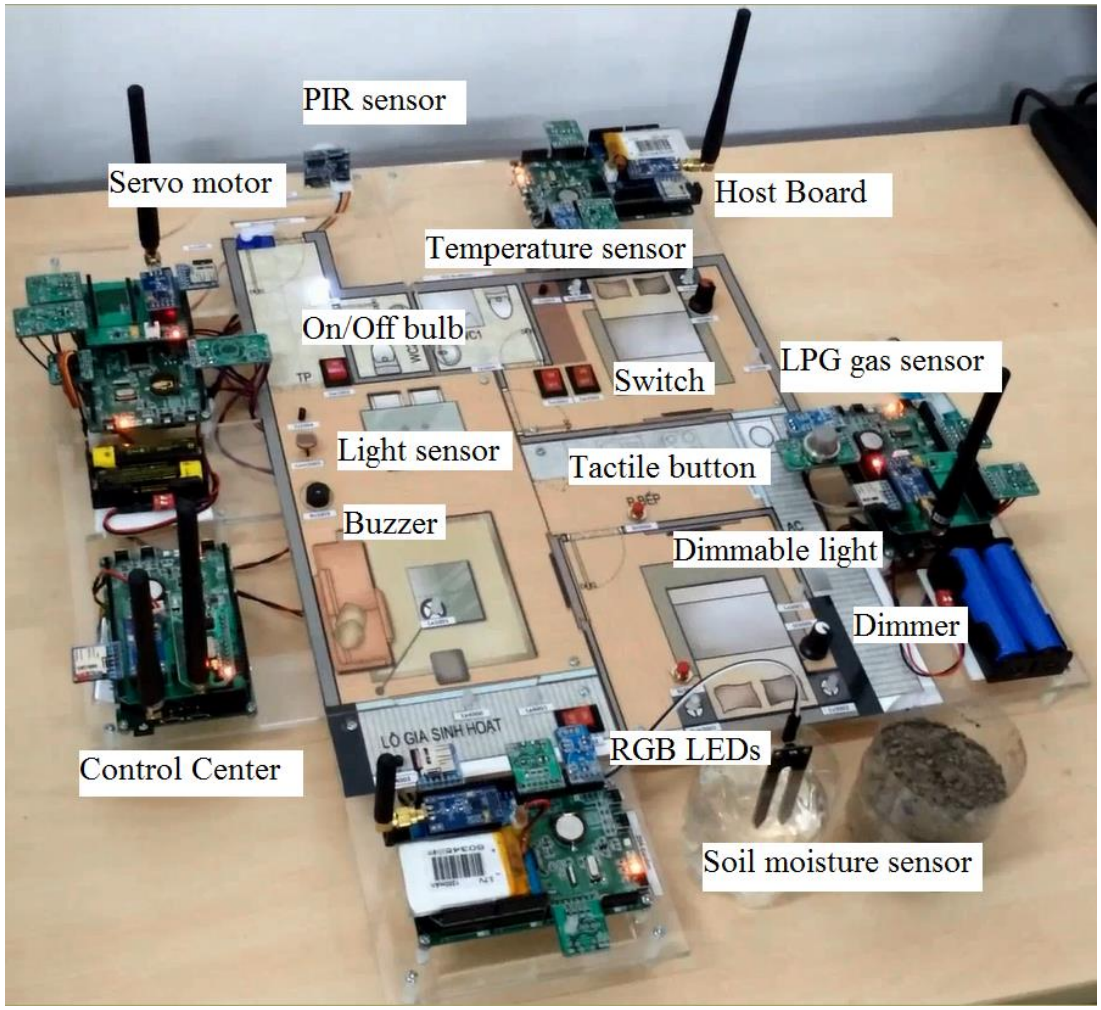
# Scenarios




- **Scenarios: default scenario, user scenarios.**
- **Supported conditions:**
  - The value of a device is =, <, <=, >, >= to a threshold.
  - The value of a device was changed.
  - The value of a device was changed over a threshold.
  - In a specific time period.
  - In a specific time period every day.
- **Supported action:**
  - Adjust the value of an end-device.

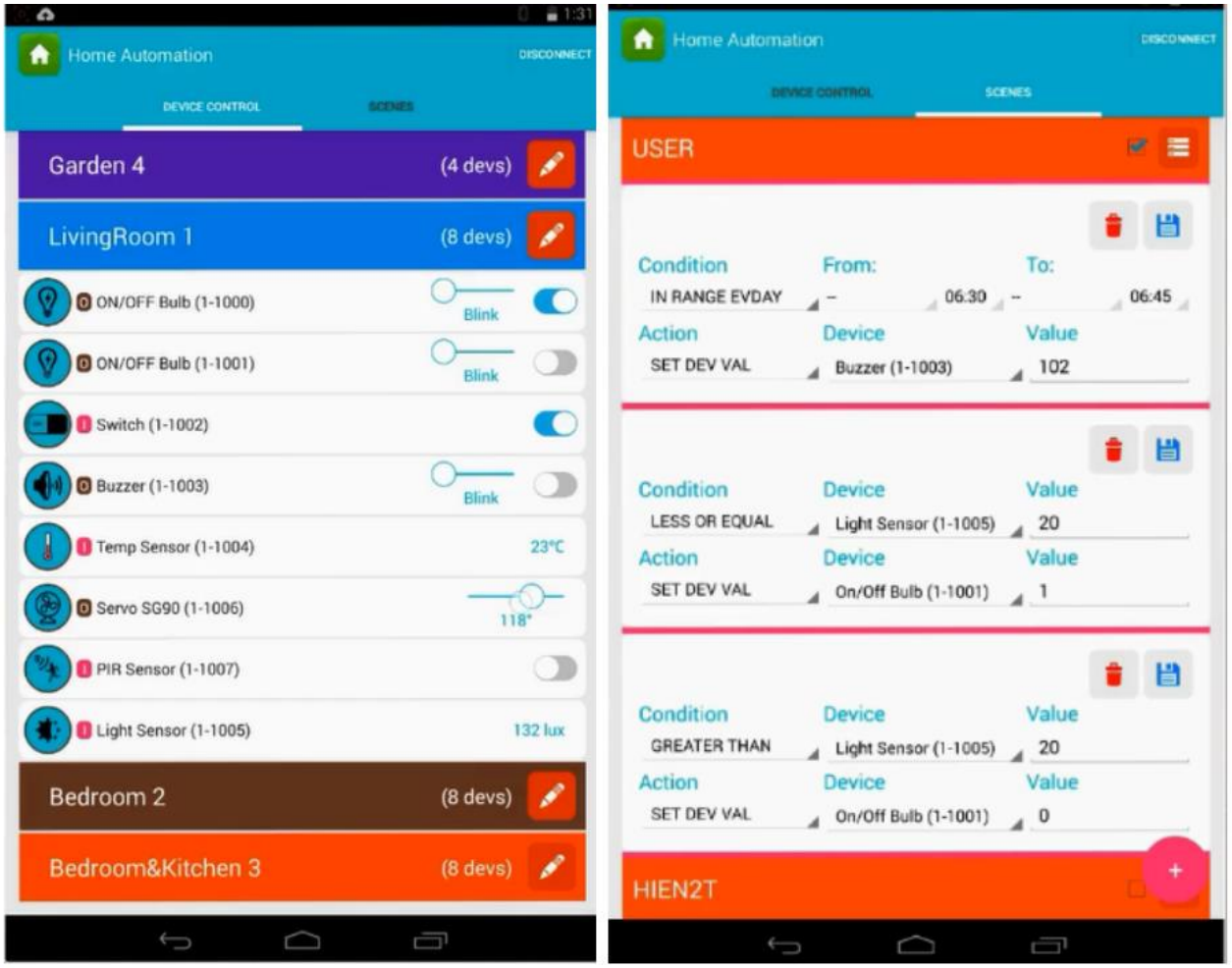
# Prototype




- Up to 25 rules for each scenarios, 3 inputs and 3 outputs for each rule.
- 28 end devices of 15 different types.
- Over 20 shell commands to manage network, scenarios, device configuration

# Android Application




- Remotely control end devices, manage (create/delete/rename) multiple scenarios and zones (rooms).

# Available Resource

- Source code: freely available on Github under LGPL v2.1 license at https://github.com/dangnhat/HA-project

- Demo video: available on Youtube at https://www.youtube.com/watch?v=3W_C0G6DxqU (title: ***"Home Automation System based on 6LoWPAN, RIOT-OS and BLE"***).

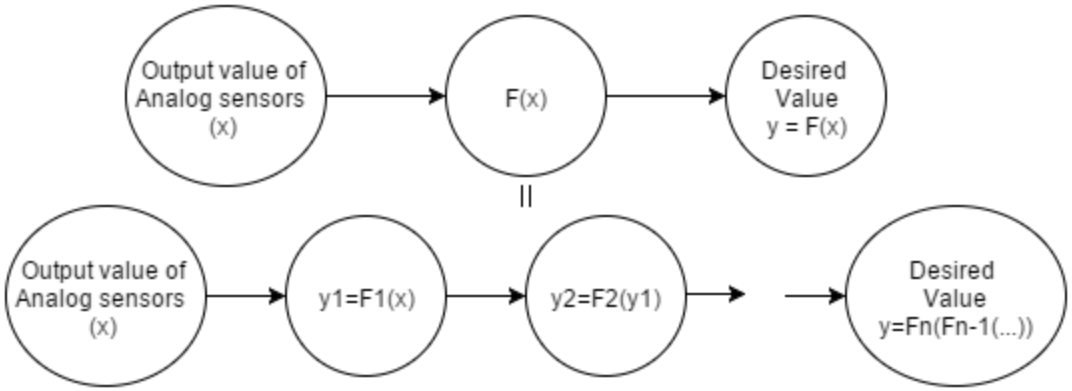

Start
DEMO


THANK

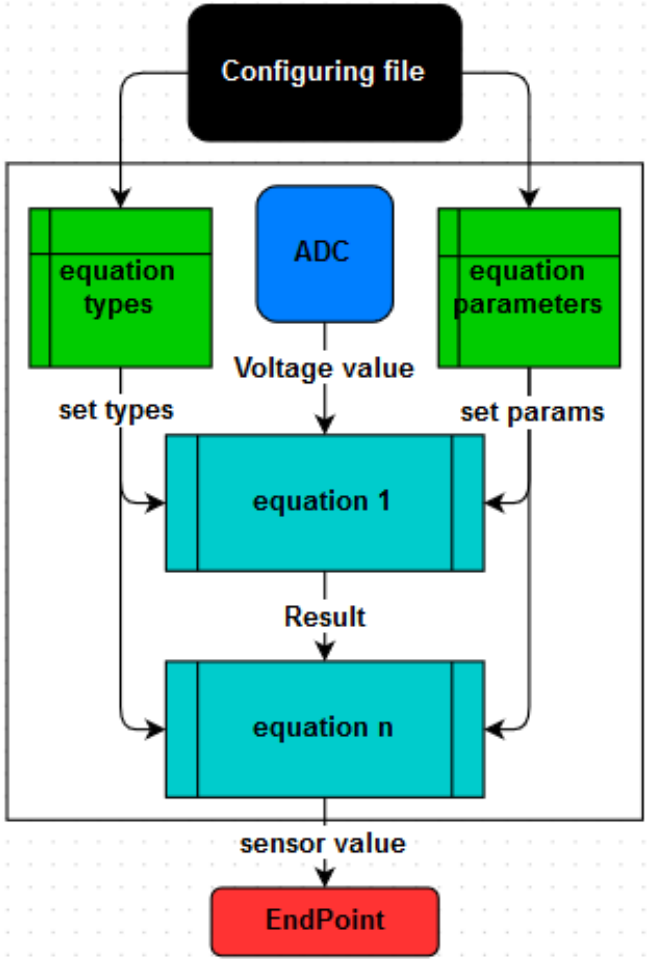

YOU


Q&A

# Analog sensor example




- $Y = F_n\ (F_{n\text{-}1}\ (\ldots\ (F_1(X))))$
- Supported equation types:
  - $Y = a.X^b + c$
  - $Y = \frac{1}{a.X+b} + c$
  - Look-up table.
- Combination of these functions.



# Analog sensor example

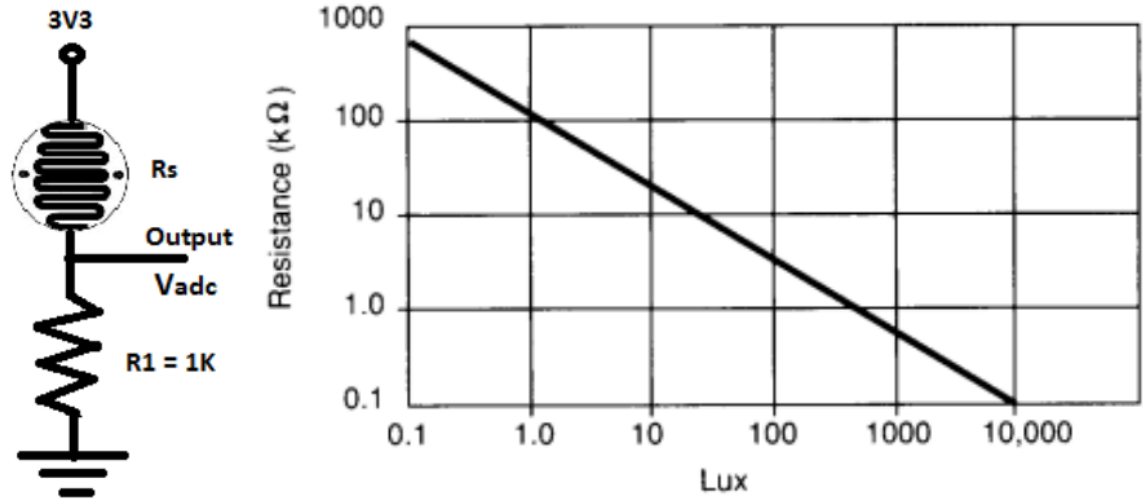



Figure 8. Schematic (left) and characteristic (right) of a photoresisitor

- Final output: L (lux)= $F_2(F_1(V_{adc})$
- $Rs = F_1(V_{adc}) = \frac{3.3 * R1}{Vadc} - R1(\Omega) = \frac{1}{0.303 * Vadc} - 1\ (k\Omega)$
- $L = F_2\ (Rs) = 10^{2.75} * Rs^{-1.25}(\text{lux})$
  - L : light intensity (lux)
  - Rs : photo-cell's resistor