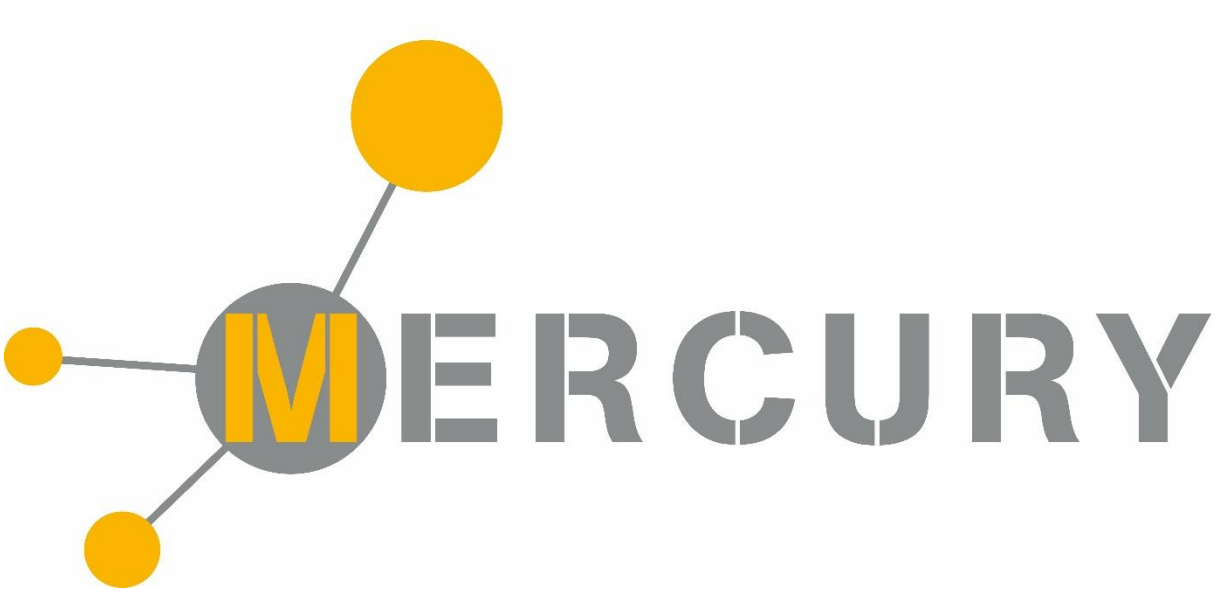
Mercury System



A Modular HW/SW Development System for IoT and Connectivity Applications

Connectivity & IoT

- **IoT** : *"Is the internetworking of physical devices, vehicles, buildings and other items, embedded with electronics, software, sensors, actuators, and network connectivity, that enable these objects to collect and exchange data"*
- **Connectivity** : *"the means by which individual terminals, computers, mobile devices, and local area networks connect to each others"*

So what…???

- By definitions, to make an IoT/Connectivity enabled object we need to:
  - **Connect** the object to the **network**,
  - Add a **programmable logic** where we can put our software,
  - Add **sensors** and/or **actuators**,
  - Realize the **software** which implements the desired IoT **application**,
  - Someway **power** the system, with the appropriate **power requirements**.

How to do this

- So far, **seems not so easy** to develop this, so called, IoT application.
- The application idea itself is not sufficient…we need to **connect** the device, to add complext **electronics** and **software** components, and probably we need some manual skills to build up the **prototype** of the “thing”.

Mercury System

- The Mercury System solve this problem.
- It is a modular HW/SW development platform composed by:
  - A **main unit** to implement the **programmable logic block**,
  - Different **networks units** to add the **connectivity**,
  - Several **slave units** to add various types of **sensors** and **actuators**
  - Some **power units** to satisfy system **power supply** requirements
  - A **SW framework** able to abstract all the low level programming issues and live to the user only the **application logic development** task

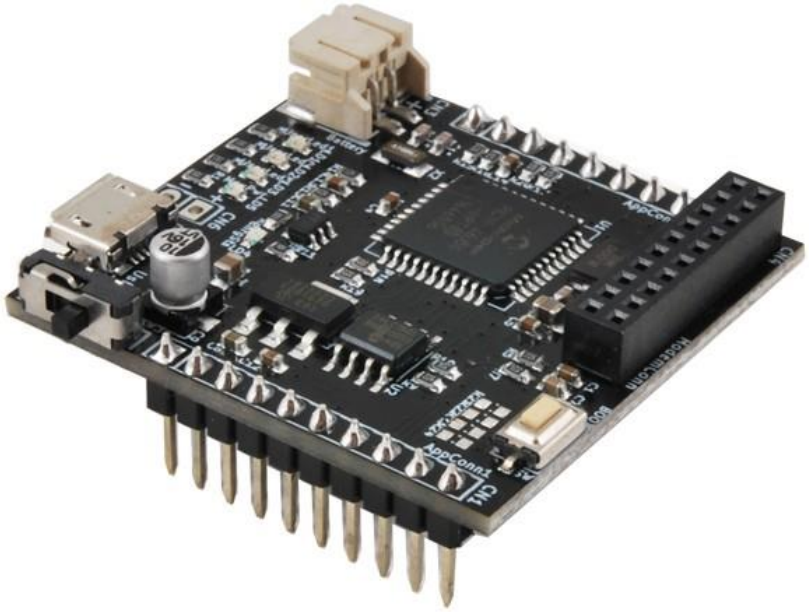
System Components (1)

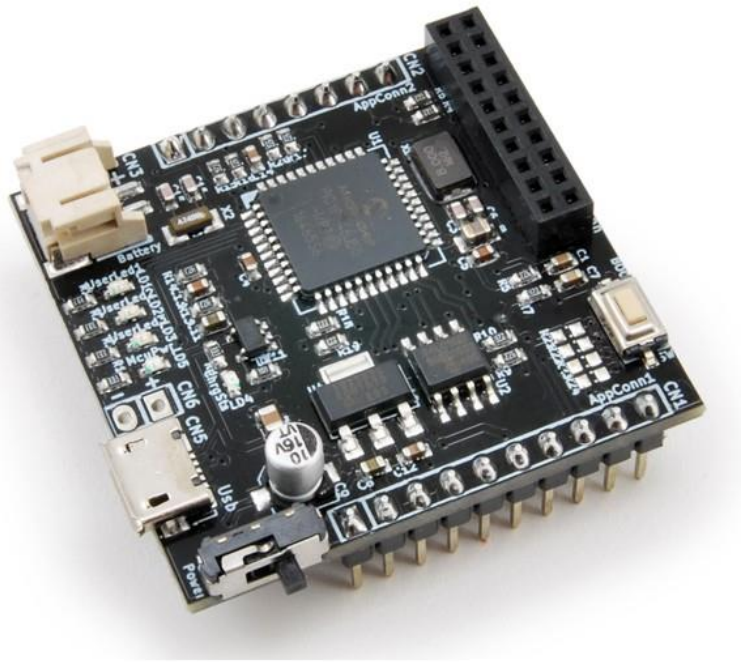
- The system is composed by 6 types of boards:
  - **Base Board (BB):** is the “brain” of the system, and hosts the programmable logic, as well as a battery connection with a recharge unit.
  - **Modem Board (MB):** is the bridge which allows the connection to the network. There are different versions depending on the communication standard used (GSM/GPRS, WiFi, BT). It connects directly to the BB.
  - **Power Board (PB):** is the board used to meet the power requirement of the system.

System Components (2)

- **Slave Board (SB):** there are many of them, and each one hosts a particular sensor/actuator, and a local logic to make it easier the connection with the BB and the application SW.
- **Expansion board (EB):** this family of board allow to easily connect the entire system, providing different socket to host BB, PB and SB, as well as backbone communication buses for local communication (e.g. between BB and SBs).
- **Brain-Less Board (BL):** the simplest board without local controller. For cost-sensitive applications.

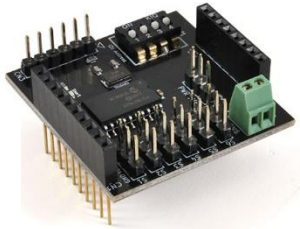
Example – BB110



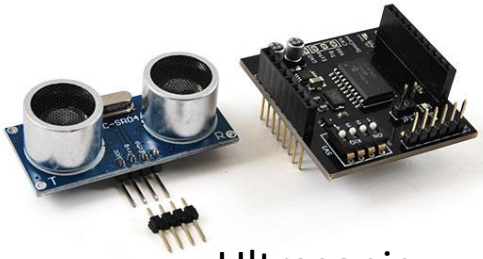


Example – BB110 + GSM/GPRS MB

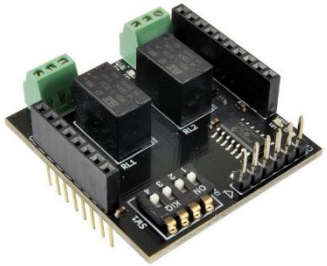
Example – BB110 + SBs



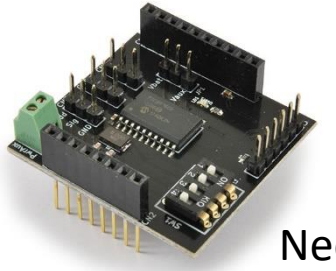
Servo



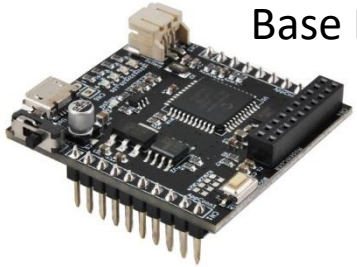
Ultrasonic



Relay

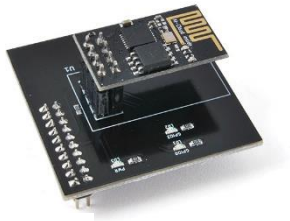


Neopixel

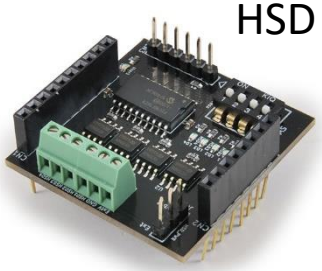


Base Board

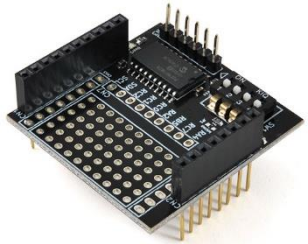
BT Modem



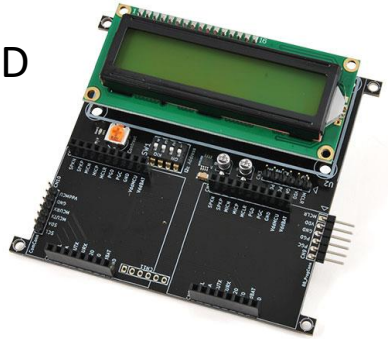
WiFi Modem



HSD



Proto



Exp. LCD

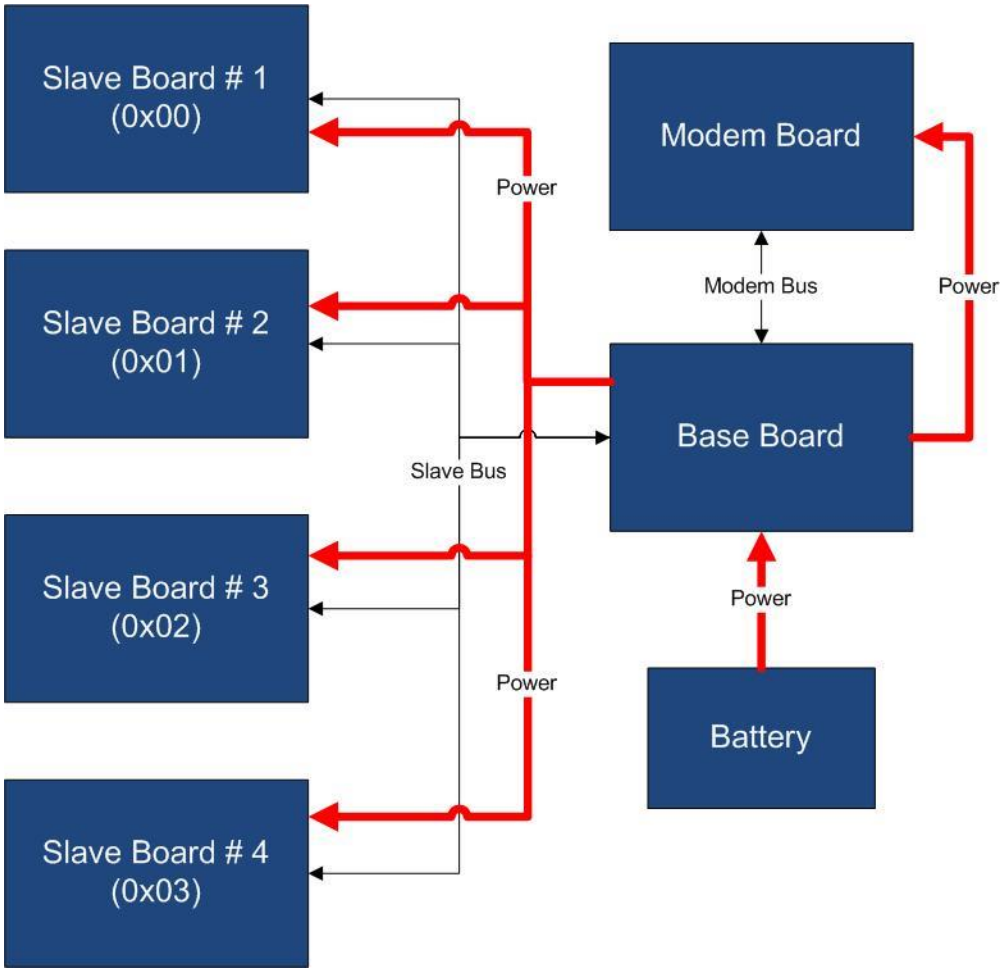
System Setup

- The typical setup of a Mercury System comprises a **BB** connected to a **MB**, and to a variable numbers of **SBs** or **BLs**, depending on the application.
- **PB** and **EB** are optional, depending on power and space allocation constraints.

System Connections

- BB, MB and SB operate as bus connected units, where MB and SB act as **slaves** of the BB.
- The typical connection setup foresees a **dedicated connection** between BB and MB, and **multidrop connection** between BB and SBs.
- SBs are **“intelligent slaves”,** where each unit is populated with a local controller implementing an high level instruction set that is used to easily control the slave.

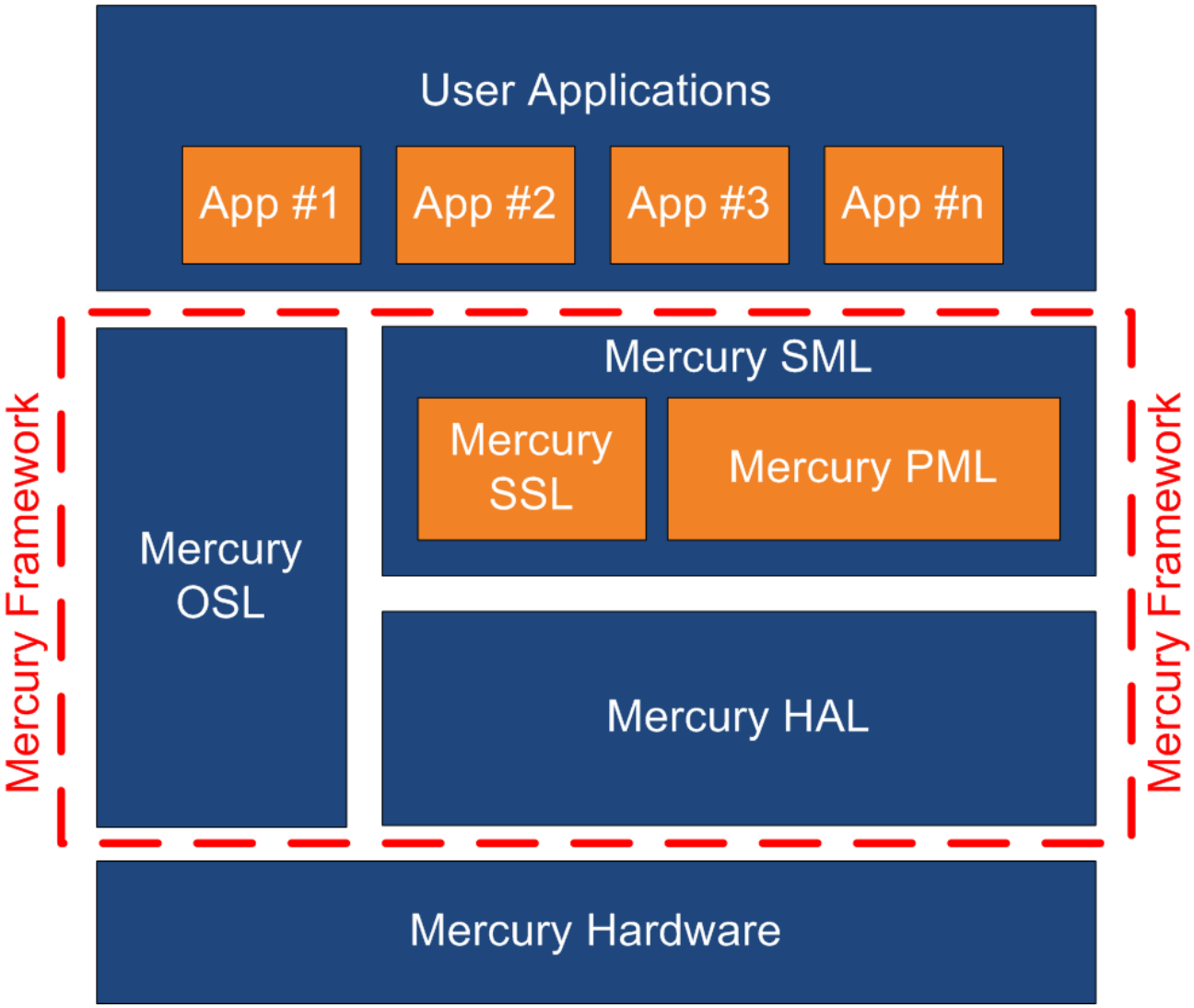
Connection Example



SW Framework

- The Mercury SW Framework allows to develop user application easily.
- User doesn’t want to get bored with handling of communication protocol to connect BB and MB ans SBs, he/she doesn’t want to take care of configurations, initializations and so on…
- The framework takes care about this boring jobs, and provides to the developer an high level abstraction of the Mercury System.

Framework Architecure

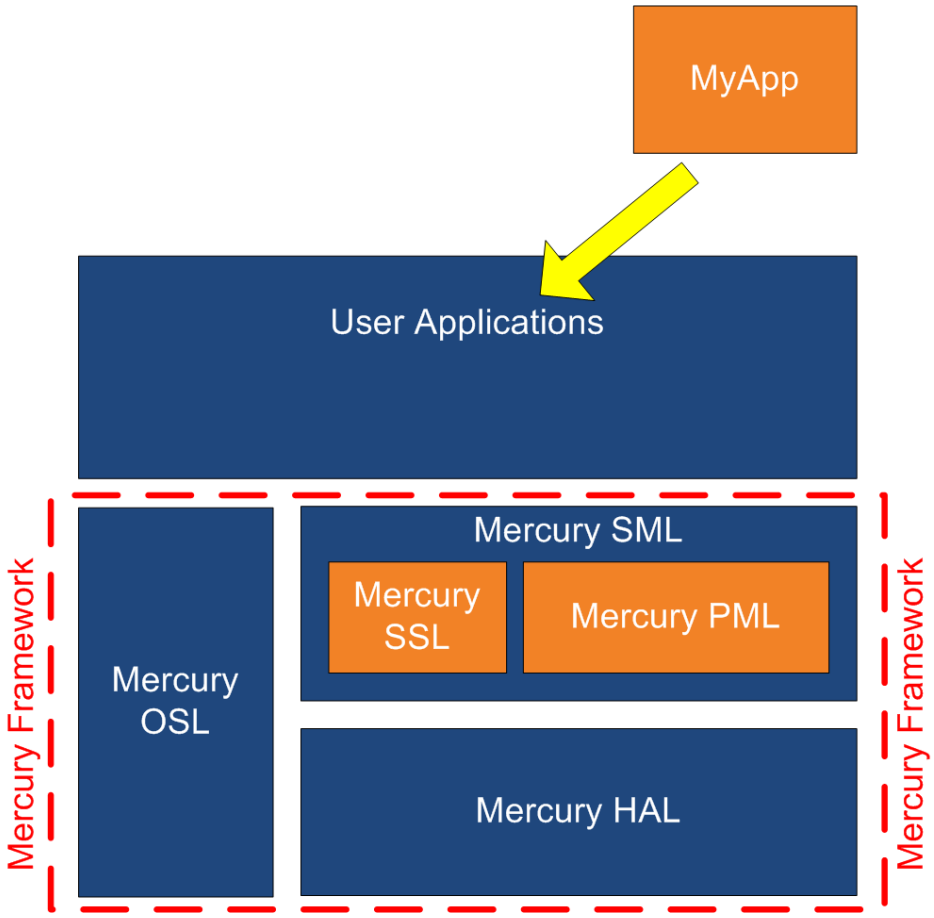


SW Framework Components

- The SW Framework is composed by few an simple layers:
  - An **HAL** (Hardware Abstraction Layer) which takes care about the low level hardware abstractions.
  - A **SML** (System Management Layer) which takes care of abstracts the details of the protocols used to communicate with MB and BBs, by meas of providing a simple API set to access these services, and also provides some basic services like power managemet, internal peripheral management, terminal, etc.).
  - A **Ligthweight OS** (Operative System layer) which has the purpose of correctly schedule the various tasks running on the BB (included the Application Task) with the right timing.

User Application

- The user application sits on top of the framework and exploits the provided services in order to implements the application specific logic.



Where to buy?

- The Mercury System is distributed by **Futura Group**.
- Visit **Futurashop** website for more information:
- https://www.futurashop.it/sistemi%20sviluppo-software_didattica-libri-documentazione_tecnica/software-e-sistemi-di-sviluppo/iot-internet-of-things

Examples

- BT Moodlamp: https://www.youtube.com/watch?v=Co3H3hVcO5o
- GSM/GPRS Intrusion detector: https://www.youtube.com/watch?v=JjRrdgCXYrM
- IoT Meteo Station: https://www.youtube.com/watch?v=HAflxVgQlsk&feature=youtu.be
- BT Controlled Rover: https://www.youtube.com/watch?v=ULyeNoNSruA
- Irrigation System with BT interface: https://www.youtube.com/watch?v=ipqQ0NbGfG0

Thanks ☺

THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
DANKSCHEEN
BİYAN
SHUKRIA
TINGKI
YAQHANYELAY
TASHAKKUR ATU
SUKSAMA
EKHMET
GRAZIE
MEHRBANI
PALDIES
KOMAPSUMNIDA
MAAKE
GOZAIMASHITA
EFCHARISTO
JUSPAXAR
BOLZÏN
MERCI