Chapter 7

IoT Physical Devices & Endpoints
Outline

• Basic building blocks of an IoT Device
• Exemplary Device: Raspberry Pi
• Raspberry Pi interfaces
• Programming Raspberry Pi with Python
• Other IoT devices


Bahga & Madisetti, © 2015
What is an IoT Device

• A "Thing" in Internet of Things (IoT) can be any object that has a unique identifier and which can send/receive data (including user data) over a network (e.g., smart phone, smart TV, computer, refrigerator, car, etc.).

• IoT devices are connected to the Internet and send information about themselves or about their surroundings (e.g. information sensed by the connected sensors) over a network (to other devices or servers/storage) or allow actuation upon the physical entities/environment around them remotely.
IoT Device Examples

• A home automation device that allows remotely monitoring the status of appliances and controlling the appliances.
• An industrial machine which sends information about its operation and health monitoring data to a server.
• A car which sends information about its location to a cloud-based service.
• A wireless-enabled wearable device that measures data about a person such as the number of steps walked and sends the data to a cloud-based service.
Basic building blocks of an IoT Device

• Sensing
  • Sensors can be either on-board the IoT device or attached to the device.

• Actuation
  • IoT devices can have various types of actuators attached that allow taking
    • actions upon the physical entities in the vicinity of the device.

• Communication
  • Communication modules are responsible for sending collected data to other
    devices or cloud-based servers/storage and receiving data from other devices
    and commands from remote applications.

• Analysis & Processing
  • Analysis and processing modules are responsible for making sense of the
    collected data.
Block diagram of an IoT Device
Exemplary Device: Raspberry Pi

• Raspberry Pi is a low-cost mini-computer with the physical size of a credit card.

• Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do.

• Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins.

• Since Raspberry Pi runs Linux operating system, it supports Python "out of the box".
Exemplary Device: Raspberry Pi

- Raspberry Pi is a low-cost mini-computer with the physical size of a credit card.
- Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do.
- Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins.
- Since Raspberry Pi runs Linux operating system, it supports Python "out of the box".
Linux on Raspberry Pi

- **Raspbian**
  - Raspbian Linux is a Debian Wheezy port optimized for Raspberry Pi.
- **Arch**
  - Arch is an Arch Linux port for AMD devices.
- **Pidora**
  - Pidora Linux is a Fedora Linux optimized for Raspberry Pi.
- **RaspBMC**
  - RaspBMC is an XBMC media-center distribution for Raspberry Pi.
- **OpenELEC**
  - OpenELEC is a fast and user-friendly XBMC media-center distribution.
- **RISC OS**
  - RISC OS is a very fast and compact operating system.
Raspberry Pi GPIO

- 3V3
- GPIO 2 (I2C SDA)
- GPIO 3 (I2C SDL)
- GPIO 4
- GROUND
- GPIO 17
- GPIO 27
- GPIO 22
- 3V3
- GPIO 10 (SPI MOSI)
- GPIO 9 (SPI MISO)
- GPIO 11 (SPI SCLK)
- GROUND
- 5V
- 5V
- GROUND
- GPIO 14 (UART TxD)
- GPIO 15 (UART RxD)
- GPIO 18
- GROUND
- GPIO 23
- GPIO 24
- GPIO 25
- GPIO 8 (SPI CE0 N)
- GPIO 7 (SPI CE1 N)

Raspberry Pi Interfaces

• Serial
  • The serial interface on Raspberry Pi has receive (Rx) and transmit (Tx) pins for communication with serial peripherals.

• SPI
  • Serial Peripheral Interface (SPI) is a synchronous serial data protocol used for communicating with one or more peripheral devices.

• I2C
  • The I2C interface pins on Raspberry Pi allow you to connect hardware modules. I2C interface allows synchronous data transfer with just two pins - SDA (data line) and SCL (clock line).
from time import sleep
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

#Switch Pin
GPIO.setup(25, GPIO.IN)

#LED Pin
GPIO.setup(18, GPIO.OUT)
state=False

def toggleLED(pin):
    state = not state
    GPIO.output(pin, state)

while True:
    try:
        if (GPIO.input(25) == True):
            toggleLED(pin)
        sleep(.01)
    except KeyboardInterrupt:
        exit()
Other Devices

• pcDuino
• BeagleBone Black
• Cubieboard