“Event driven O.S.”
Z-Stack: a complete protocol stack conforming to ZigBee standard

Giovanni Perbellini
Agenda

• Overview
  – 802.15.4 + Zigbee
• Z-Stack
  – Event-driven programming
• Texas Instrument (TI) developer kit
• Z-Stack Application development
• Example 1: uartSOA (Event driven application)
• Example 2: GenericApp
  – Deploy in Flash
  – Run
• Example 3: SimpleApp (Temperature app.)
What is ZigBee?

- A high level communication protocol using small, low-power digital radios based on the IEEE 802.15.4 standard for wireless networks
- ZigBee is targeted at applications that require a
  - Low data rate
  - Long battery life
  - Secure networking
- ZigBee 3.0 on Q4 of 2015
  - Simplify IoT and Smart Home communication
The ZigBee Alliance

- Organized as an independent, non-profit corporation in 2002
  - Includes major names in Semiconductor, Software Developer, End Product Manufacturer and Service Provider Industries
- Open and global
- Activity includes
  - Specification creation
  - Certification
  - Market development and user education
The ZigBee Promoters

...plus over 225 member companies from around the world
ZigBee Stack Architecture

Application
- Initiate and join network
- Manage network
- Determine device relationships
- Send and receive messages

Security functions
- Network organization
- Route discovery
- Message relaying

Medium Access (MAC)
- Device management
- Device discovery
- Service discovery

Physical Radio (PHY)
- Device binding
- Messaging

Application
- ZDO
- NWK
- App Support (APS)
- SSP
802.15.4

- Standard to define the Physical and MAC layer of a WPAN (Wireless Personal Area Network)
  - **GOAL:** low power and low cost
- Physical: Operates on one of three possible frequency
  - 868MHz – 1 channel
  - 900 MHz – 10 channel
  - 2.4 GHz – 16 channel
- MAC: manages access to the physical channel
- Upper layers
  - ZigBee, WirelessHART, 6LoWPAN, etc.
- Node identifier: 64 bit (named IEEE address)
802.15.4

- Node identifier: 64 bit (named IEEE address)

**Star**
- Full function device (FFD)
  - Any topology
  - PAN coordinator capable
  - Talks to any other device
  - Implements complete protocol set

**Cluster Tree**
- Reduced function device (RFD)
  - Limited to star topology or end-device in a peer-to-peer network.
  - Cannot become a PAN coordinator
  - Very simple implementation
  - Reduced protocol set
ZigBee Node Types

- **802.15.4 FFD → ZigBee Coordinator**
  - One required for each ZB network
  - Initiates network formation and stores information about the network

- **802.15.4 FFD → ZigBee Router**
  - Participates in routing of messages

- **802.15.4 RFD → ZigBee End Device**
  - Enables very low cost solutions
ZigBee Network Topologies

Star

Cluster Tree

Mesh

- ZigBee Coordinator
- ZigBee Router
- ZigBee End Device
Application Profiles

- Application profiles define what messages are sent over the air for a given application
- Devices with the same application profiles interoperate end to end
Why Do We Need Profiles?

• Need a common language for exchanging data
• Need a well defined set of processing actions
• Interoperability across different manufacturers
• Simplicity and reliability for end users
• Consumer flexibility for products
ZigBee Profile

- **Home Automation**
  - Supports a variety of devices for the home including lighting, heating and cooling, and even window blind control

- **Smart Energy**
  - More energy efficient home environment

- **Telecommunication Services**
  - Enabling information delivery, mobile gaming, location based services, secure mobile payments, mobile advertising, zone billing, mobile office access control, payments, peer-to-peer data sharing services

- **Health Care**
  - To monitor non-critical information on people in hospitals, nursing homes, and in their own home with no wires

- **Remote Control**

- **Specifications profile under development**
  - Building automation
  - Retail services
Home Automation Profile

• Management of lighting, heating and cooling system from anywhere in your home
• Automate control of multiple home systems to improve conservation, convenience and safety
• Embed intelligence to optimize consumption of natural resource
• Install, upgrade and network home control system without wires
• Easily install wireless sensors to monitor a wide variety of conditions
• Receive automatic notification upon detection of unusual events
Working phases

• **Network formation**: the Coordinator creates the network to which all other nodes will connect.

• **Join**: performed by non-coordinator nodes after powering on. The network topology is created (star, tree). Each node sends a message to the Coordinator or to a Router which registers the node as a child.

• **Binding**: creation of a logical link between two or more end systems for data exchange.

• **Data transmission**: data exchange between end devices or with the Coordinator.
ZigBee stack implementation

- Z-Stack provided by Texas Instrument
  - Next slides
- BitCloud provided by Atmel
  - Atmel BitCloud is a full-featured ZigBee PRO stack supporting reliable, scalable, secure wireless applications running on Atmel wireless platforms
- BeeStack provided by Freescale
- EmberZNet ZigBee PRO provided by Ember
- Xbee provided by Digi International
Event-driven programming (I)

- Paradigm for small-footprint embedded systems with significant memory constraints
- Refers to programming style which each invocation of an API function is delivered through a callback associated with the initial request
- All “user code” executes in a callback either a priori known to the system or registered with the stack by the user application
- An event driven system typically runs an event loop, that keeps waiting for such activities, e.g. input from devices or internal alarms
Event-driven programming (II)

Event emitters
(e.g., OTA message, press button, click)

Event queue

Event loop
(process event)

Event handler (callback)
Z-Stack (I)

- A complete protocol stack conforming to ZigBee Alliance standards
- Provided by Texas Instrument
- Microsoft Windows-based Free Software
- Includes all layers of ZigBee stack (IEEE 802.15.4 layer, ZigBee layer, O.S. layer)
- Includes support for SmartEnergy Profile and Home Automation Profile (Z-Stack 2.X)
Z-stack (II)

- HAL (Hardware abstraction layer)
- OSAL (Operating system abstraction layer)
- ZigBee Stack + IEEE 802.15.4 MAC
- User Application
- MT (Monitor Test) – Used to communicate with a PC-based test tool via the UART
- ZigBee 2006
  - Last version: ZStack-1.4.3 (CC2430/31)
- ZigBee 2007 (ZigBee and ZigBee PRO)
  - ZigBee PRO: routing, security, fragmentation, etc.
  - Last version: Z-Stack-2.3.1 (CC2530)
OSAL - Functionalities

• Task registration, initialization, starting
• Message exchange between tasks
• Task synchronization
• Interrupt handling
• Timers
• Memory allocation
OSAL – SW layer

Applications

Z-Stack

OSAL API

OSAL
[Event, ...]

Target independent

Target specific

Hardware Abstraction Layer (HAL)
[Timer, LCD, ...]

Interrupt
[ISR]

Target Hardware
[CC2430, CC2530, ...]
Z-Stack Application Development

• Be event driven
  – Local (key stroke, sensor alarm)
  – Remote (remote message from another device)
  – Timers

• Follow a precise schema
• Interact with the ZigBee stack
Z-Stack Application structure

- Each application must contain:
  - Init method
    - The application must provide information to the ZigBee stack
      - Node descriptor
      - Hardware specific information
      - Profile information
  - Event-loop method
    - The event loop method is invoked whenever an event is transmitted to the application
      - Mandatory (system events)
      - Application specific (timers)

- Applications can contain
  - Local functions
  - Callbacks
    - to process specific events
Event processing

- Events are received from the ZigBee stack
- Application events are then transmitted to application level
- The corresponding callback or function for each event is activated
Texas Instrument: Hardware (I)

**Chipcon SmartRF04EB Evaluation Boards**
- Includes a USB and Serial interfaces
- Fitted with a CC2430EM
- LCD panel
- Can be powered from:
  - Batteries
  - Over the USB interface
  - The 9V DC connectors
- Includes a variety of sensors
  - Potentiometer
  - Pushbuttons and joystick
  - 2 LEDs
  - Audio filter and amplifier enabling transmission and reception of audio signals
Texas Instrument: Hardware (II)

**CC2430DB Demonstration Boards**
- Includes a USB interface
- Can be powered from:
  - Two AA batteries
  - Over the USB interface
  - The 9V DC connectors
- Includes a variety of sensors
  - Light sensor
  - 2-way accelerometer
  - Temperature sensor
  - Battery monitor
  - Potentiometer
  - Pushbutton and joystick
Texas Instrument: Hardware (III)

**Chipcom SOC BB Battery Board**
- Fitted with a CC2430EM
- Can be powered from:
  - Two AA batteries
- Includes a variety of sensors
  - Temperature sensor
  - Battery monitor
  - Pushbutton
IAR C-compiler

• IAR Embedded Workbench (EW8051) suite
• It supports project management, compiling, assembling, linking, downloading, and debugging for various 8051-based processors
  – Chipcon CC243x family
Example: “uartSOA” (i)

- The uartSOA example write on UART port on bottom pressed
- Target: CoordinatorEB
Example: “uartSOA” (ii)

- `uint8 HalUARTOpen (uint8 port, halUARTCfg_t *config);`
  - `port` – specified serial port to be opened
  - `config` – Structure that contains the information that is used to configure the port

```c
typedef struct {
    bool configured;
    uint16 baudRate;
    bool flowControl;
    uint16 flowControlThreshold;
    uint8 idleTimeout;
    uint16 rx;
    uint16 tx;
    bool intEnable;
    uint32 rxChRvdTime;
    halUARTCBack_t callBackFunc;
}halUARTCfg_t;
```
Example: “uartSOA” (iii)

- `uint16 HalUARTWrite (uint8 port, uint8 *buf, uint16 length);
  - port – specified serial port that data will be read
  - buf – buffer of the data.
  - length – the length of the data
Example: “uartSOA” (iv)

```c
void uartSOA_Init( uint8 task_id ) {
    ...
    uartConfig.baudRate  = UART_SOA_BAUD;
    HalUARTOpen (UART_SOA_PORT, &uartConfig);
}

UINT16 uartSOA_ProcessEvent( uint8 task_id, UINT16 events ) {
    ...
    while ( event) {
        switch (event ) {
            case KEY_CHANGE:
                callback;
                break;
            case EVENT2:
                callback2;
                break;
            default:
                break;
        }
    }
}
```
Project File for IAR
Target selection

Coordinator (Demonstration Board Evaluation Board)

Router (Demonstration Board Evaluation Board)

End-Device (Demonstration Board Evaluation Board)
Build/Clean Application

Executable file generation:
- a51 (HEX file: text file)
- s51 file (binary file with debug information)
Download Application (I)
Download Application (II)

Chipcom Flash Programmer
- Device select
- Read/Write IEEE Address
- Download application on the Flash
Example: “GenericApp”

• The GenericApp example provides a simple interface:
  – the ZigBee node finds an appropriate peer upon one button being pressed
  – It binds to that peer upon another button being pressed
  – It sends a packet containing “Hello World” every 5 seconds. Upon receiving a packet, it would display the contents on the LCD in the evaluation.
1. Build and Download GenericApp

- **Node1**
  - Uses EndDeviceEB configuration
  - Receive packets

- **Node2**
  - Uses CoordinatorDB configuration
  - Send packets
2. Start Node2 (Coordinator)
   • Red led light on
   • Network formation

3. Start Node1 (End Device)
   • Led orange light on
   • Join Network

4. SW4 Key Press on the Node1
   • Auto Find: start match description request
   • Led green light on
GenericApp (III)

5. Node2 sends “Hello world” packets to the Node1 every 5 seconds
   - View results on the LCD
   - View results on the Z-Tool provided by Texas Instrument
     • By using serial port
Example: “SimpleApp”

The SimpleApp example allows:
- the ZigBee node finds an appropriate peer upon one button being pressed
- It binds to that peer upon another button being pressed
- It sends a packet containing “Temperature” data. Upon receiving a packet, it would display the contents sending it to the serial interface.
SimpleApp (I)

1. Build and Download SimpleApp
   - Node1
     - Uses SimpleCollectorEB configuration
     - Receive packets containing temperature data
   - Node2
     - Uses SimpleSensorDB configuration
     - Send temperature data
SimpleApp II

2. Start Node1 (SimpleCollector) and Node2 (End Device)
   - Allow Bind mode by pressing SW1 Key on Node1
   - Turn on LED1 on Node1

3. On the Node 1 (SimpleCollector device), any received sensor data are written to the serial port