BUILDING MULTIPROTOCOL IOT NODES
WITH THREAD, BLE, AND ZIGBEE

ALIN LAZAR
FEBRUARY 23, 2017
PORTLAND, OR

NXP | SECURE CONNECTIONS FOR A SMarter WORLD

OpenIoT Summit
Summary

• Benefits of Multiprotocol Systems
• Protocol Standards
• Use Cases
• Platforms and Stacks
• Application Considerations
• Examples
Speaker: Alin Lazar

Software Engineering Manager at NXP Semiconductors
10+ years experience with low power wireless protocols
Shipped ZigBee, Thread, BLE network stacks and tools for microcontrollers
Focus on standardization and certification
Vice Chair of Thread Group Technical Committee
Benefits of Multiprotocol Systems
Benefits of Multiprotocol Systems

**Connect++**
Expanded, flexible connectivity from the same Device

**Reduce Design Costs**
One SKU, single firmware build

**Opens Path to IoT Convergence**
Applications can leverage best aspects of multiple standards, reduce lock-in
Wireless Protocol Standards of Focus

**Bluetooth LE (4.0+)**
- Connect to smartphones, PCs
- Accessories, Wearables, Beacons

**Zigbee**
- Low power mesh protocol
- Connect to smart home hubs
- 100s of smart home & lighting certified products

**Thread**
- IPv6 network layer scalable to low power IoT
- Mesh network without Single Point of Failure
- Border Routers: IP network gateways for mobile and cloud
Use Cases
Use Cases for Multiprotocol Edge Nodes

- **Direct Communication (BLE)**
- **Mesh Network with 10s-100s of nodes**
- **Border Routers / Hubs Gateways to Cloud and LAN (Wi-Fi)**

- **Smartphone to Device**
- **Device to Device**
- **Device to Cloud**
Multiprotocol Platforms and Stacks
Integrated Microcontrollers with Multimode Radios

- Available from several MCU vendors
- Commonly based on ARM® Cortex®-Mx
- Most common multimode Transceiver configuration: BLE and IEEE 802.15.4
- Various on-chip memory sizes
- Optimized for deep sleep low power
- Integrated security/TRNG acceleration
# Multi-Protocol Stacks for IoT Edge Nodes

<table>
<thead>
<tr>
<th>Device Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAP</td>
</tr>
<tr>
<td>BLE Host</td>
</tr>
<tr>
<td>BLE Controller</td>
</tr>
<tr>
<td>Multimode Transceiver</td>
</tr>
</tbody>
</table>
Application Considerations
Multiprotocol MCU Application Considerations

Firmware System and RTOS

Manage Radio Concurrent Operation and Co-Existence

Sleep modes and wake-up patterns

OTA Updates

Application layer protocol and ecosystem

Security
Concurrent Radio Protocol Operation API

typedef enum
{
    gMWS_BLE_c,
    gMWS_802_15_4_c,
    gMWS_ANT_c,
    gMWS_GENFSK_c,
    gMWS_None_c
} mwsProtocols_t;

mwsStatus_t MWS_Register (mwsProtocols_t protocol, pfMwsCallback cb);
mwsStatus_t MWS_Acquire (mwsProtocols_t protocol, uint8_t force);
mwsStatus_t MWS_Release (mwsProtocols_t protocol);
mwsStatus_t MWS_SignalIdle (mwsProtocols_t protocol);
mwsStatus_t MWS_Abort (void);

uint32_t MWS_GetInactivityDuration (mwsProtocols_t currentProtocol);
mwsProtocols_t MWS_GetActiveProtocol (void);
Radio Co-Existence with MWS API

```c
void MWS_CoexistenceEnable (void);
void MWS_CoexistenceDisable (void);

mwsStatus_t MWS_CoexistenceInit(void *rfDenyPin, void *rfActivePin, void *rfStatusPin);
mwsStatus_t MWS_CoexistenceRegister (mwsProtocols_t protocol, pfMwsCallback cb);
void MWS_CoexistenceSetPriority(mwsRfSeqPriority_t rxPrio, mwsRfSeqPriority_t txPrio);

mwsStatus_t MWS_CoexistenceRequestAccess(mwsRfState_t newState);
mwsStatus_t MWS_CoexistenceChangeAccess(mwsRfState_t newState);
uint8_t MWS_CoexistenceDenyState(void);
void MWS_CoexistenceReleaseAccess(void);

typedef uint32_t(*pfMwsCallback) (mwsEvents_t event);
typedef enum
{
    gMWS_Init_c,
    gMWS_Idle_c,
    gMWS_Active_c,
    gMWS_Release_c,
    gMWS_Abort_c,
    gMWS_GetInactivityDuration_c
}mwsEvents_t;
```
Multiprotocol IoT Gateways
Gateways, Hubs, Border Routers

Gradual Transition from Application Layer Gateway to Network Layer Gateway
What’s Next for Multiprotocol IoT
What’s Coming Next for Multiprotocol IoT Systems

Even more standards / protocols integrations at the edge:
  Wi-Fi
  LPWAN

Commercial / Professional use cases

Even more flexible radios

Mesh networks everywhere!

IPv6 (and end-to-end) everywhere!
IP as Network Convergence Layer (Projection)

Device Application

Libraries for Application and Ecosystem Protocols for IP Networks
- OCF
- dotdot
- LWM2M
- Weave
- ...

IPv6 Sockets with UDP, DTLS

BLE Interface
- 802.15.4/Thread Interface
- Wi-Fi Interface

Multimode Transceiver
Your Next Steps

Get some Multiprotocol IoT platforms:
- NXP KW41Z
- FRDM-KW41Z
- USB-KW41Z
- Modular Gateway Reference Design

Get platform drivers, firmware SDKs, Linux Host SDKs:
- NXP MCUXpresso Config Tools
- KW41Z SDK Software and Design Tools

Join and contribute to the standard groups:
- Influence standard spec definitions
- Achieve quicker, certified interoperability
- Public events: ZigBee Winter Summit – Monday March 6, Austin, TX
  Thread Technology Workshop – Monday March 27, Mountain View, CA

Contribute to OSS (most OSS support is WIP and needs your help):
- Zephyr, Mynewt, NimBLE, IoTivity, OpenThread
Hands-on Examples
Example 1: Thread Network Shell with Kinetis BLE Toolbox App
Example 2: Host SDK

Using Python Bindings (multimode.py) for Linux Host scenario

Available as part of KW41Z Connectivity Software Package

KW Host API Wrappers – Python, Java
- FSCI Packets → Objects
- Send Commands
- User Defined Actions Upon Events

KW Host API – Set of C Libraries
- Device Detection
- Serial Communication
- Raw FSCI Packets
- FSCI over RNDIS

Operating System
- Windows
- Linux
- OS X

Kinetis-W MCUs
- FRDM-KW24D
- TWR-KW24D512
- USB-KW24D512
- FRDM-K64F & FRDM-MCR20A
- FRDM-KW41Z
- USB-KW41Z
Looking forward to your Questions

alin.lazar at nxp.com
https://community.nxp.com/community/wireless-connectivity