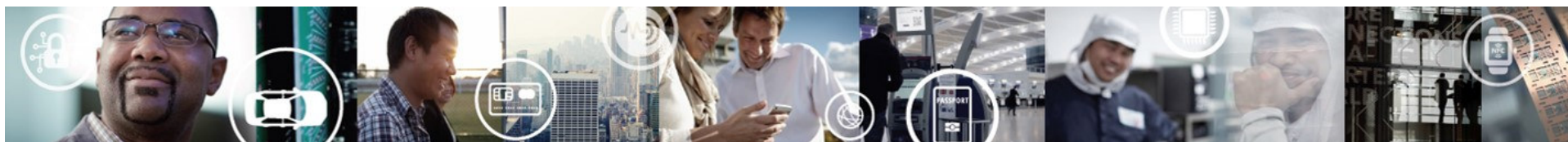


THREAD WORKSHOP

LYON & STRASBOURG
TECHNOLOGY DAYS

PHILIPPE MANGAUD
MAY 2016



EXTERNAL USE



SECURE CONNECTIONS
FOR A SMARTER WORLD

THREAD NETWORKING ARCHITECTURE



PHREAD The need for a new wireless network

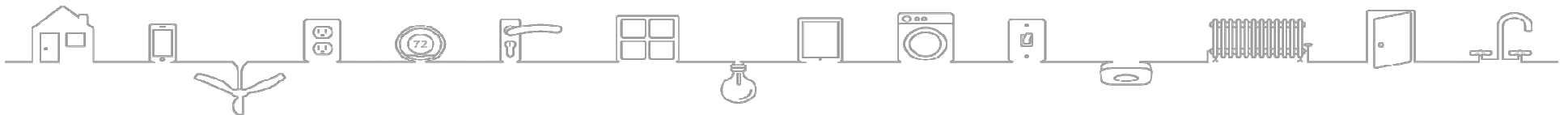
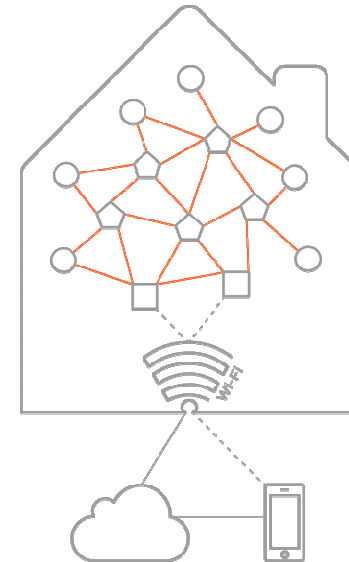


Target Applications

Thread is designed for all sorts of products in the home

- Appliances
- Access control
- Climate control
- Energy management
- Lighting
- Safety
- Security

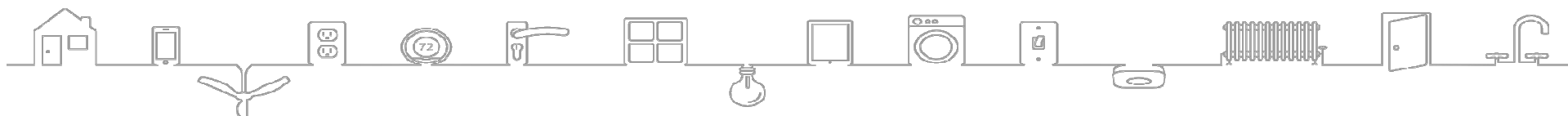
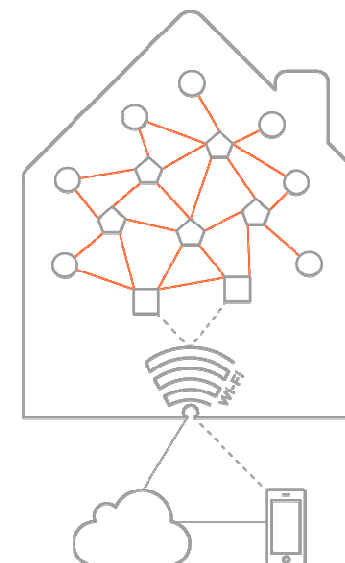
Devices working together to form a cohesive mesh network



Business Benefits

Membership to Thread comes with its benefits:

- Access to the technology
- Immediate product planning and development
- Use Thread Certification Program and test suite
- Participation in Marketing and PR campaigns
- Get involved in one of the working committees – Marketing, Use Cases, Certification
- Network with an ecosystem of companies building connected products for the home
- Help promote Thread and Thread-enabled products



What is Thread?

A secure wireless mesh network for your home and its connected products

Built on well-proven, existing technologies

- Runs on existing 802.15.4 silicon
- Uses 6LoWPAN with IPv6 addressing
- UDP Transport

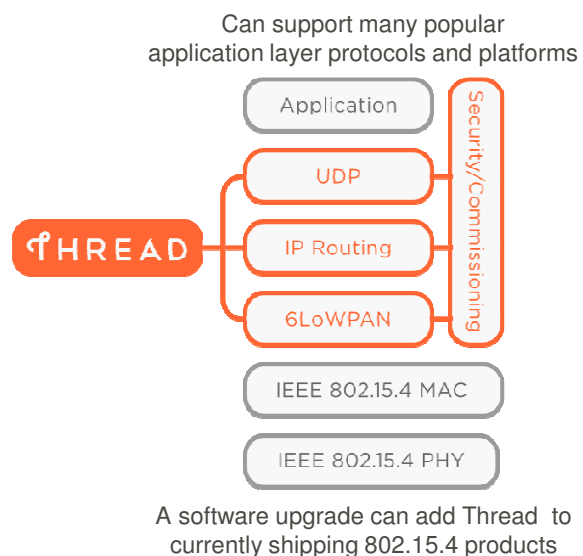
New mandatory security architecture

Simple and secure to add / remove products

Scalable to 250+ products per network

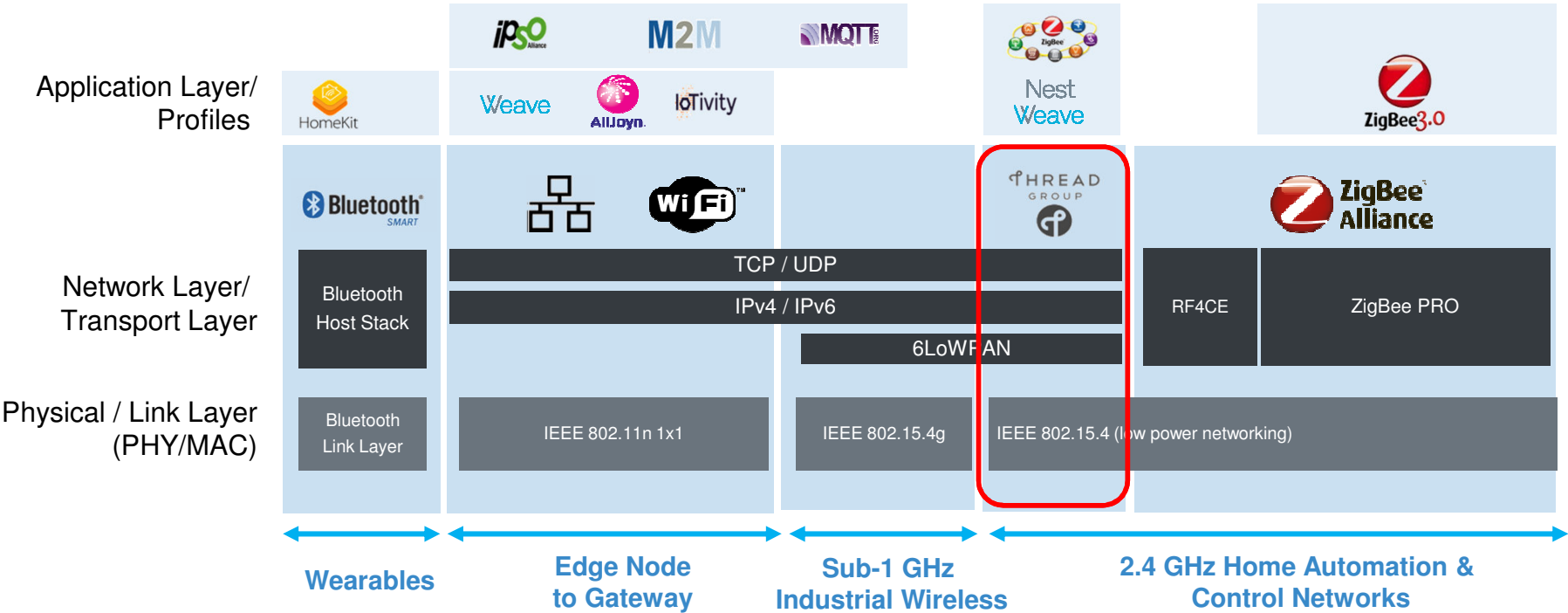
Designed for very low power operation

Reliable for critical infrastructure



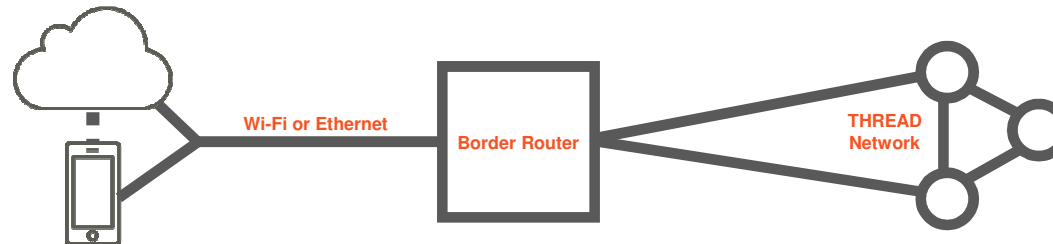
Thread Specification is available to Thread Group members

IoT Connectivity Landscape – Where does Thread play?



Promise of IoT requires IP all the way to the end node

- Cloud Services can address devices from the Internet
- Home Network can directly address devices through Border Routers
- Devices can address local devices on HAN or off network devices using normal IP addressing



Cloud Connectivity

For control when not at home
When within the home, phone or tablet
must go direct to gateway to eliminate
latency of going to the cloud
Has to be seamless to consumer

Border Router

Bridge from the Thread Network
to Wi-Fi/Ethernet
Forwards data to cloud
Provides Wi-Fi connectivity to
phone, tablet or other devices in
the home network.

Device Communication

Device to device communication
within the Thread network for
operations in the home

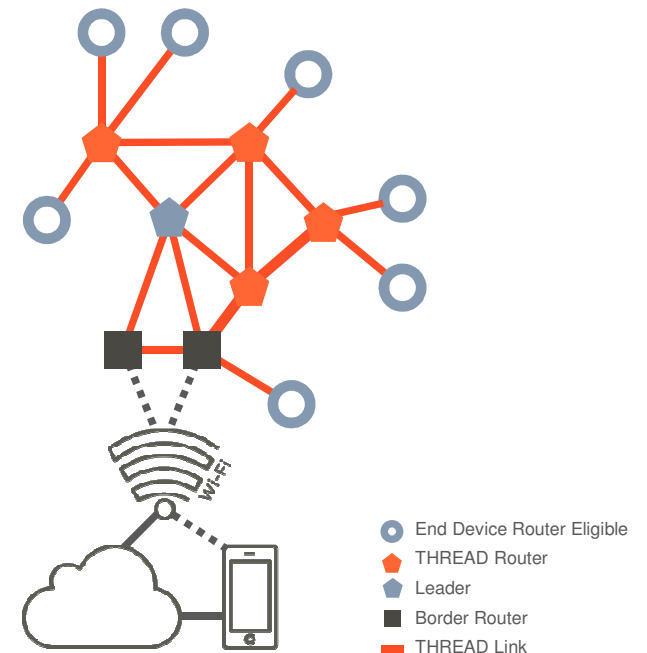
Direct Addressability of Devices

All devices have IPv6 addresses plus 16-bit short address on HAN

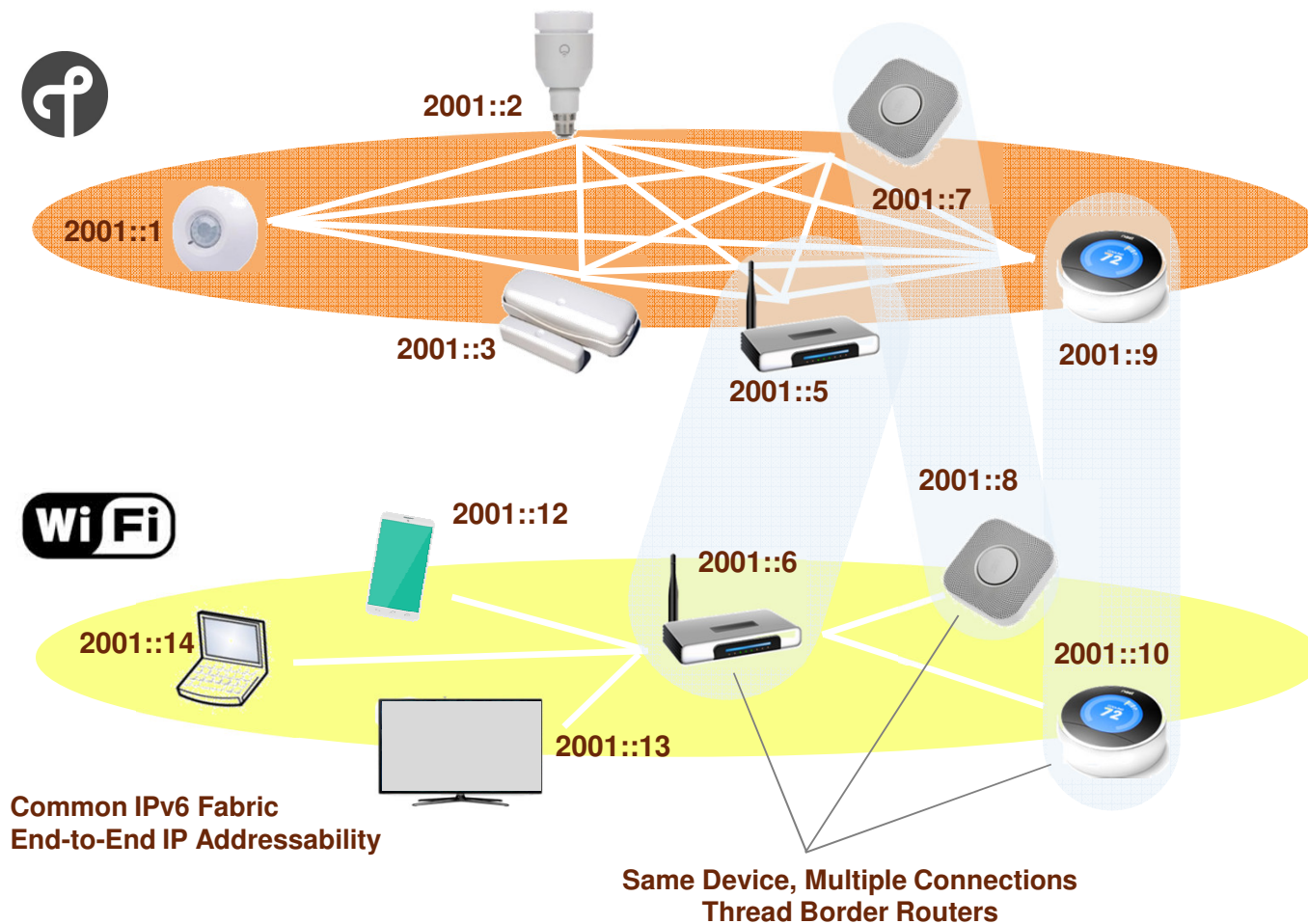
Home Network can directly address devices through Border Routers

Cloud Services can address devices from the Internet

Devices can address local devices on HAN or off network devices using normal IP addressing



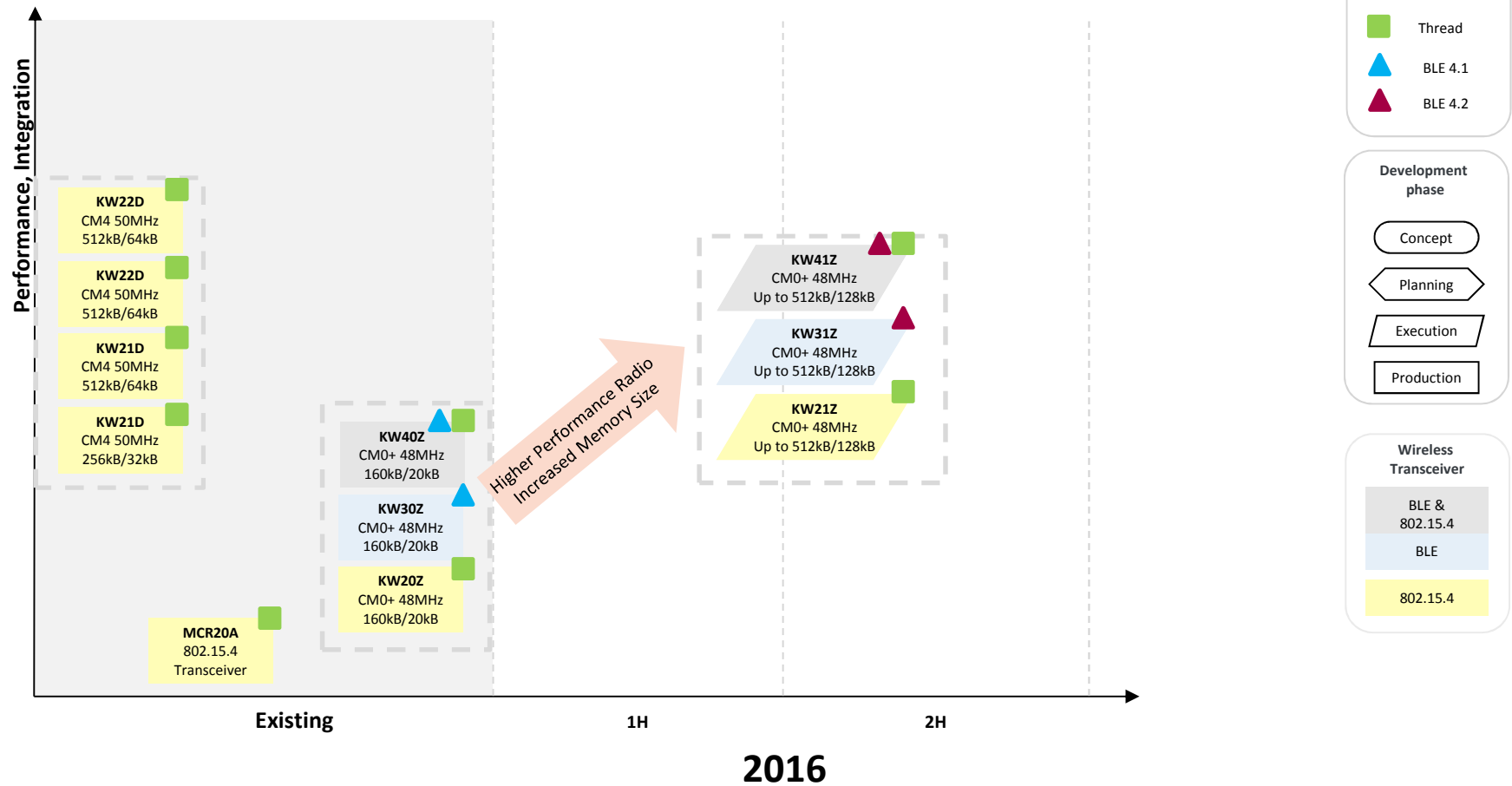
Thread and Wi-Fi: Common IP Infrastructure



PRODUCTS



Kinetis 2.4GHz Wireless Connectivity Portfolio



KW21D/22D/24D/ block diagram

Cortex M4, 256/512/512 kB Flash - 32/64/64 kB RAM

• CPU

- 50 MHz ARM Cortex-M4 core, Up to 512 kB Flash & up to 64 kB RAM

• 2.4 GHz radio transceiver

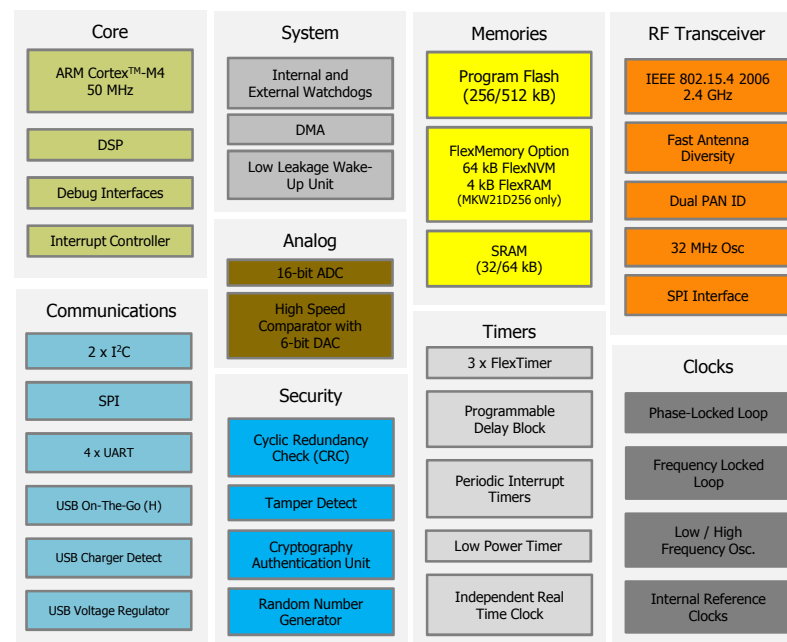
- IEEE-802.15.4 2011 compliant
- Dual personal area network (PAN) support in hardware
 - Run two RF networks simultaneously
- Antenna diversity with automatic antenna selection
- +8 dBm output power
- -102 dBm RX sensitivity
- Peak typical current:
 - 17mA TX @ +0dBm, 19mA RX

• Security

- Active and passive tamper detection with RTC timestamp
- Crypto engine: DES, 3DES, AES 128-256, SHA-1, SHA-256, MD5, RNG

• System

- Operating range: 1.8 V to 3.6 V
- Ambient temperature: -40 °C to +105 °C
- LGA 8x8 mm



KW41Z/31Z/21Z

• CPU

- 48 MHz ARM Cortex-M0+ core, Up to 512 kB Flash & 128 kB RAM

• 2.4 GHz radio transceiver

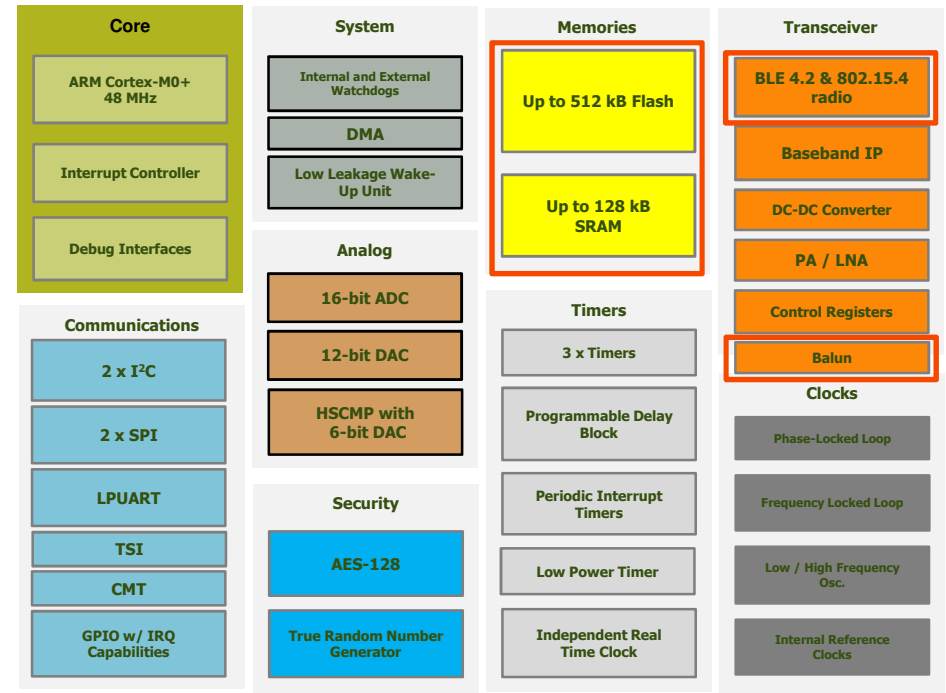
- [IEEE-802.15.4 2011 compliant](#)
- Dual PAN & Antenna diversity support
- [Bluetooth Smart 4.2 compliant](#)
- Programmable output power : -30 to +3.5 dBm
- -101 dBm RX sensitivity (IEEE 802.15.4)
- -95 dBm RX sensitivity (Bluetooth Smart)
- Peak typical current: 6.5mA TX @+0dBm and 6.5mA RX with DC/DC activated
- IEEE 802.15.4 & Bluetooth Smart [concurrent](#) mode supported
- Integrated [balun](#) (~9% board area saving)

• Security

- [Crypto](#) engine: AES-128, TRNG

• System

- Buck Boost DC/DC working from 0.9V to 4.2V
- Ambient temperature: -40 °C to +105 °C
- QFN 7x7mm, WLCSP



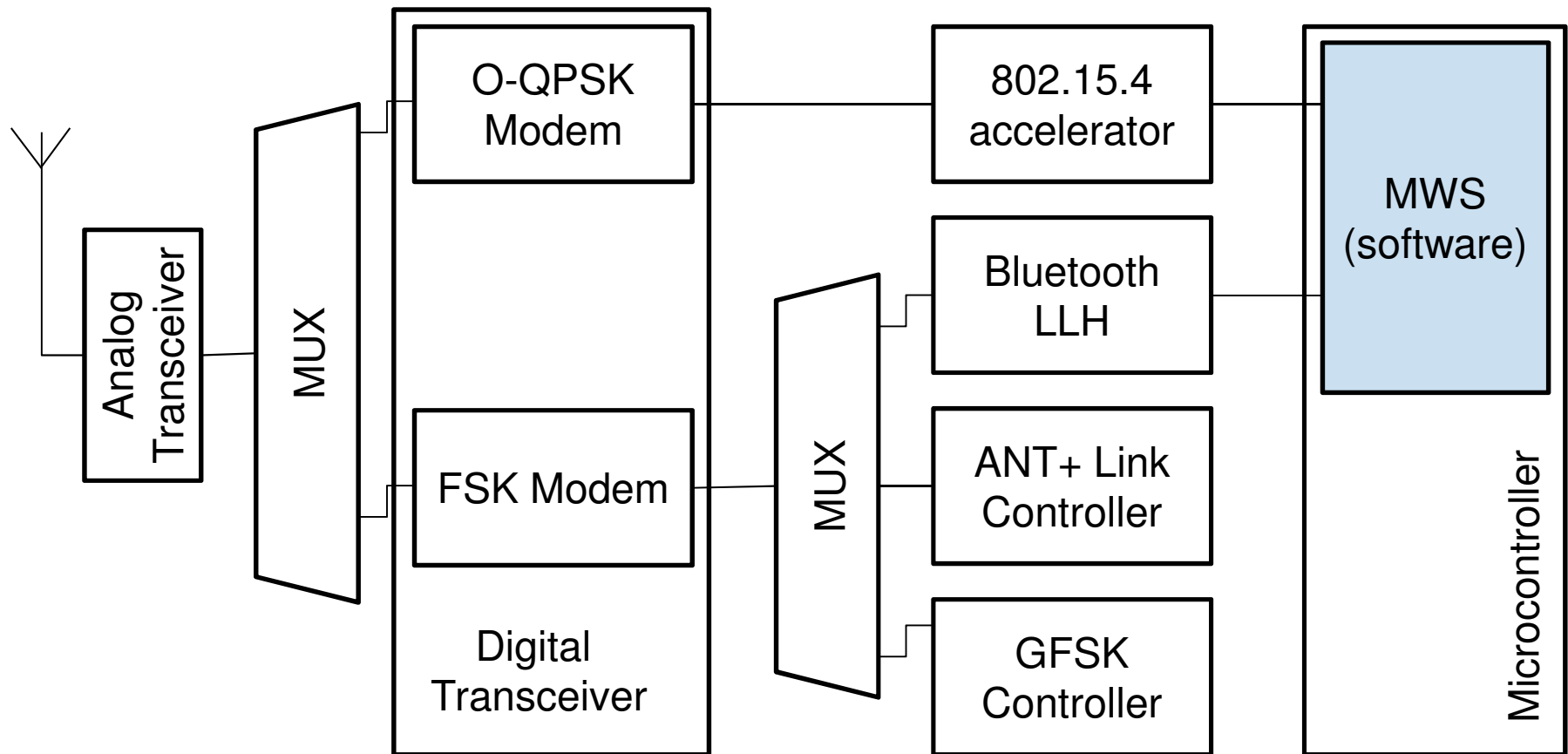
Differences from KW40Z/30Z/20Z



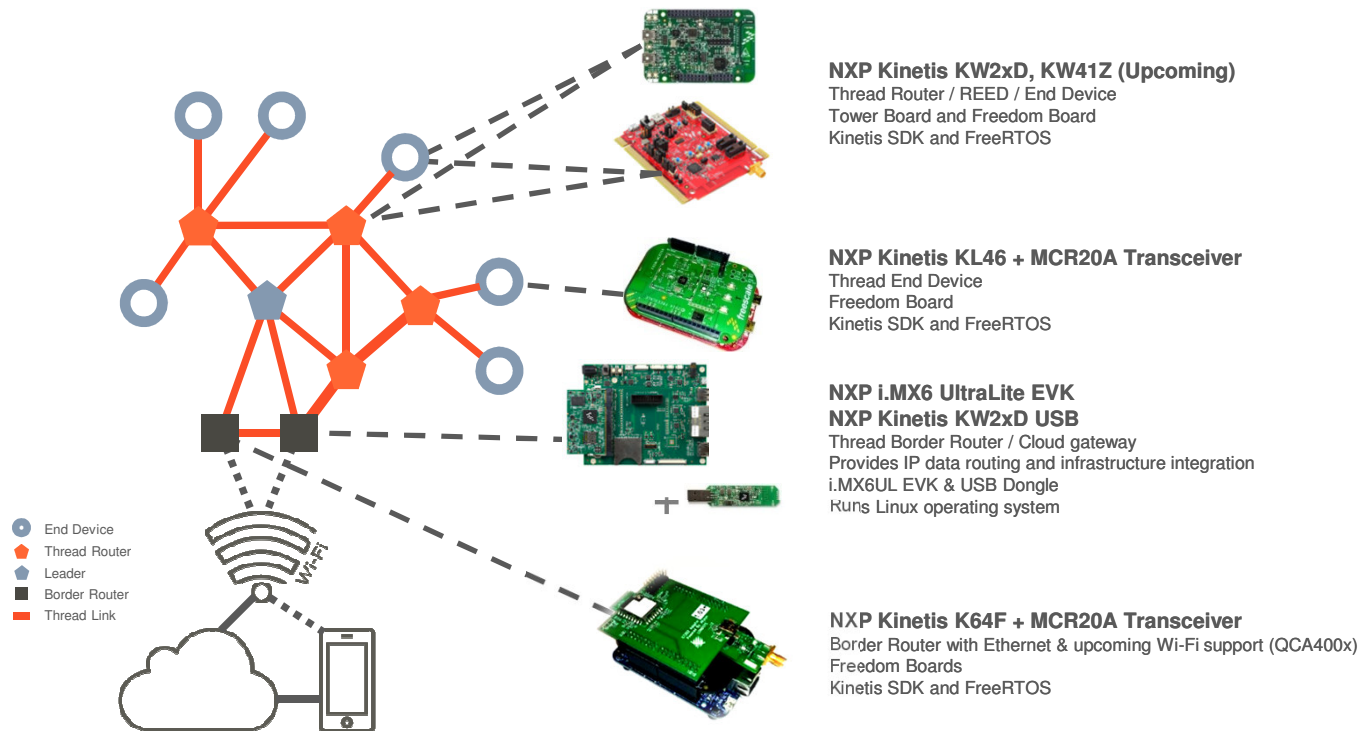
Differences between KWx0z (current) and KWx1z (next gen)

	KW40/30/20Z	KW41/31/21Z
MCU	Cortex-M0+ w/ 160KB Flash, 20KB SRAM	Cortex-M0+ w/ 512KB Flash, 128KB SRAM
Protocols	802.15.4 / BLE 4.1	802.15.4 / BLE 4.2 / Generic FSK
Incremental BT-LE features	BLE v4.1 with 1 connection	BLE v4.2 with 2 simultaneous connections
BLE sensitivity	-91dBm	-95dBm with integrated balun
RF pins	Differential input/output with external balun	Single ended with integrated balun (BOM cost reduction)
TX Max Output Power	+5dBm	+3.5dBm with integrated balun
TX output power steps	Non-linear gain steps (-18dBm to +5 dBm)	Finer gain steps over a larger control range (-25 dBm to +4dBm)
Power consumption	6.5 mA Rx 8.4 mA Tx	6.5 mA Rx 6.5 mA Tx
Package	LQFN	LQFN, WLCSP (PYW)

Kinetis KW41Z System Simplified Block Diagram



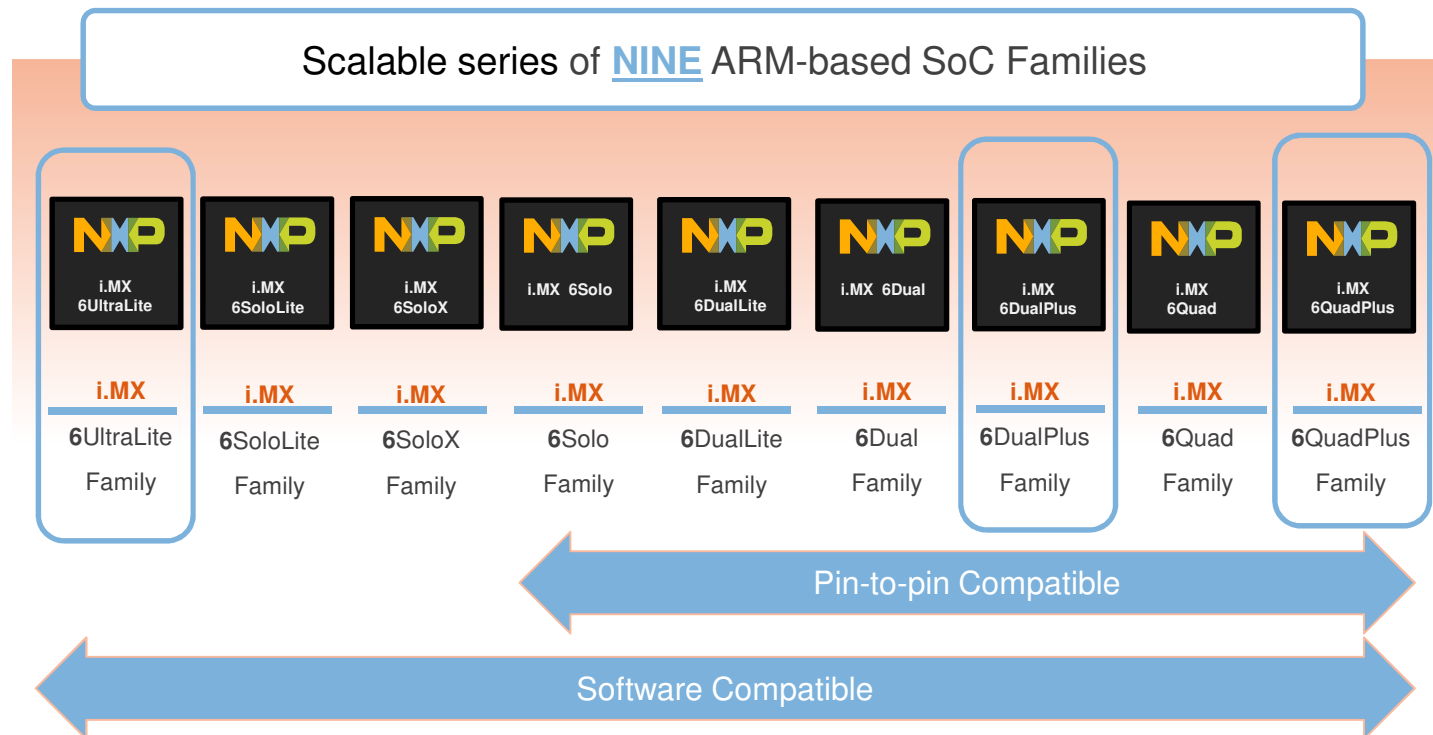
NXP's Kinetis & i.MX Thread Hardware Offering



The most **complete** Thread end to end platform available!

i.MX 6 Series: Supreme Scalability and Flexibility

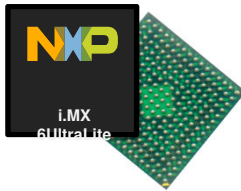
Leverage One Design Into Diverse Product Portfolio



i.MX 6UltraLite Advantages

Lowest cost and smallest i.MX 6 member

- ARM Cortex- A7 @ 528 MHz
 - The 14x14 289 MAPBGA with 0.8mm pitch for simple and low cost PCB design.
 - The 9x9 289 MAPBGA with 0.5mm pitch for space constrained applications.



Most Power efficient Applications Processor

- Integrated power management module that reduces the complexity of external power supply and simplifies power sequencing.



"It provides up to 20% more single thread performance than the Cortex-A5 and provides similar performance to mainstream Cortex-A9 based smartphones in 2012 while consuming less power."

www.arm.com/products/processors/cortex-a/cortex-a7.php

Connectivity optimized for Industrial and IoT applications

- 2x high-speed USB on-the-go with PHY
- Multiple expansion card ports (high-speed)
- 2x 12-bit ADC modules (up to 10 input channels)
- 2x smart card interfaces compatible with EMV Standard v4.3 and a variety of other popular interfaces
- 2x CAN ports



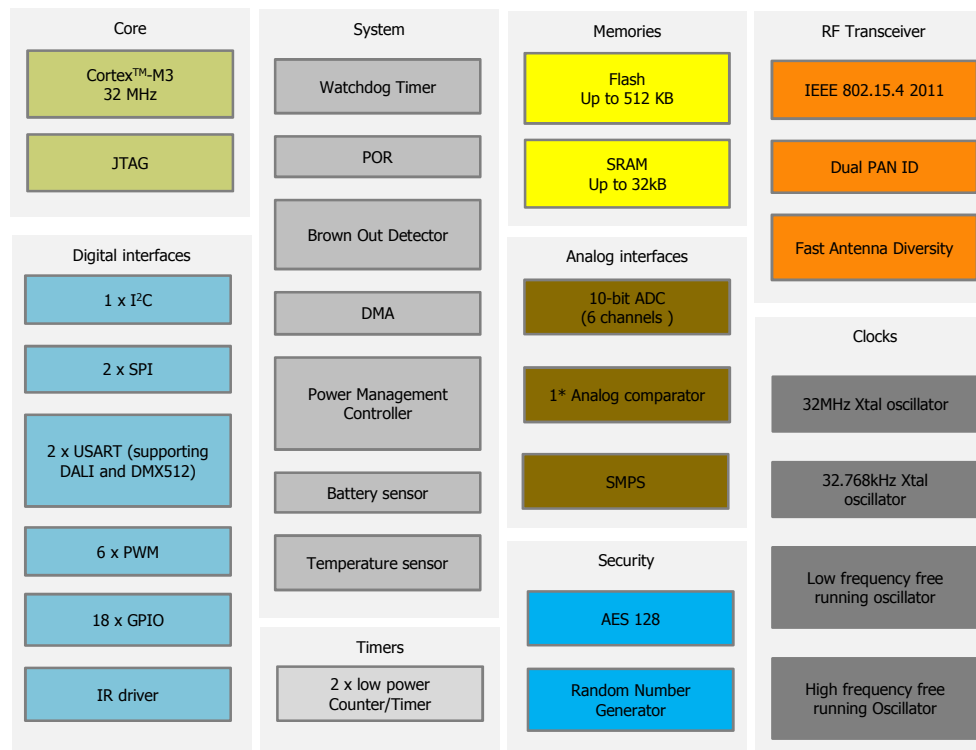
Advanced Security

- Hardware-enabled security features that enable secure e-commerce, digital rights management (DRM), information encryption, On-The-Fly DRAM encryption, secure boot and secure software downloads

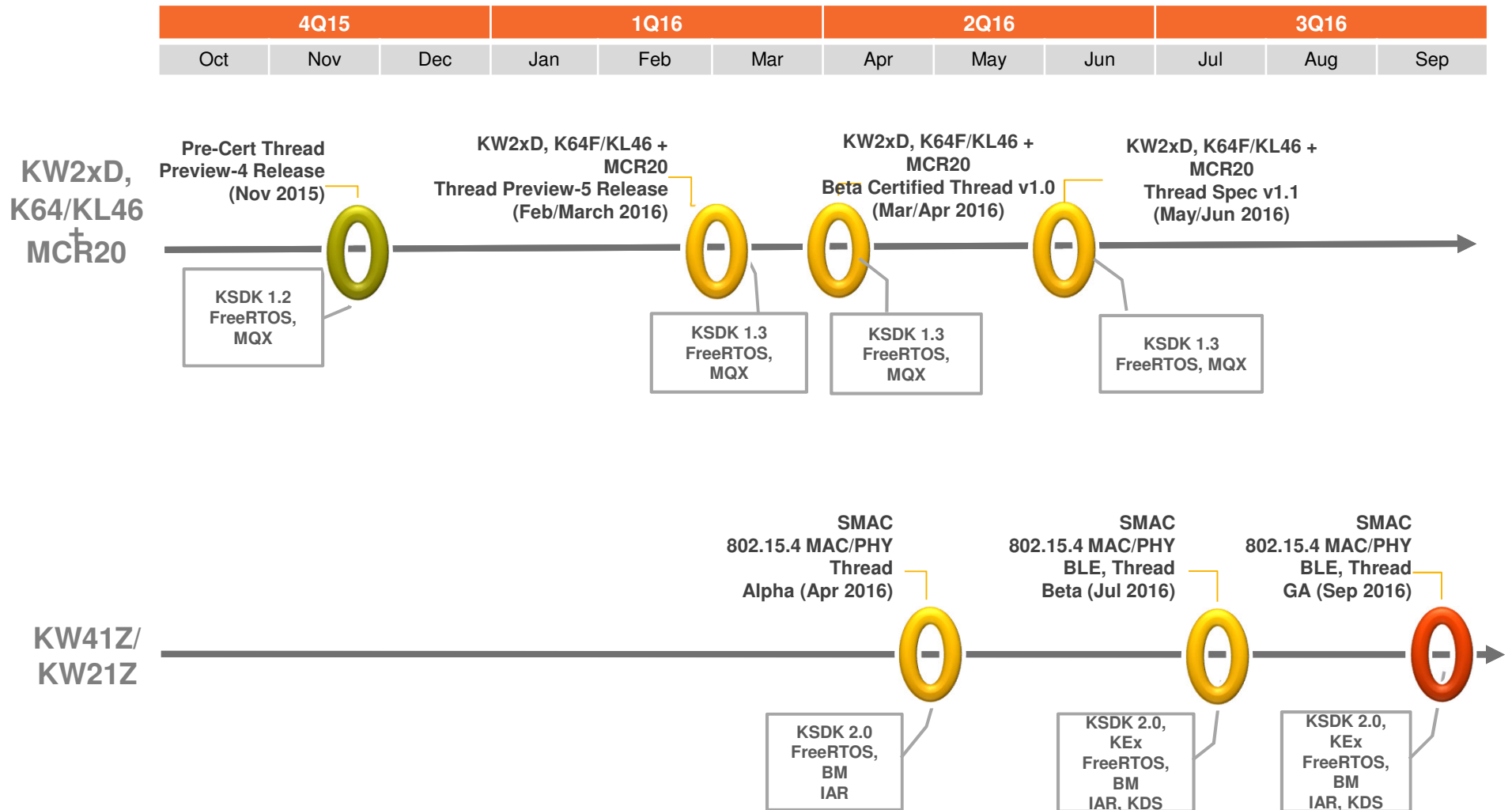


JN517x: Wireless MCU

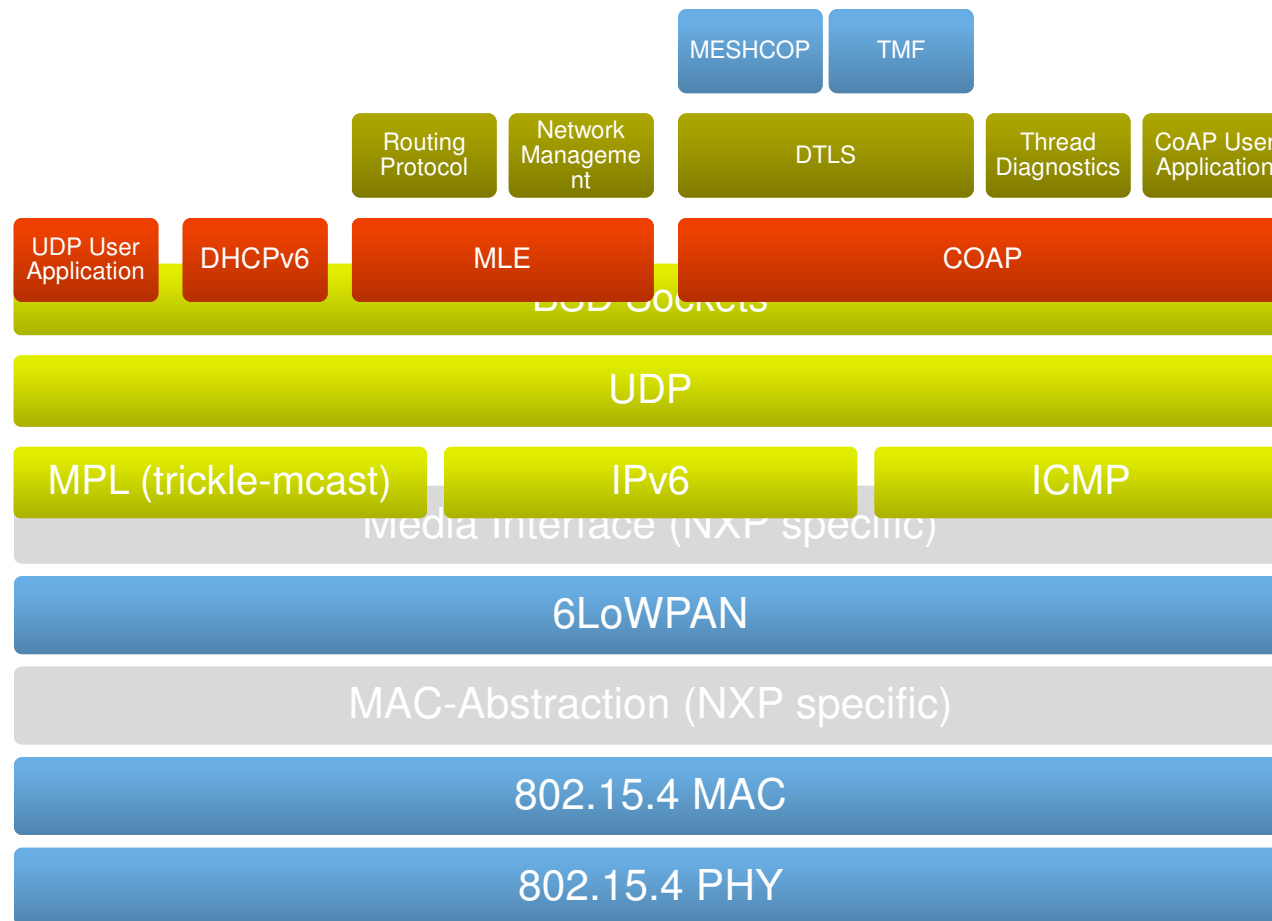
- **CPU**
 - 32 MHz ARM Cortex-M3 core
 - Up to 512 KB Flash & up to 32 KB RAM
- **2.4 GHz radio transceiver**
 - IEEE-802.15.4 2011 compliant
 - Dual PAN support
 - Antenna diversity
 - +10 dBm power amplifier
 - -96 dBm RX sensitivity
 - Peak typical current:
 - 22.5mA TX @ +10dBm, 14mA @ +3dBm
 - 14.8mA RX
- **Security**
 - Crypto engine: AES 128-256, RNG
- **System**
 - Ambient temperature: -40 °C to +125 °C
 - HVQFN40 6x6 mm



Thread Software Timeline



Thread High Level Block Diagram



HANDS ON TIME



Hands-on Lab Inventory

#	Item	Quantity
1	FRDM-KW24D512 Development Board	2
2	Mini-USB Cables	2

Board Setup

1. Plugin both FRDM-KW24D512 boards into the PC, using the mini-USB cables provided. Be sure to use the USB port marked as “Debug”



Board Setup

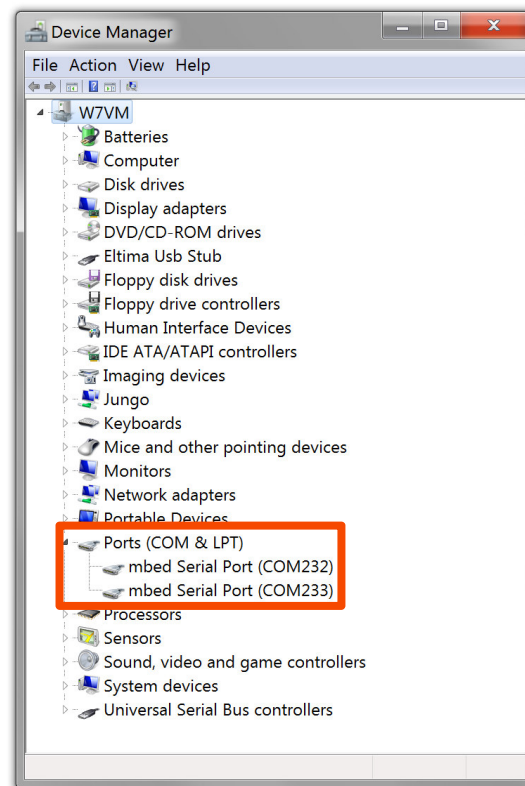
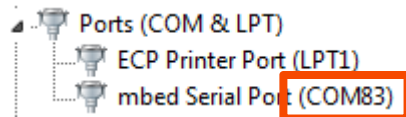
Note the **R1** and **R2** (**R**outer Eligible End Devices) shorthand when referring to the 2 devices:



COM Ports

With both boards plugged in please open device manage to inspect the COM ports assigned:

In Device Manager, under **Ports (COM & LPT)**, note corresponding **COMxx** port number assigned for each connection

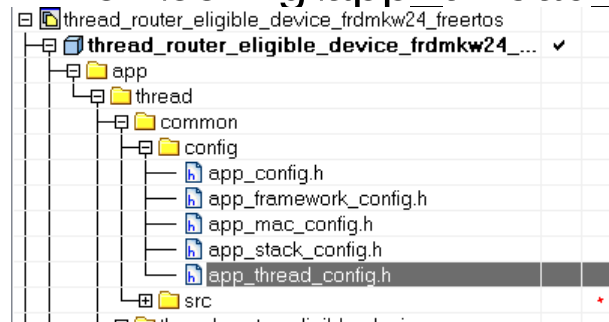


Write down a device to COM port assignment list:

Device	Port
R1	COMxx
R2	COMxx

Those settings have been changed in the source code

- Found in app\thread\common\config\app_thread_config.h



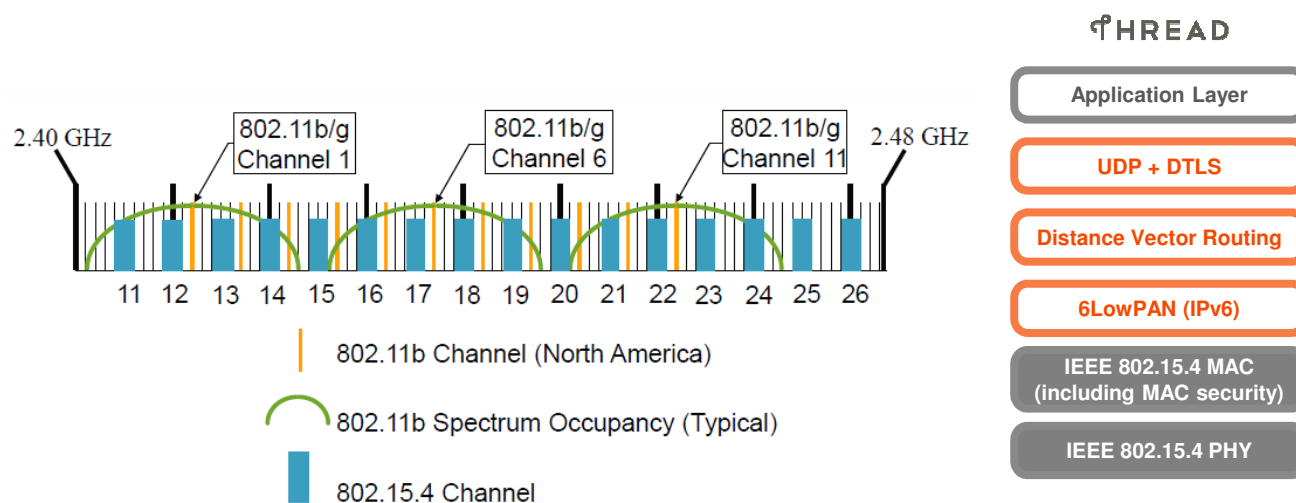
- change channel mask and / or Network Name to avoid overlaps:

```
#define THR_SCANCHANNEL_MASK    (0x00000001 << CH)
#define THR_NETWORK_NAME        {14, "Kinetis_Thread"}
```

- Example:
- CH = 11 for channel 11
- MYNAME = make sure you put the correct length for the string

IEEE 802.15.4

IEEE 802.15.4 channel occupancy on 2.4GHz

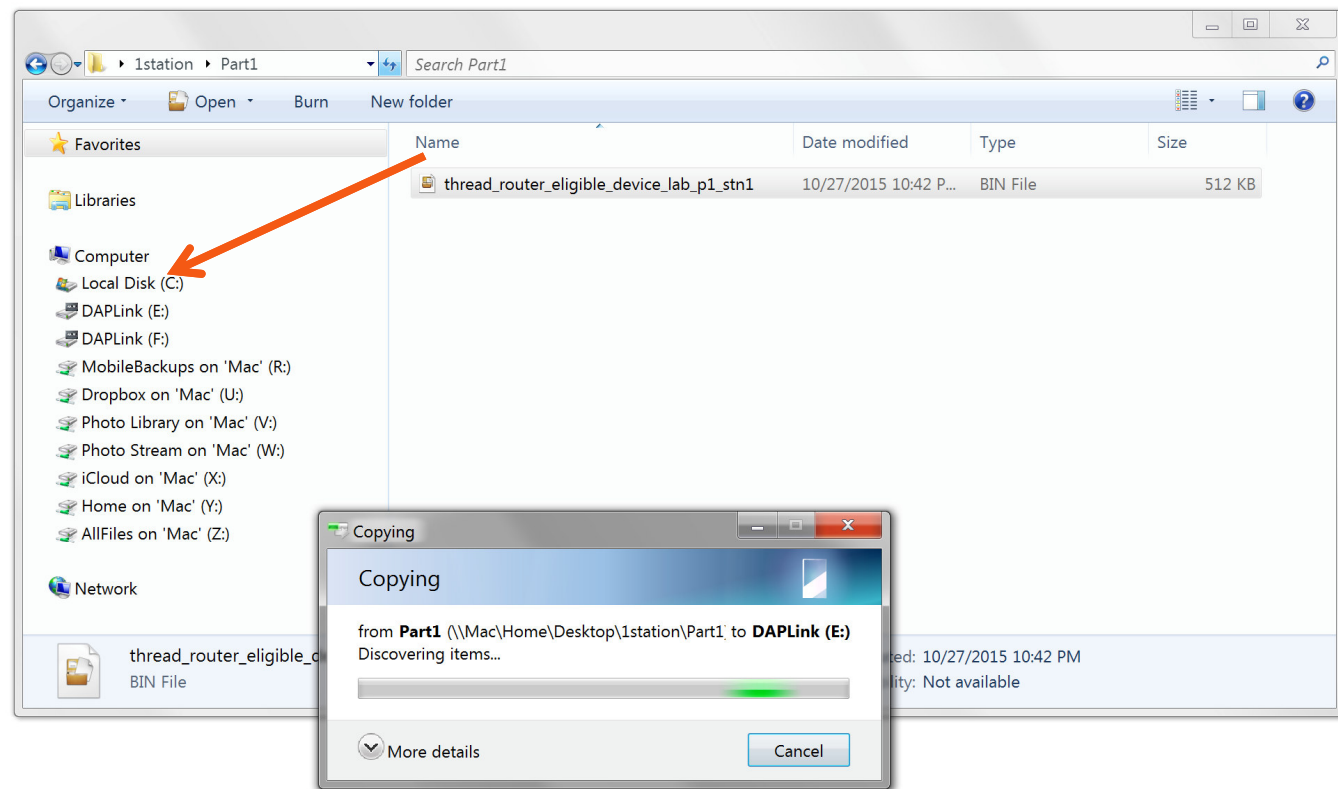


802.15.4 open channels when Wi-Fi fully utilized the band

- 15, 20, 25, 26.

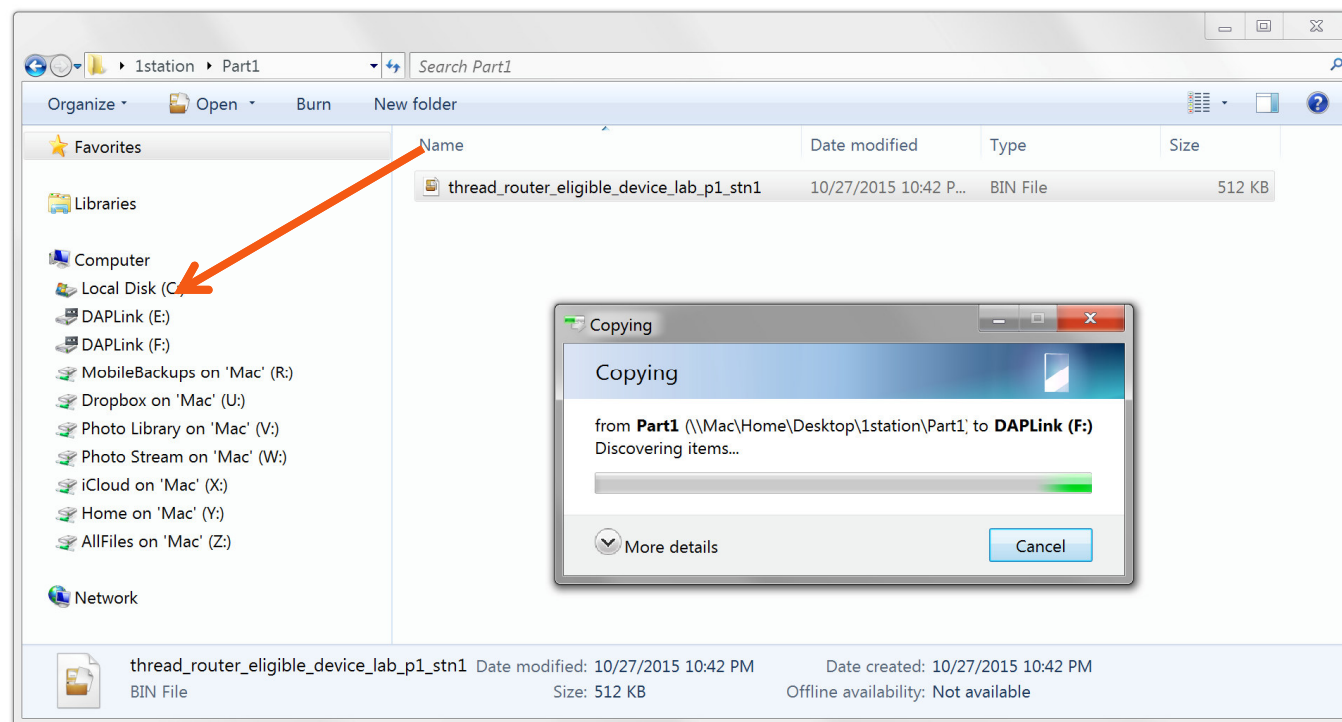
Program the First FRDM Board: R1

Drag and drop the “thread_router_eligible_device_channelx.bin” on Desktop to the “DAPLink (E):” drive. Note: you may have a different drive letter.



Now Program both the Second FRDM Board: R2

Now repeat the process for the second board. Drag and drop the “thread_router_eligible_device_channelx.bin” on Desktop to the “DAPLink (F:)” drive. Note: you may have a different drive letters.



Board Setup

After the second binary completes transfer. Verify the the RGB LED and LED are blinking blue on both boards, indicating idle state.

If LEDs are not blinking, press Reset SW to initialize the board. You might also have to un-plug/plug the boards on some setups.



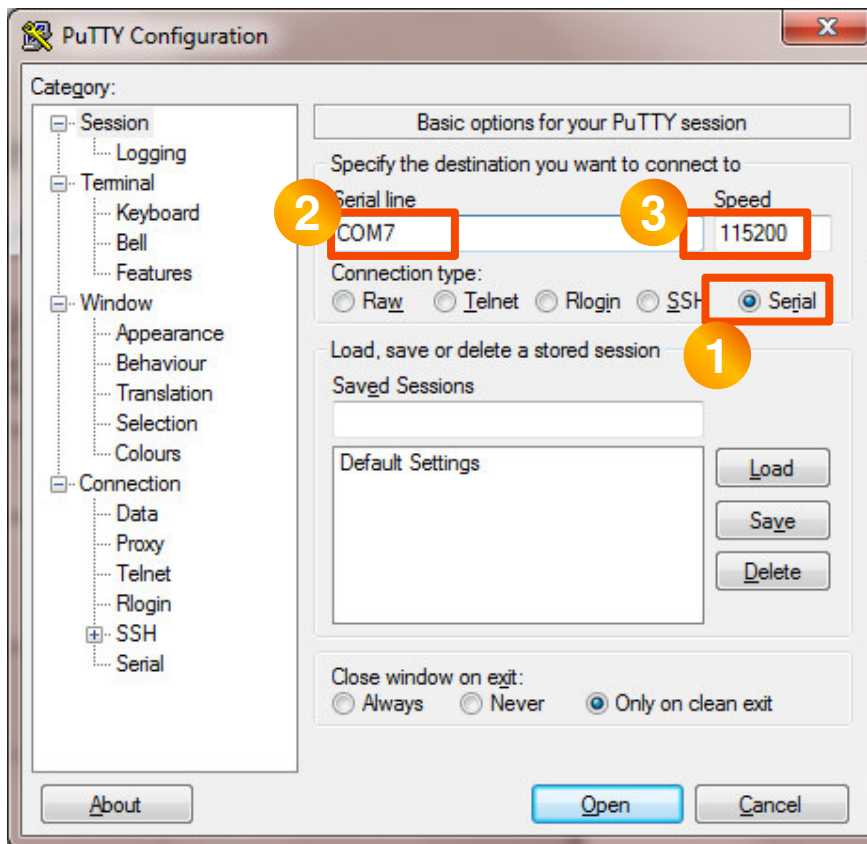
Reset



Reset

Shell Terminal

For R1 board: Launch a shell terminal (e.g.: putty) on serial port COMxx of the board. Speed:115200 bps

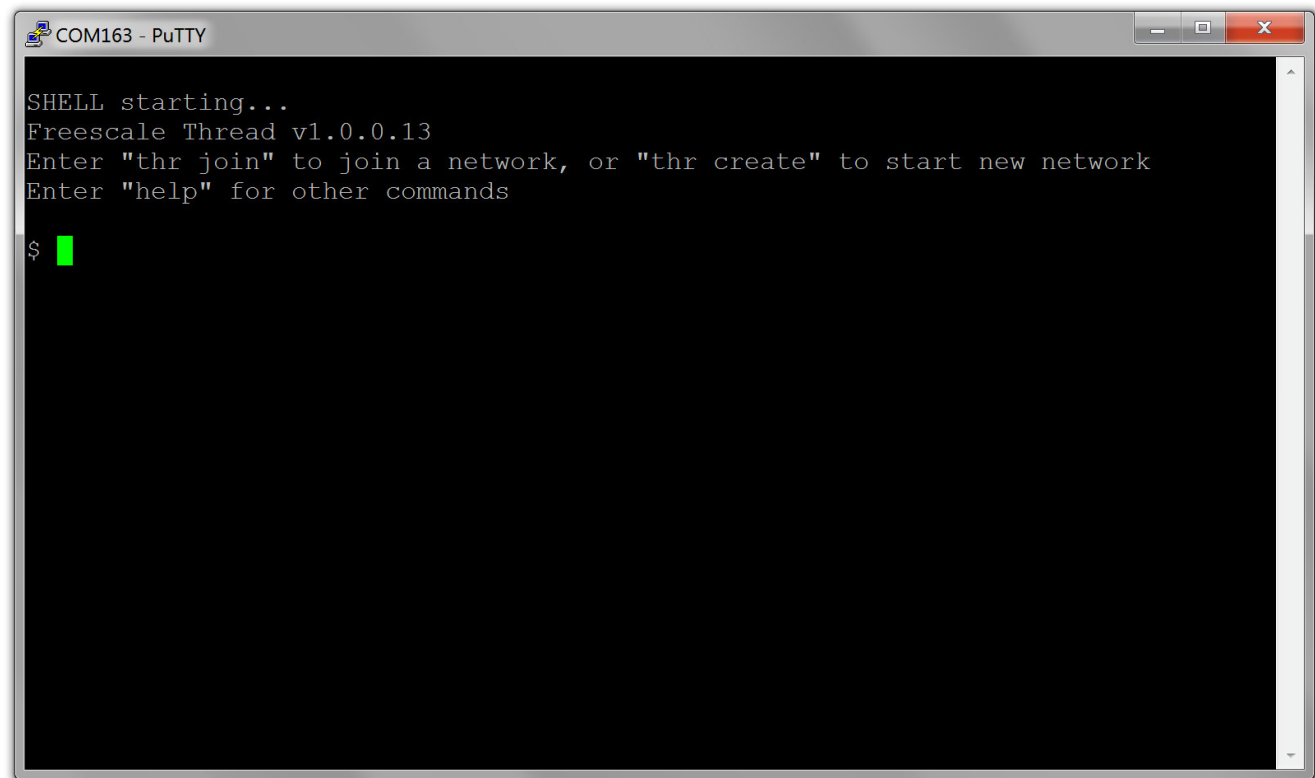


Shell Terminal

On R1: press reset switch again. Note the shell initialization messages:



Reset



```
COM163 - PuTTY

SHELL starting...
Freescale Thread v1.0.0.13
Enter "thr join" to join a network, or "thr create" to start new network
Enter "help" for other commands

$
```

Kinetis THREAD Connectivity Shell

- The Shell module is offering a command line user interface over a serial interface, usually UART.
- Implements a basic set of commands (not as many as FSCI does).
- The commands can be viewed by typing the 'help' command in the host serial terminal application.
- The Shell can be used for several purposes, including configuring the network, testing the network connectivity, MAC filtering, sockets usage and so on.

\$ help	
help	print command description/usage
version	print version of all the regd modules
history	print history
thr	Thread Stack commands
ifconfig	IP Stack interfaces configuration
ping	IP Stack ping tool
coap	Send CoAP message
reboot	MCU Reset
factoryreset	FactoryReset Command
echoudp	Echo udp client
identify	Identify Device

```
$ help thr
thr - Commands for Thread Network
thr join
thr scan [active|energy|both]
thr get attributes - displays a list of all attributes
thr get <ATTRNAME/TABLENAME>
thr set <ATTRNAME> <value>
```

```
$ thr get attributes
Thread network attributes:
```

Establishing a new Thread network

1. On R1: type **thr create** to start a new Thread network
2. Note network initialization messages in the shell that R1 is a Thread Leader of a new network



RGB->Green

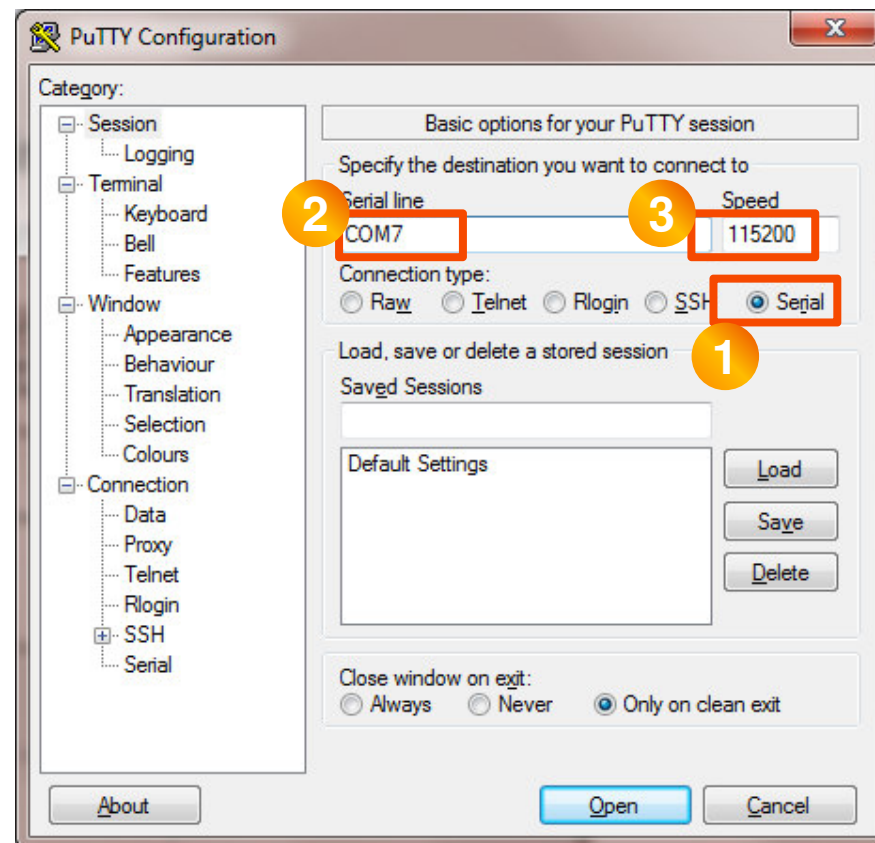
```
$ thr create
Creating network...
$
Node has taken the Leader role
Created a new Thread network on channel 11 and PAN ID:0x11

Interface 0: 6LoWPAN
  Mesh local address (ML64): fd42:ac13:a064::d1ab:4766:a905:2ba
  Mesh local address (ML16): fd42:ac13:a064::ff:fe00:0
(Local) Commissioner Started

$
```

Shell Terminal

For R2 board: Launch a shell terminal (e.g.: putty) on serial port COMxx of the board. Speed:115200 bps

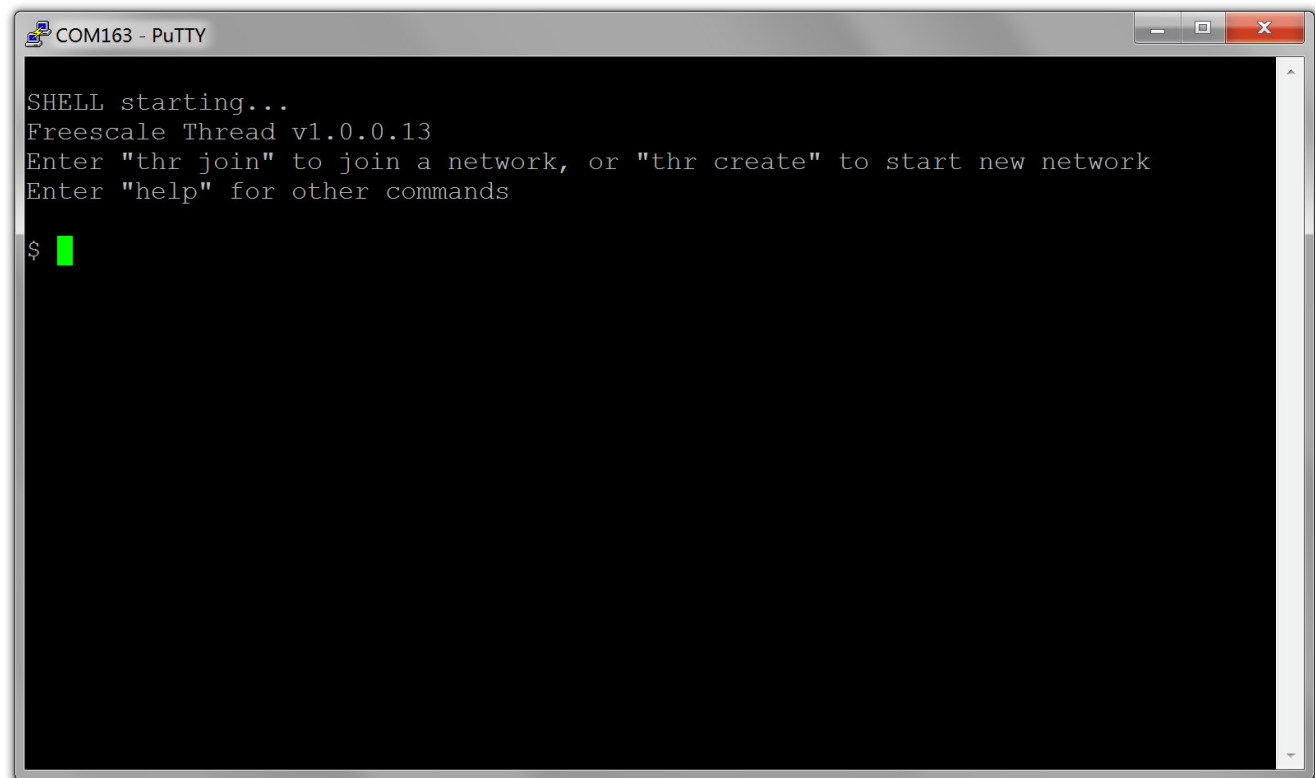


Shell Terminal

On R2: press reset switch again. Note the shell initialization messages:



Reset



```
COM163 - PuTTY

SHELL starting...
Freescale Thread v1.0.0.13
Enter "thr join" to join a network, or "thr create" to start new network
Enter "help" for other commands

$
```

Establishing a new Thread network

1. On R2: type **thr join** to start a new Thread network
2. Note network initialization messages in the shell that R1 is a Thread Leader of a new network



RGB->Off

```
$ thr join
Joining network...
$
Joining a Thread network...
Commissioning successful

Attached to network with PAN ID: 0x11
Requesting to become Active Router...
Success

Interface 0: 6LoWPAN
Mesh local address (ML64): fd06:d333:cdd3::912f:7992:466d:e45a
Mesh local address (ML16): fd06:d333:cdd3::ff:fe00:400
```

Inspecting the address configuration

In the R1 shell, type the **ifconfig** command to display its interface and IP address configuration

```
$ ifconfig
```

```
Interface 0: 6LoWPAN
```

```
Link local address (LL64): fe80::adc9:a4da:3f68:9ef4
```

```
Mesh local address (ML64): fd06:d333:cdd3::e041:1ba4:3459:cd87
```

```
Mesh local address (ML16): fd06:d333:cdd3::ff:fe00:0
```

```
Link local all Thread Nodes(MCast): ff32:40:fd06:d333:cdd3::1
```

```
Realm local all Thread Nodes(MCast): ff33:40:fd06:d333:cdd3::1
```


IPv6 Address Notation Basics

- An IPv6 address has 16 bytes (128 bits), represented as eight 2-byte groups:

2001:3344:5566:7788:99AA:BBCC:DDEE:0000

- All 0000 groups can be represented as a single 0
- Leading 0s in groups can be omitted
- Longest consecutive set of all 0 groups may be represented as 2 colon signs:

2001:0044:0000:0000:0000:BBCC:00EE:0000

is equivalent to

2001:44::BBCC:EE:0

- Addresses are usually divided as **Network Prefix** and **Network Address**:

2001:3344:5566:7788:99AA:BBCC:DDEE:FF00

- Representing the length of the network prefix can be done with a /prefixlen notation:

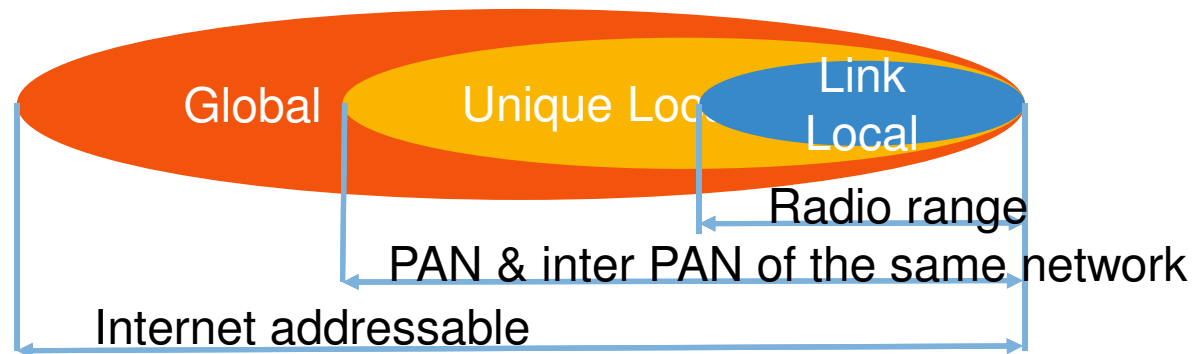
2001:3344:5566:7788:99AA:BBCC:DDEE:FF00/64

IPv6 Address Classes

- **Unicast** – Identifies a single Media Interface on a single network host:
Network Prefix: 64 bits
Network Address = Interface Identifier (IID): 64 bits
- **Multicast** – Identifies all network hosts which have assigned a multicast group to one or more Media Interfaces
Multicast Prefix: 8 bits all 1s (multicast address format FF::)
Flags: 4 bits
Scope: 4 bits
Multicast Group: 112 bits
- **Anycast** – Identifies a unicast-like address assigned to multiple interfaces where the final destination of packets can be any of the nodes using the address

IPv6 Unicast Address Scopes

- **Scopes** specify the boundaries of networks when using and forwarding packets for an address
- Defined by IANA: <http://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xhtml>
 - **Link Local Scope** – Addresses can be used only on the same network segment which shares link layer. Prefix: FE80::/10
 - **Unique Local Scope** – Addresses which are private to an administratively configured network and cannot be routed outside the network. Prefix: FC00::/7 (FC00::/8 and FD00::/8)
 - **Global Scope** – All other valid, non-reserved addresses. Prefix (currently): 2000::/3



Thread IPv6 Unicast Address Scopes

- **Link Local scope** applies only to Thread neighbors in direct IEEE 802.15.4 radio range (1 “hop”). Thread Link Local scope addresses are used primarily for link establishment and link maintenance between neighbors.
- A single, specific **Unique Local scope** prefix based on the Extended PANID is used for each Thread network to assign a set of addresses for the Thread interfaces used on the same Thread network.
- The specific prefix UL prefix is called the **Mesh Local (ML) Prefix**. Addresses assigned with the ML Prefix are called **Mesh Local (ML) Addresses** and may be used network-wide across multiple hops.
- **Mesh Local Prefix** = FD:<XPANID-bytes-1-to-5>::/48
- Thread interfaces may be assigned other **supplemental Unique Local scope** or **Global scope** addresses based on administrative, user, or application configuration.

Thread IPv6 Multicast Scopes

- **Link Local scope** and **Realm-Local scope** multicast addresses are defined generically for IPv6 multicast for All Nodes FF0x::1 and All Routers (FF0x::2)
- Assigned by IANA: <http://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xhtml>
- LL and RL scopes used in Thread networks for packet transmission based on the following rules:

FF02::1	All Neighbors but not forwarded to Low Power/Sleepy EDs
FF02::2	All Router Neighbors but not forwarded to Low Power/Sleepy EDs
FF03::1	All Nodes in Mesh but not forwarded to Low Power/Sleepy EDs
FF03::2	All Routers in Thread Mesh but not forwarded to Low Power/Sleepy EDs

IPv6 Configuration Overview

- Once joined to a network, a Thread device:
 - assigns at least 3 Unicast IPv6 addresses to the Thread Interface:
Link local address (LL64): fe80::c180:1192:c3ea:ef55
Mesh local address (ML64, ML-EID): fda3:217b:338e::213e:34b4:2659:10f8
Mesh local address (ML16, RLOC): fda3:217b:338e::ff:fe00:400
 - assigns All Thread Nodes multicast addresses:
Link local all Thread Nodes(Multicast): ff32:40:fda3:217b:338e::1
Realm local all Thread Nodes(Multicast): ff33:40:fda3:217b:338e::1
- **Mesh Local Prefix** is based on Extended PAN ID bytes 1-5
- **LL64 interface ID** is based on IEEE 802.15.4 Random Extended Address
- **ML-EID interface ID** is random
- **RLOC interface ID** is based on IEEE 802.15.4 Short address (6-bits Router ID + Child ID)
- Use `ifconfig` command in Kinetis Thread Stack shell to obtain IP address configuration

Ping From R1 to R2

In the R1 shell, type the **help ping** command to list usage. See example shown below to ping R2 on its IPv6 (ML16 or ML64) address:

```
$ ping fd06:d333:cdd3::ff:fe00:400
Pinging fd06:d333:cdd3::ff:fe00:400 with 32 bytes of data:
Reply from fd06:d333:cdd3::ff:fe00:400: bytes=32 time=19ms
Reply from fd06:d333:cdd3::ff:fe00:400: bytes=32 time=33ms
Reply from fd06:d333:cdd3::ff:fe00:400: bytes=32 time=30ms
Reply from fd06:d333:cdd3::ff:fe00:400: bytes=32 time=31ms
```

Try some CoAP commands from R1 to R2

In the R1 shell, try some of the following commands :

```
$ coap CON GET fd06:d333:cdd3::912f:7992:466d:e45a /temp  
coap rsp from fd06:d333:cdd3::912f:7992:466d:e45a ACK Temp:24.08
```

```
$ coap CON POST fd06:d333:cdd3::912f:7992:466d:e45a /led toggle  
coap rsp from fd06:d333:cdd3::912f:7992:466d:e45a ACK
```

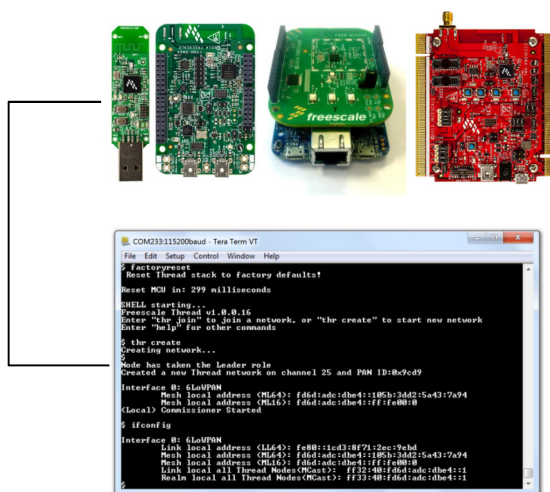
```
$ coap CON POST fd06:d333:cdd3::912f:7992:466d:e45a /led rgb r255 g000 b000  
coap rsp from fd06:d333:cdd3::912f:7992:466d:e45a ACK
```

```
$ coap CON POST fd06:d333:cdd3::912f:7992:466d:e45a /led flash  
coap rsp from fd06:d333:cdd3::912f:7992:466d:e45a ACK
```

Pressing SW2, 3 & 4 also send some command from leader to joiner, SW1 is for join / create command or factory reset is pressed more than 10 seconds.

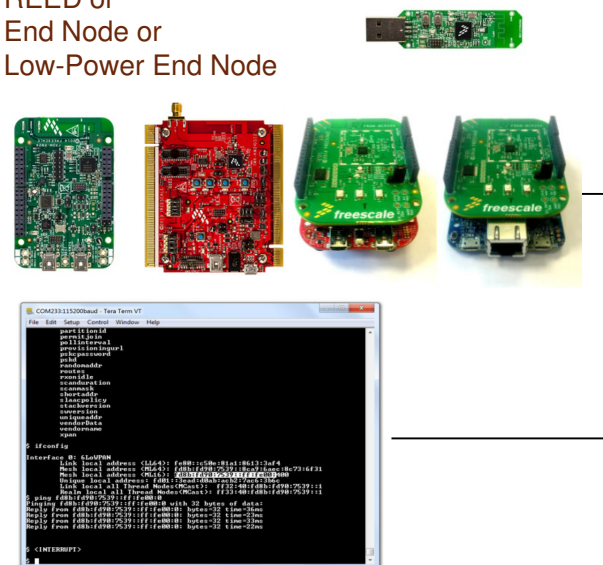
That was Kinetis Embedded Demos for Self Contained Networks

Router Eligible Devices



- CLI interface for REED and End Nodes
- Human readable commands implemented for network creation, commissioning, configuration, testing
- Push button demo to send Node temperature multicast to all nodes
- Simple push button lighting demo.

REED or
End Node or
Low-Power End Node



- CLI interface for REED and End Nodes
- Human readable commands implemented for setup nodes for commissioning, join network, network configuration, testing
- Push button demo to send Node temperature multicast to all nodes
- Simple push button lighting demo.

WIRESHARK



Sniffer with Wireshark + USB-KW24D512 or USB-KW40Z

- Download and install the **Kinetis Protocol Analyzer Adapter**



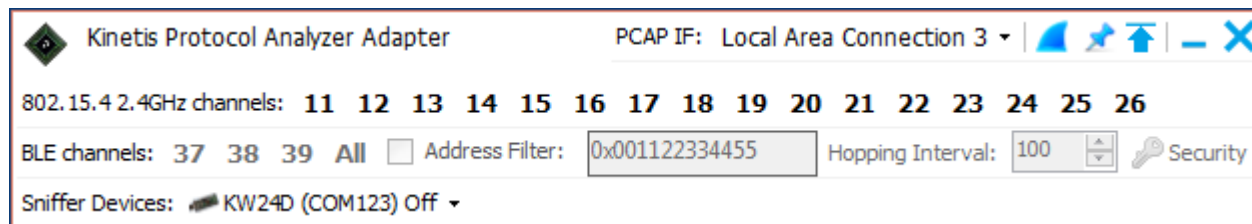
Kinetis Protocol Analyzer Adapter (REV 1.0.0)

Kinetis Protocol Analyzer Adapter enables the use of USB-KW40Z and USB-KW24D512 as a protocol analyzer for PCAP interfaces and Wireshark on Windows...

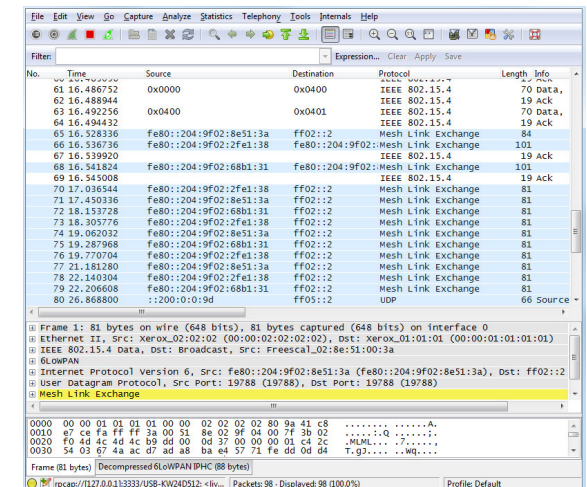
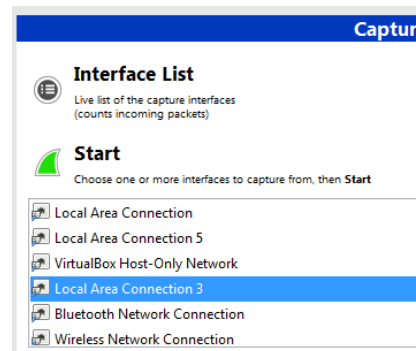
exe (16.0 MB) Kinetis Protocol Analyzer Adapter

10/12/2015

Download



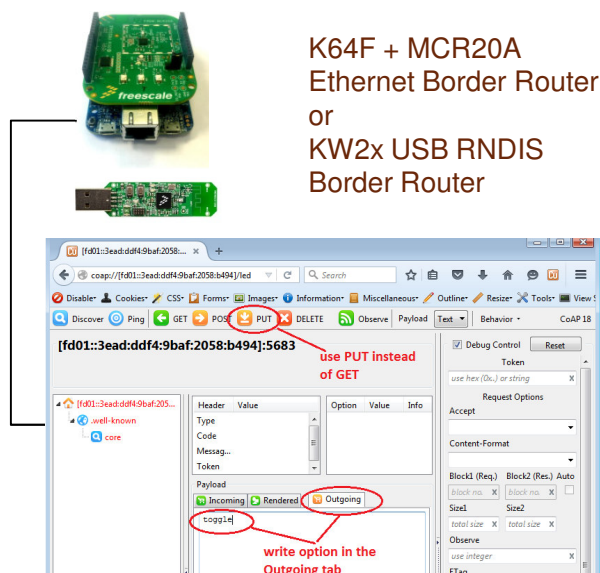
- Download and Install Wireshark.



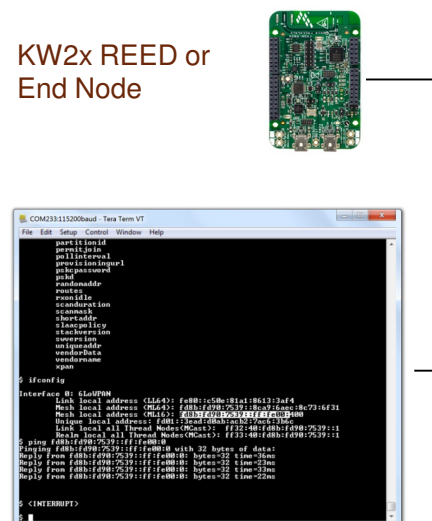
BORDER ROUTER



Kinetis Ethernet Border Router – Out of Box Demo

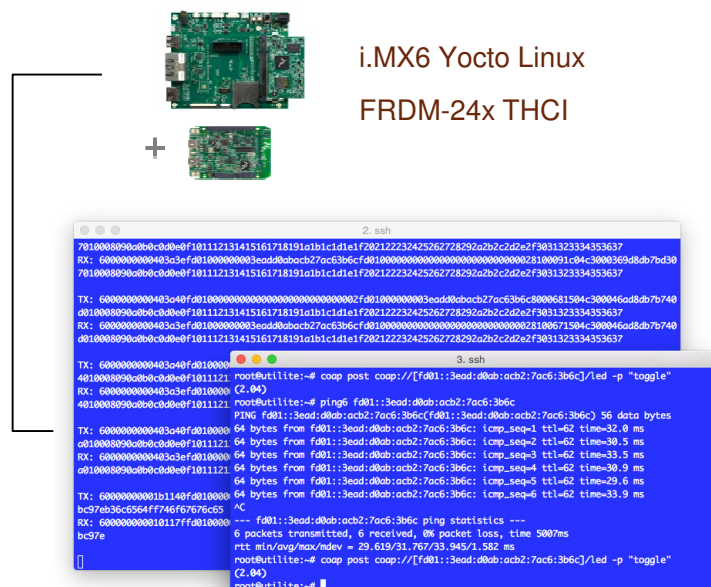


- Communicate with Thread devices from Windows via Ethernet or USB RNDIS interfaces
- CoAP (get/post) examples for remote temperature and LED control/toggle using Copper Firefox add-on

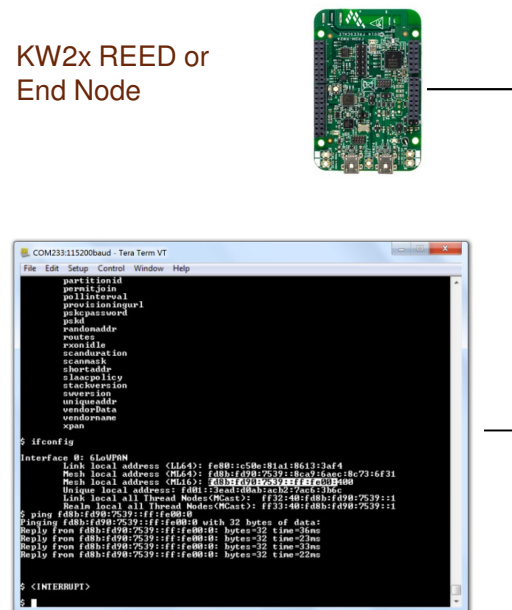


- CLI interface for REED and End Nodes
- Human readable commands implemented for network creation, commissioning, configuration, testing
- Push button demo to send Node temperature multicast to all nodes
- Simple push button lighting demo.

i.MX6 Linux Border Router + THCI – Out of Box Demo

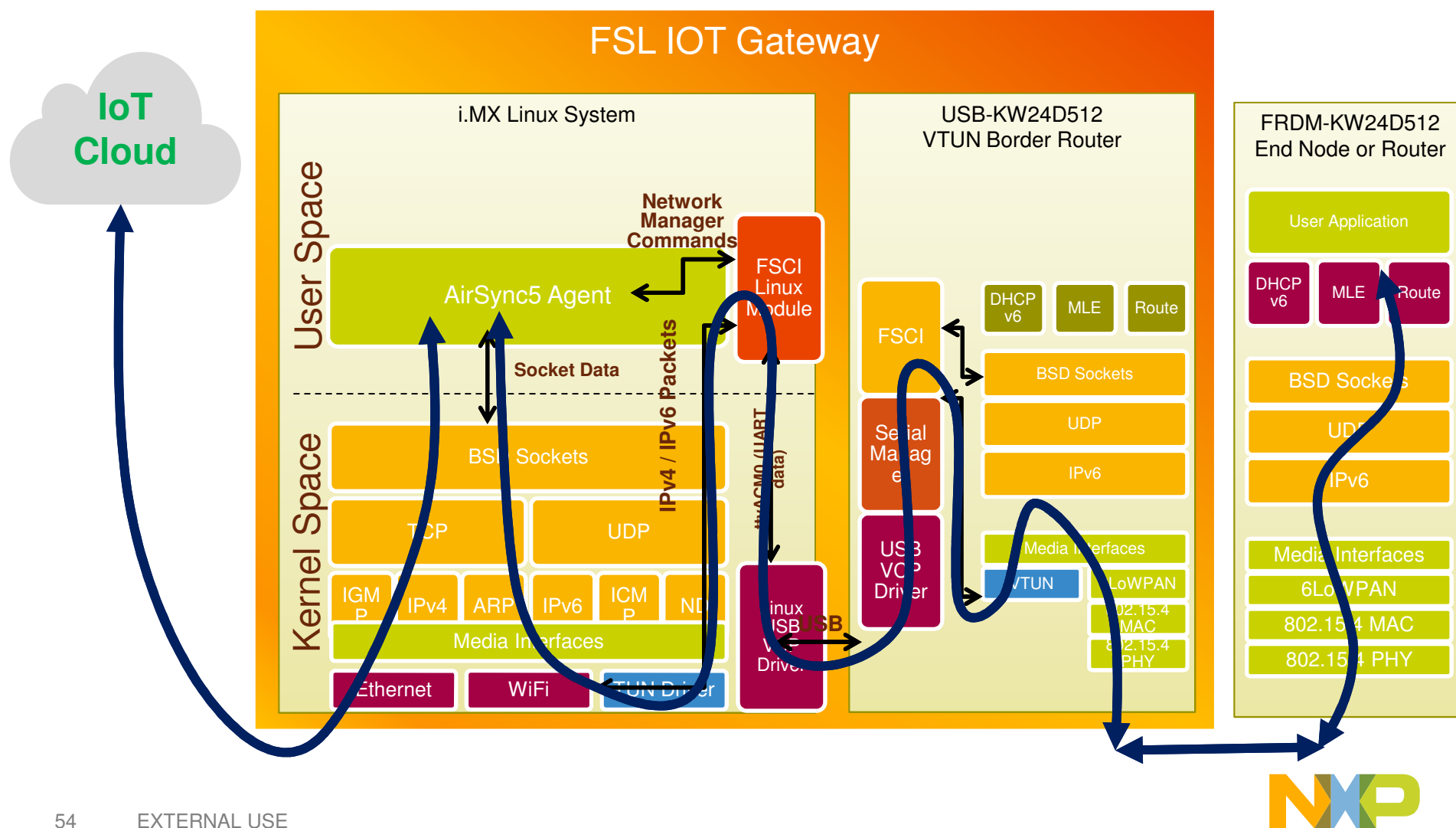


- Linux Host control interfaces
- C and Python Based Virtual TUN Tap drivers for IPv6 packet routing
- Commissioning example code
- CoAP (get/post) examples for remote temperature and LED control/toggle

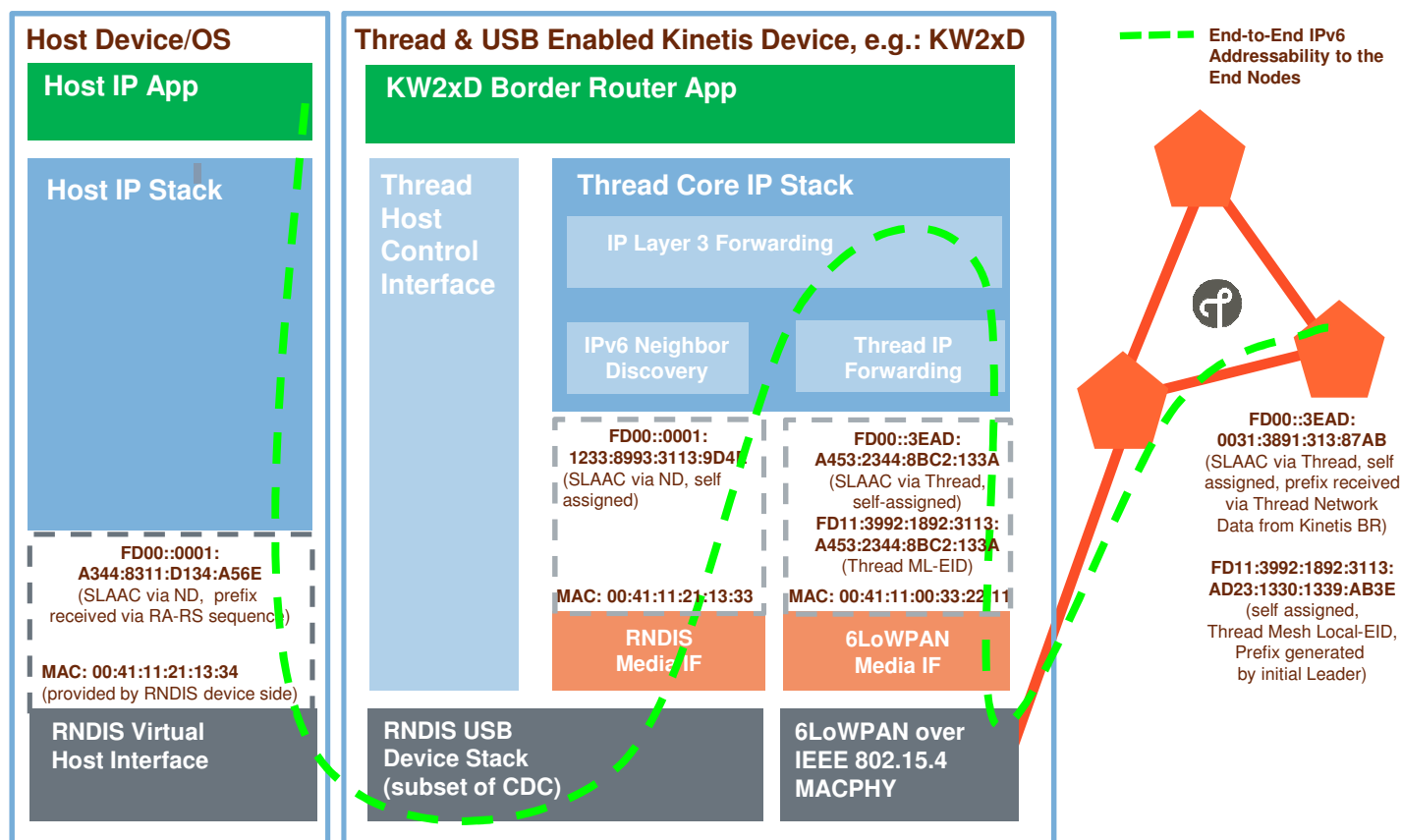


- CLI interface for REED and End Nodes
- Human readable commands implemented for network creation, commissioning, configuration, testing
- Push button demo to send Node temperature mutlicast to all nodes
- Simple push button lighting demo.

Example usage of Thread with FSCI and TUN Driver in a Linux i.MX System Paired with Kinetis KW2x



Example usage of Thread with RNDIS



FD00::0001:: – Default Prefix set on the RNDIS link, generated and advertised by KW2xD as ND router
FD00::3EAD:: – Default “global” On-Mesh Prefix assigned by to the Thread subnet
FD11:3992:1892:3113:: – Randomized Mesh Local-only Prefix used on the Thread subnet

Let's all join the same network

On IMX6UL-EVK, I will start network (or VTunStartCommissioner.py):

```
root@imx6ulevk-iotg:./host_sdk/hsdk/demo/bin/Thread_KW2xD_Tun  
/dev/ttymx1 fslthr0 &
```

```
[THR] Set 802.15.4 channel      OK!
```

```
[THR] Create Network           ALREADY CONNECTED!
```

```
[THR] Border Router Add Prefix OK!
```

```
[THR] Border Router Sync Prefix OK!
```

```
[MESH COP] Start Commissioner  OK!
```

```
[MESH COP] Add Expected Joiner OK!
```

```
[MESH COP] Sync Steering Data  OK!
```

In the R1 and/or R2 shell, factory reset and change channel and join :

```
$ factoryreset
```

```
$ thr set channel 25
```

```
$ thr join
```

Getting routing information

1. Try the **thr get neighbors**

```
$ thr get neighbors
```

Index	Extended Address	ShortAddr	LastTime	LinkMar
Child				
0	0x82F6E31A63812A23	0x0400	3	27

no



2. Type **thr get routes** in the shell to get the R1 Thread routing table information:

```
$ thr get routes
```

```
ID Sequence: 71
```

```
Router ID Mask: C000000000000000
```

RouterID	Short Address	Next Hop	Cost	NOut	NIn
1	0x0400	0x0400	1	3	3

Kinetis Thread Stack – Number of Nodes per network

- **Nodes per network** = Active Routers + End Devices (Router Eligible End Devices, Powered End Devices, Sleepy End Devices)
- **Number of Active Routers:**
 - managed adaptively by Thread protocol
 - regularly 16-32 active routers in larger network depending on connectivity thresholds
 - maximum of 32 active router IDs specified by protocol
- **Number of End Devices:**
 - Recommended number: 250 devices per subnet
 - Theoretical limit allowed by Thread specification: 16k devices
 - 32 Active Routers
 - 512 Children End Devices per Parent Active Router
 - Practical limit of Children depends on Parent memory capacity
 - Kinetis Thread Stack for KW24/KW22 default setting is 640 devices:
 - 32 Active routers
 - 20 Children per Parent, but can be increased with considering memory tradeoffs up to 50 Children

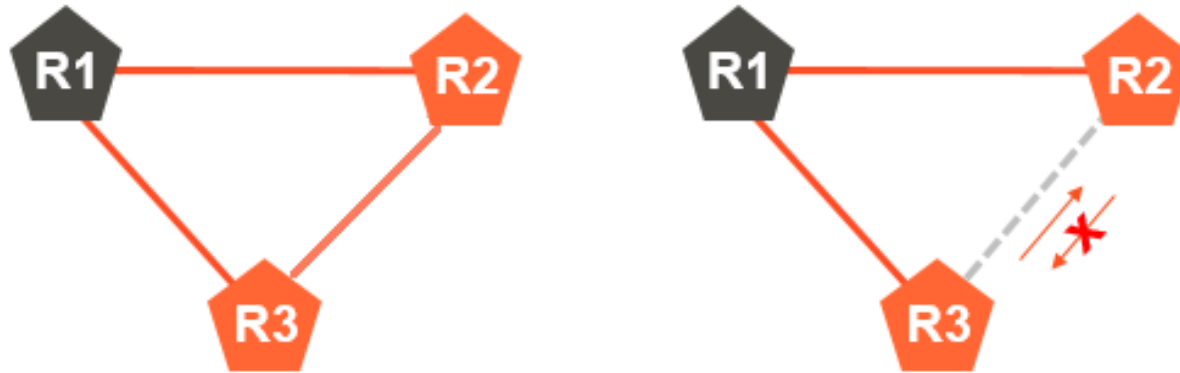
Routing Protocol – Link Margin, Quality and Cost

Link Margin	Quality	Link Cost
> 20dB	3	1
> 10dB	2	2
> 2dB	1	6
<= 2dB	0	infinite

The outgoing and incoming Link Qualities and Routing cost to an Active Router as advertised by any other Active Router can be encoded in 1 byte (per Router):

- Neighbor Out Quality – 2 bits (value 0 if self or not neighbor)
- Neighbor In Quality – 2 bits (value 0 if self or not neighbor)
- Routing Cost (minimal sum of Link costs) – 4 bits

MAC filtering and self healing network



On the Thread Router R2, get the node MAC random address :

```
$ thr get randomaddr
```

```
randomaddr: 0xF310BB8D2434C450
```

Enable the MAC filtering on router R3 :

```
$ macfilter enable accept
```

```
$ macfilter add 0xF310BB8D2434C450 reject 1 lqi 255
```

This will only be valid after 100s (Last time field), ping R2 from R3 :

```
$ ping fd1a:a020:4a4::ff:fe00:400 -t
```

```
Pinging fd1a:a020:4a4::ff:fe00:400 with 32 bytes of data:
```

```
Request timed out.
```

Until network find a different way through R1. Disable macfilter :

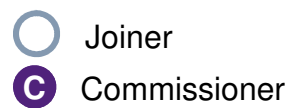
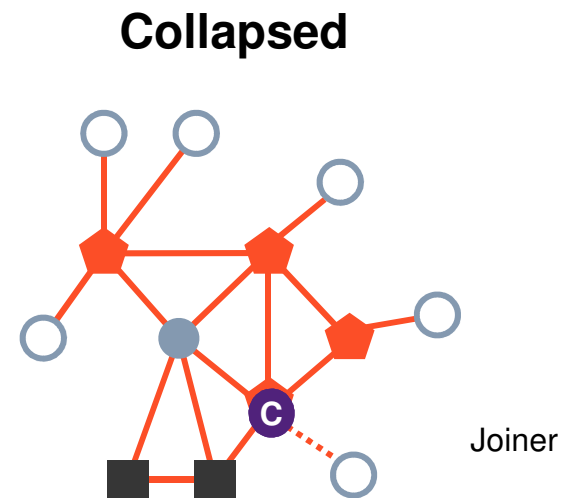
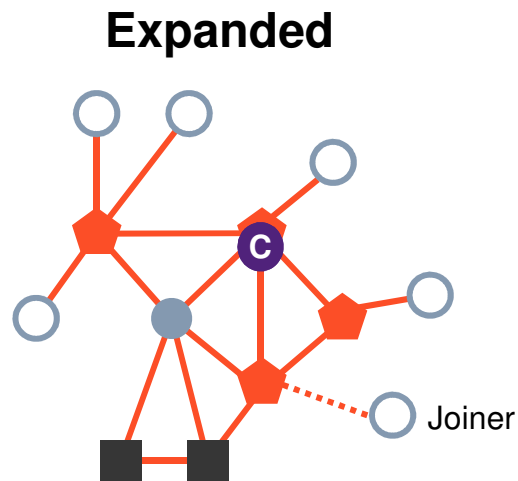
```
$ macfilter disable
```

Now powering off R1 to check who will become the new chosen Leader (after 100s) !

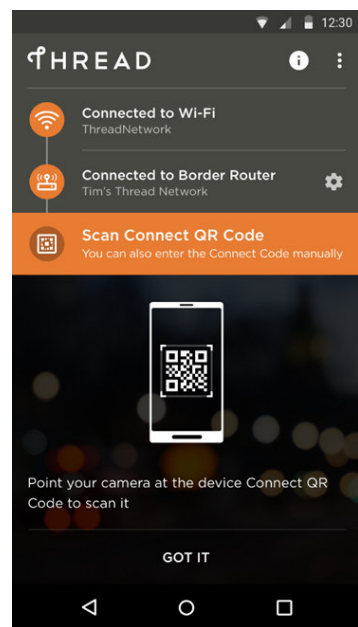
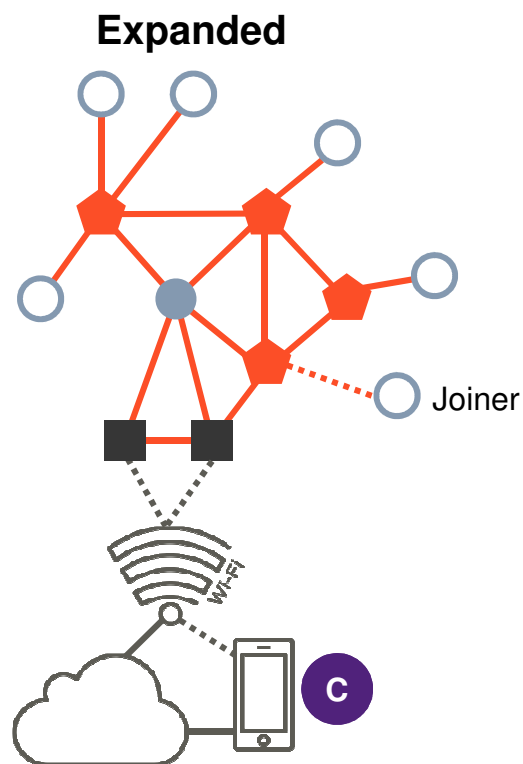
COMMISSIONING



Topology in-band (in-mesh) commissioner

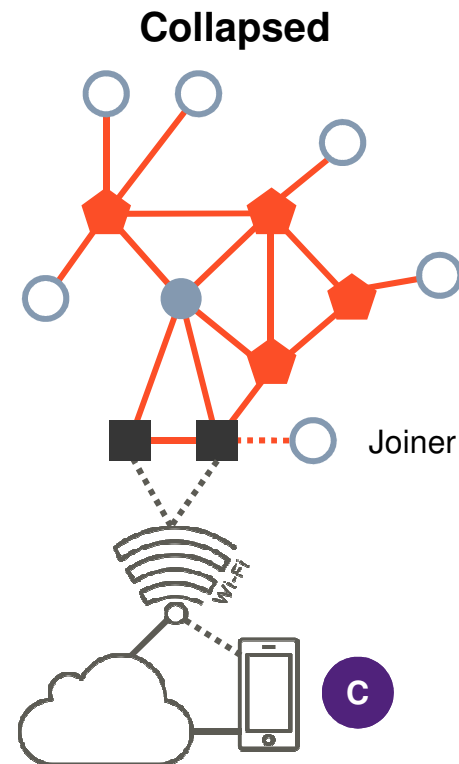


Topology out-of-band commissioner (eg. WiFi, BLE, NFC, other) : demo of relay script



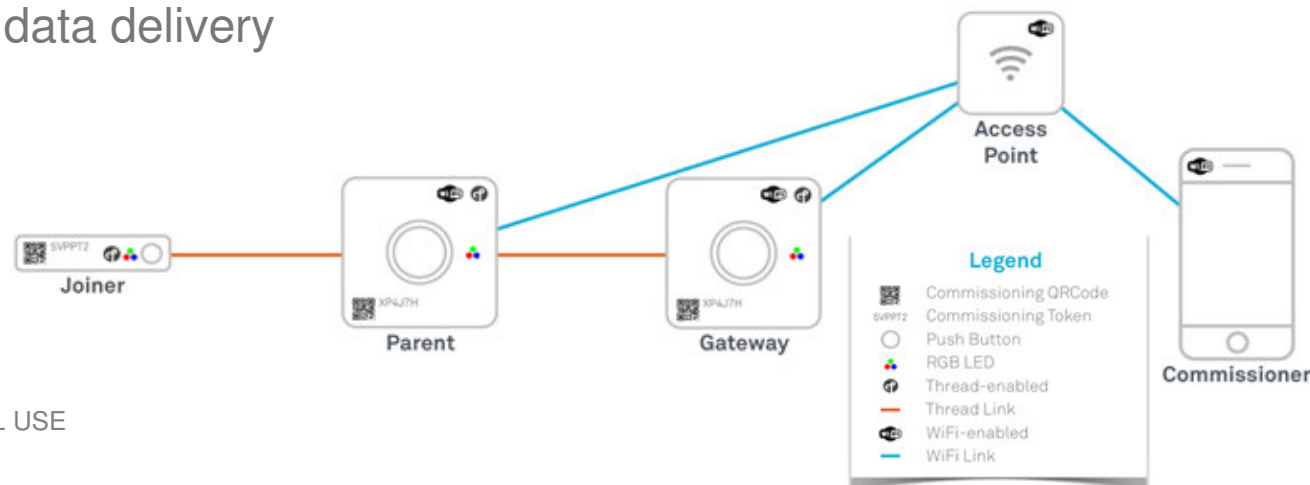
Thread Commissioning
Mobile App

- Joiner
- Ⓢ Commissioner

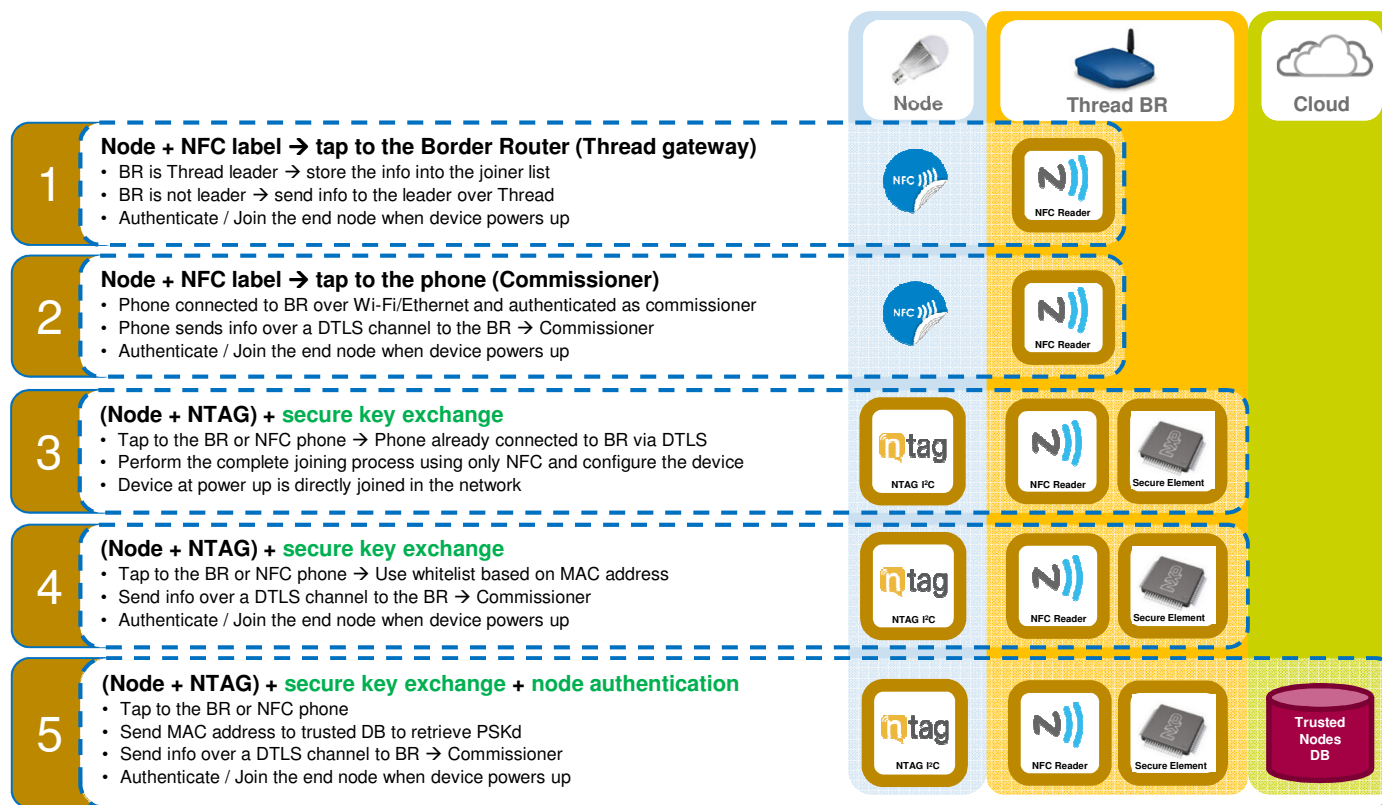


Secure Commissioning

- Joiner device will regularly be commissioned by a smart phone
- The smart phone does not have to connect directly to the Thread network. It can connect via Wi-Fi then through a Thread Border Router Gateway
- Commissioning is based on **end-to-end security and encryption between Thread Nodes and the Commissioner**. Other nodes relay DTLS but cannot decrypt
- Shared secret usually established by scanning device QR code
- **Elliptic Curve J-PAKE** used to generate high-strength key used for message protection and secure configuration data delivery



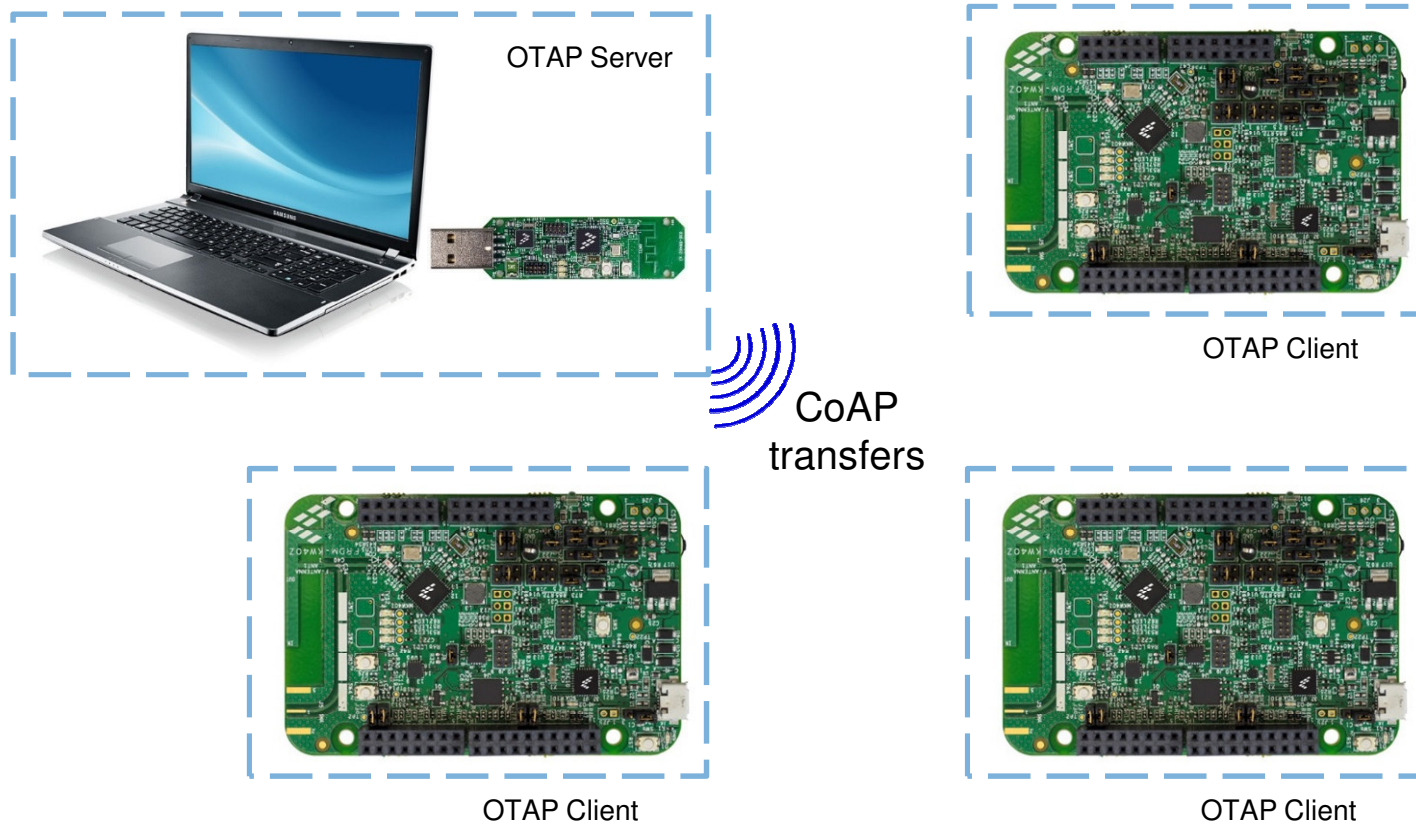
Example: Thread Device Out-of-Band commissioning with NFC : demo of multi_nfc script



OTHER DEMO



Kinetis Thread - Over The Air Firmware Update





SECURE CONNECTIONS
FOR A SMARTER WORLD