



2014.8/23 - 24

at Softpia Japan Center Building
Gifu, JPN

TOCOS **TWE-Lite**をつかってみた
「トワイライトの世界にようこそ！」



matsujirushi(Takashi Matsuoka)

2014/8/17 17:00～

自己紹介

✓ **電子工作** やります。

✓ EAGLE CAD

✓ **プログラミング** やります。

✓ PIC10F322

✓ PIC12F629,675,683,1822

✓ PIC16F88,628

✓ PIC18F1320

✓ PIC18F2550,4550

✓ ATmega64A

✓ LPC1114

✓ Visual Studio

✓ 1年前から **レーザー加工**。

✓ LaserVelocity / DraftSight



Takashi Matsuoka
(matsujirushi)



takashi.matsuoka.37



@matsujirushi12

エンベデッドシステムスペシャリスト
データベーススペシャリスト
マイクロソフト認定テクノロジースペシャリスト
ORACLE MASTER
電気工事士
工事担任者



出展

2012年

2013年

2014年

ぼっち

一緒に

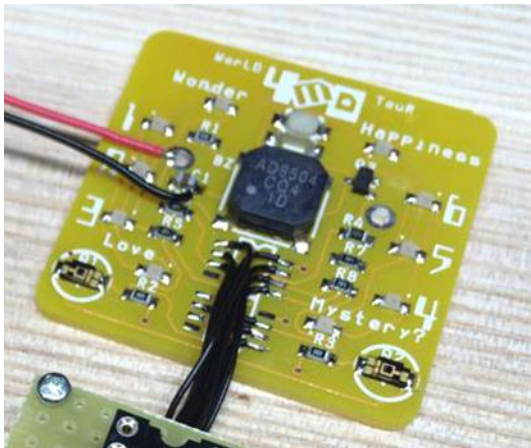
後方支援

● Make Ogaki Meeting 2012 ● クリエーターズマーケット vol.28

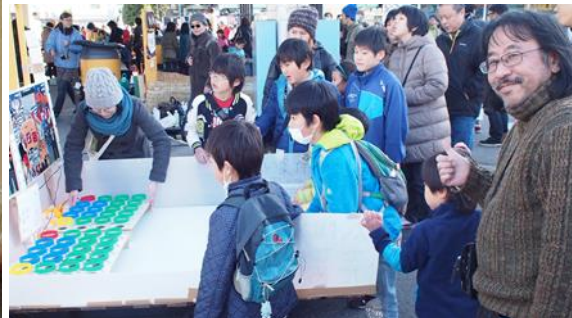
● Maker Faire Tokyo 2012 ● CONSOLARE 2013 EXHIBITION



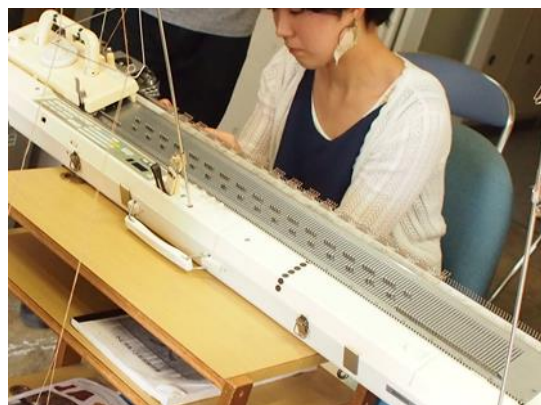
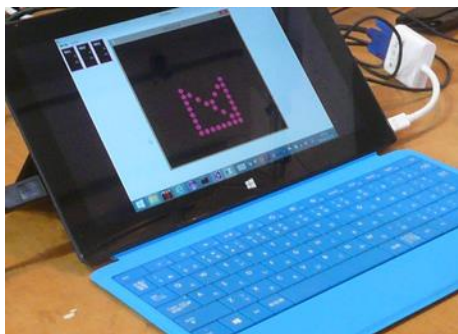
後方支援プロダクト



4MO/バッヂ



跳跳樂



編み機HACK

プレゼンの動機



オモチャ到着。



フレームワークが独特。体系的に知りたい。

買ったけど、標準機能(リモートI/O)は使い道が無いなあ。



プログラマブル！開発についてkws k !



TWE-Liteのプログラミングをプレゼンするぞー！

トワイライトの世界にようこそ！

✓ 気になる特徴

- 通信距離
- 消費電力
- ToF(タイム・オブ・フライト)
- パケットスニファ
- ネットワークディスカバリー

✓ 内蔵マイコンのプログラミング

- フレームワーク
- ワイヤレス
- ペリフェラル

本資料はmatsujirushiが作成したものであり、東京コスモス電機が保障するものではありません。

TOCOS TWEシリーズ

トワイライト

超小型装置の**無線化**を
実現できます。

<http://tocos-wireless.com/jp/products/TWE-001Lite.html>



TWE-EH Solar



TWE-Lite

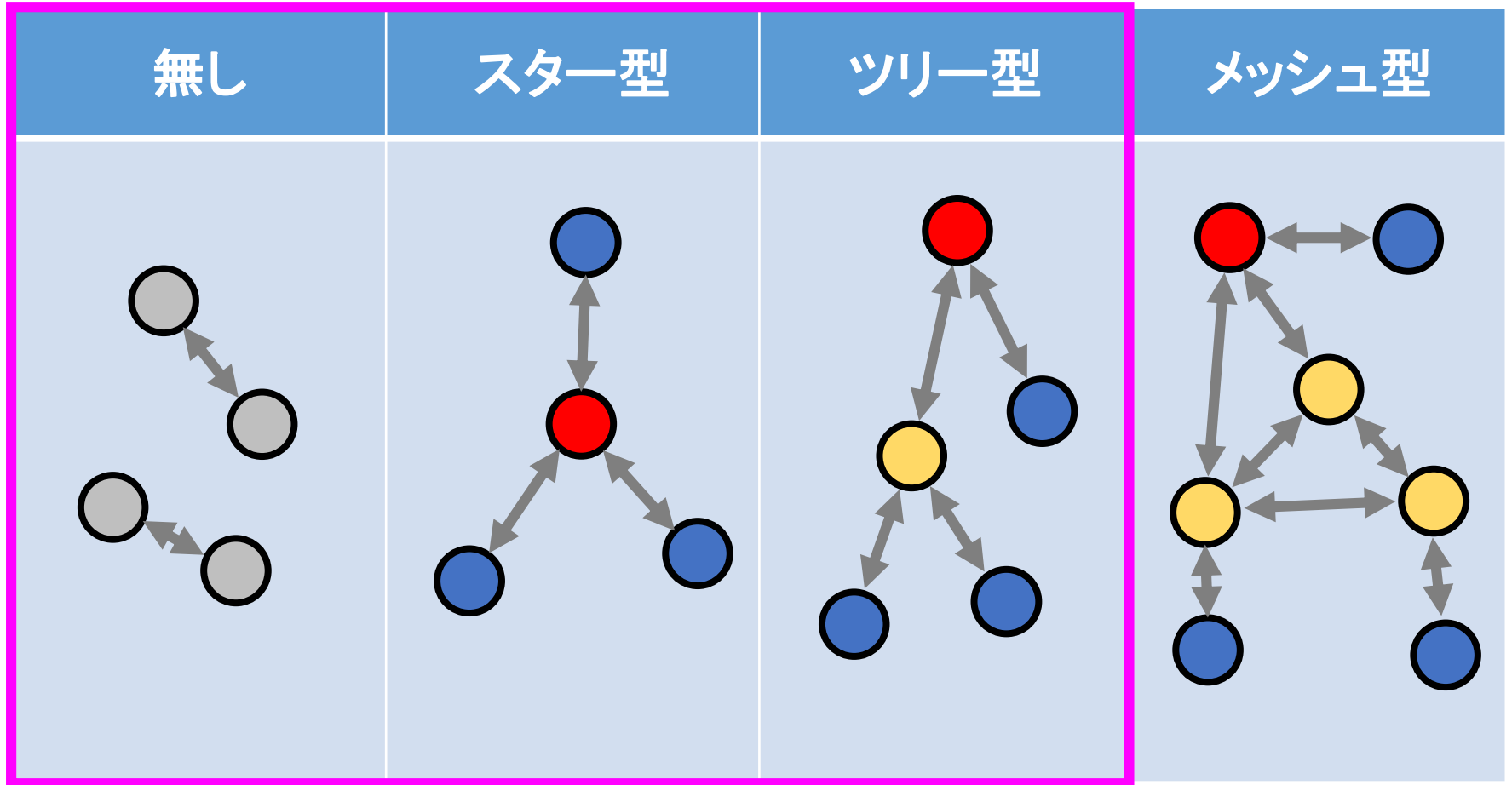


TWE-Lite DIP



ToCoStick
(TWE-Lite USB)

ネットワーク形態



 COORDINATOR

 ROUTER

 END DEVICE

TWE-Lite 気になる特徴

<http://tocos-wireless.com/jp/products/TWE-001Lite.html>

- ✓ 通信距離は**1km**
- ✓ コイン電池で**数年**動作
- ✗ ~~動作温度 **-40~105°C**~~
- ✓ **ToF**距離測定
- ✓ パケット**スニファ**
- ✓ ネットワーク**ディスカバリー**ツール
- ✓ 内蔵マイコンで**制御**可能



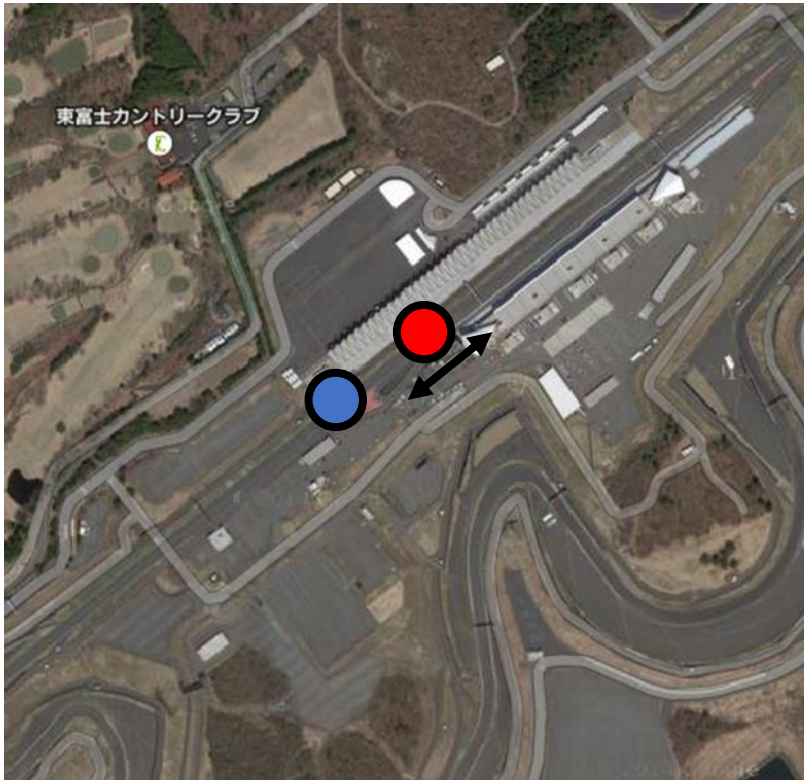
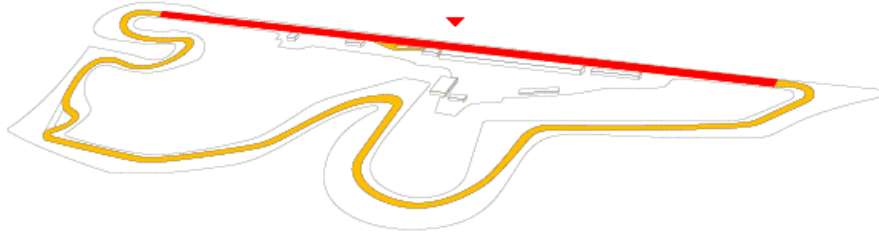
通信距離は1km

TWE-Liteは低消費電力を保ちつつ見通しで外部アンテナ（利得2dBi、無指向性）使用時に1kmと非常に長い到達距離を実現しております。

<http://tocos-wireless.com/jp/products/TWE-001Lite.html#TWE-001-12>

通信距離を測ってみた

某サーキット メインストレート(約1.5km)



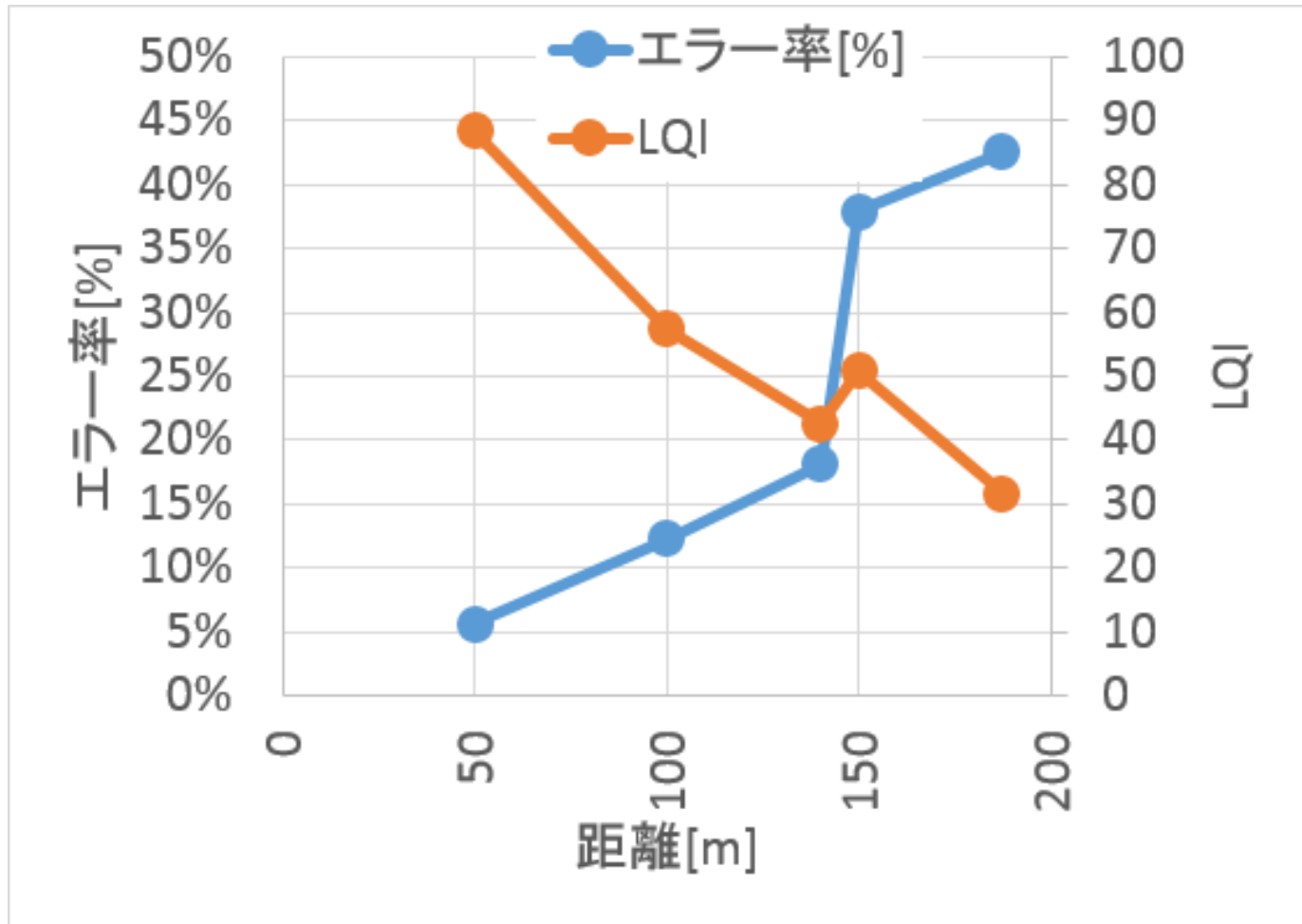
● END DEVICE
TWE-Lite + CR2032 高さ1m



● COORDINATOR
TWE-Lite + 単三x2 + パソコン 高さ1m



測った結果



通信距離限界は**180m**

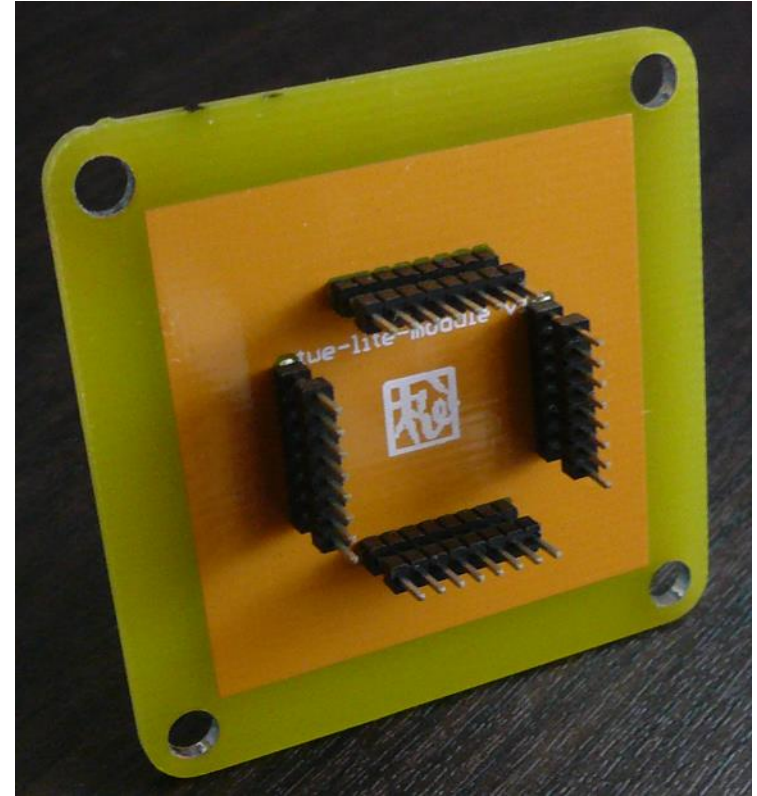
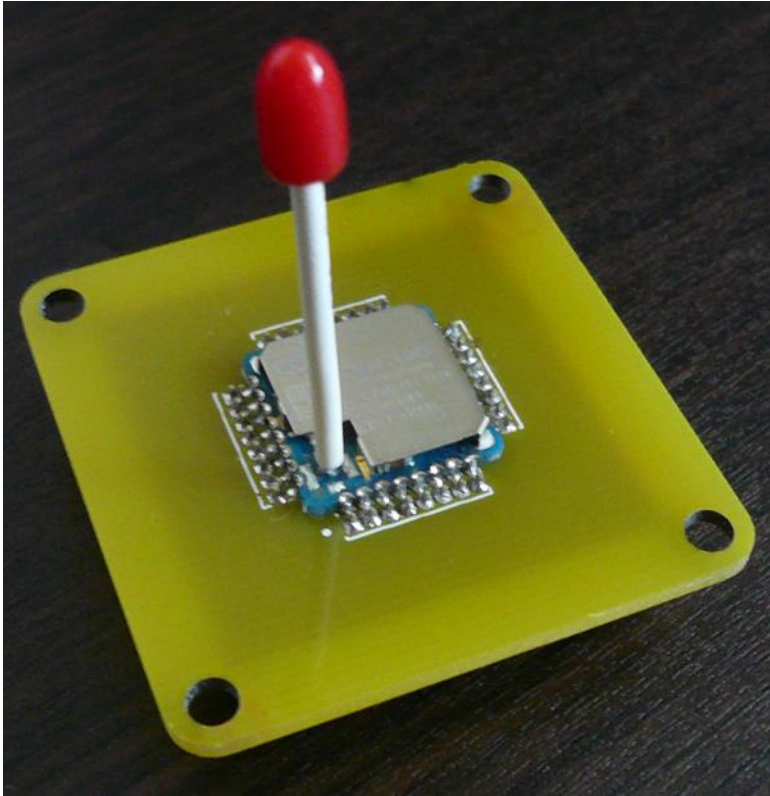
基板パターン？

マッチ棒アンテナは、TWE-LITEを実装する**プリント基板パターン**のレイアウトにより利得が大きく変化します。

モジュール裏面に**30mm × 30mm**の銅箔面を作成の上、**電池BOX**、ベタGNDなどを可能な限り遠ざけます。

http://tocos-wireless.com/jp/tech/Hardware_guide/Lite_Guide/Lite_MotherPCB_Ant_Artwork.pdf

基板作った



基板裏面に、**30 x 30mm** の銅箔

通信距離を測ってみた 2回目

名古屋市北区 新川堤防(約600m)



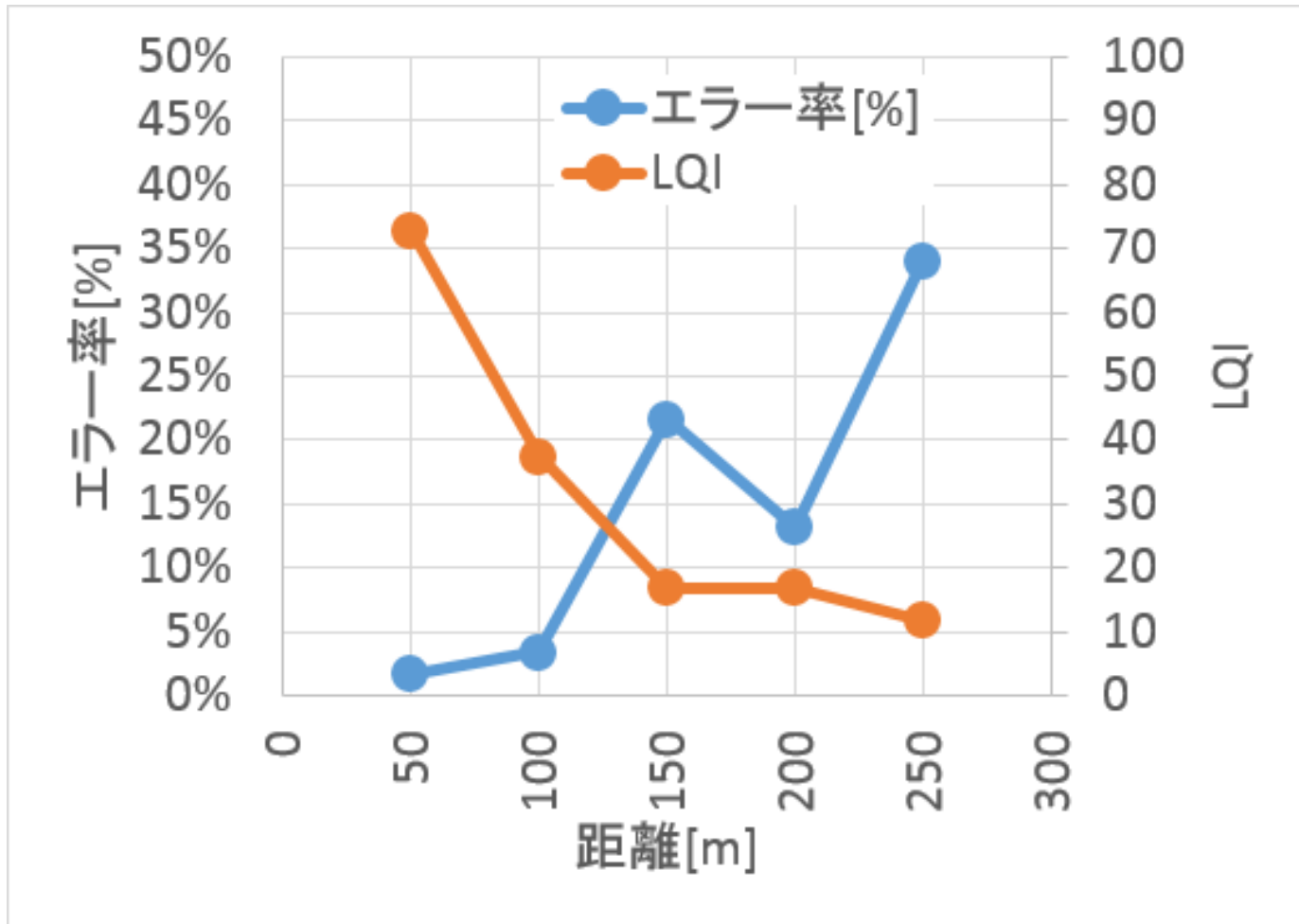
● END DEVICE
TWE-Lite + 単三x2 高さ1.2m



● COORDINATOR
TWE-Lite + 単三x2
+ パソコン 高さ1.2m



測った結果 2回目



通信距離限界は250m

高さ不足？

アンテナの位置が十分に**高い**場合(※1)に最も通信距離が長くなります。

※1 端末間が100mでの通信の場合は約1m、1Kmの場合は**約3m**の高さが必要です。

電波ノイズが少ない**環境**で通信距離が延びます。

電波ノイズが多い街中や自動車道路、工場等の近所では通信距離が短くなります。

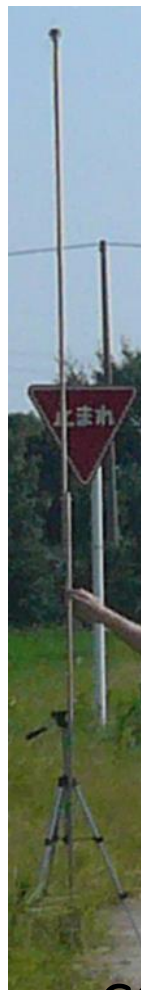
<http://tocos-wireless.com/jp/products/TWE-Lite-DIP/range.html>

通信距離を測ってみた 3回目

稲沢市 田園(約550m)



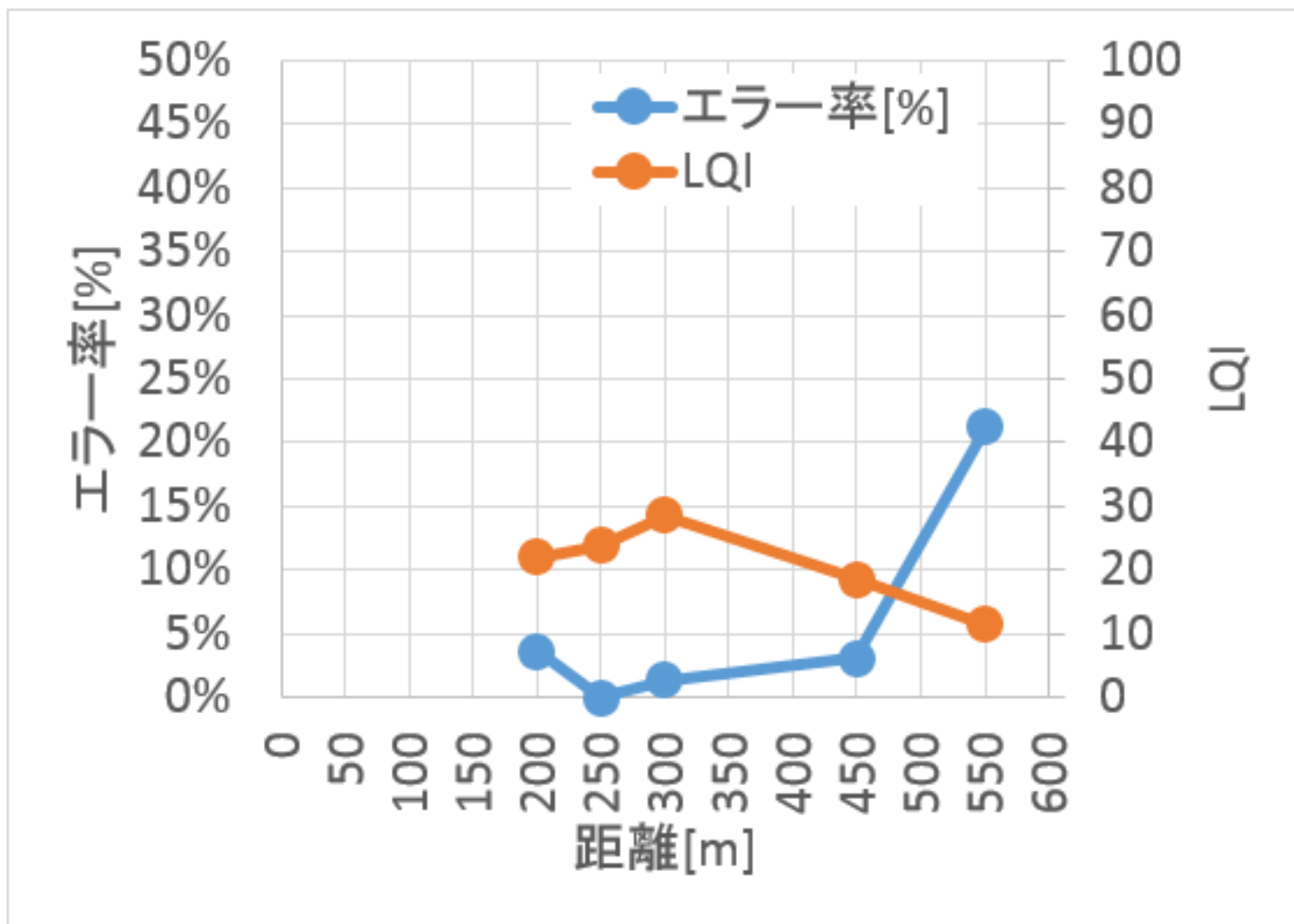
● END DEVICE
TWE-Lite + 単三x2 高さ3.2m



● COORDINATOR
TWE-Lite + 単三x2
+ パソコン 高さ3.2m

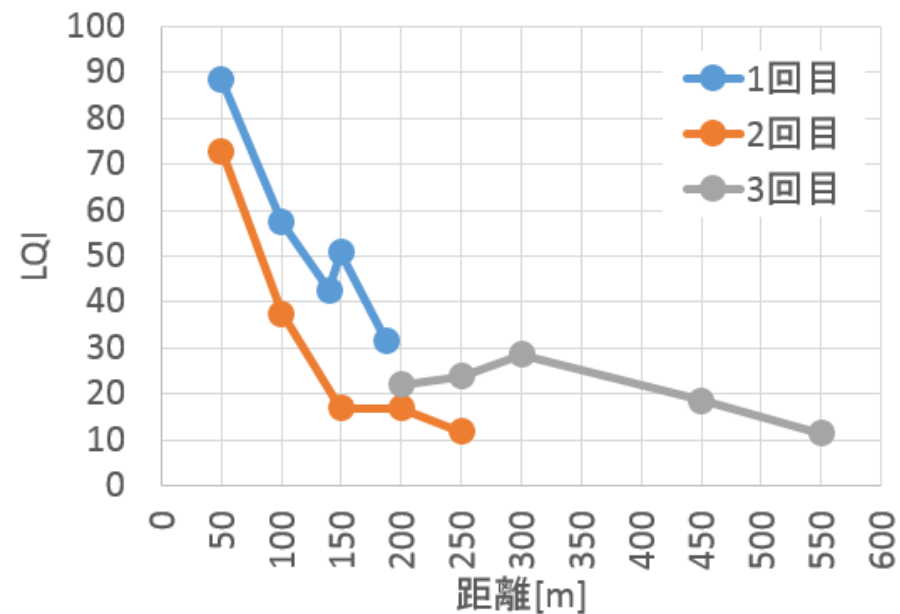
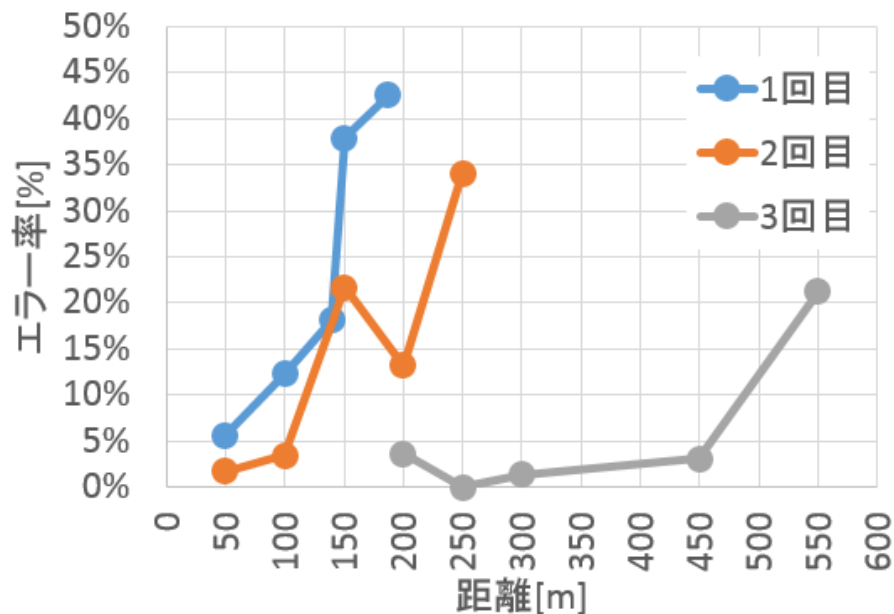


測った結果 3回目



通信距離限界は**550m以上!**

通信距離は1km？



- ✓ 実験結果は通信距離**550m以上**。
- ✓ **電波ノイズ**が少なく、見通し1kmの場所が無いと分からない。

メーカーより
高さ不足と
のアドバイス
有り。

1km飛ばすには5.6m

メーカーより
高さ不足と
のアドバイス
有り。

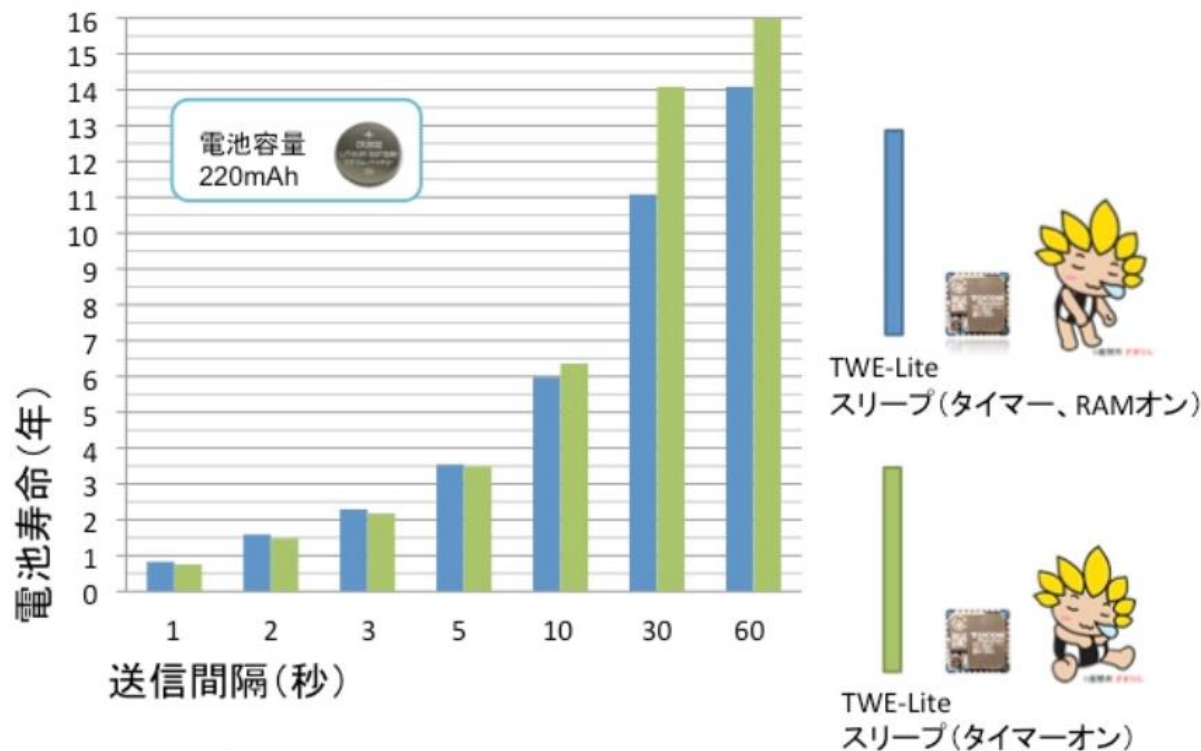
アンテナ間が100mの場合、2.4GHzは1.8m、920MHzは2.9m
のアンテナ高が必要です。**1000m**の場合、2.4GHzは**5.6m**、
920MHzは9mのアンテナ高が必要です。

<http://tocos-wireless.com/jp/products/TWE-001Lite.html#TWE-001-11>

コイン電池で数年動作

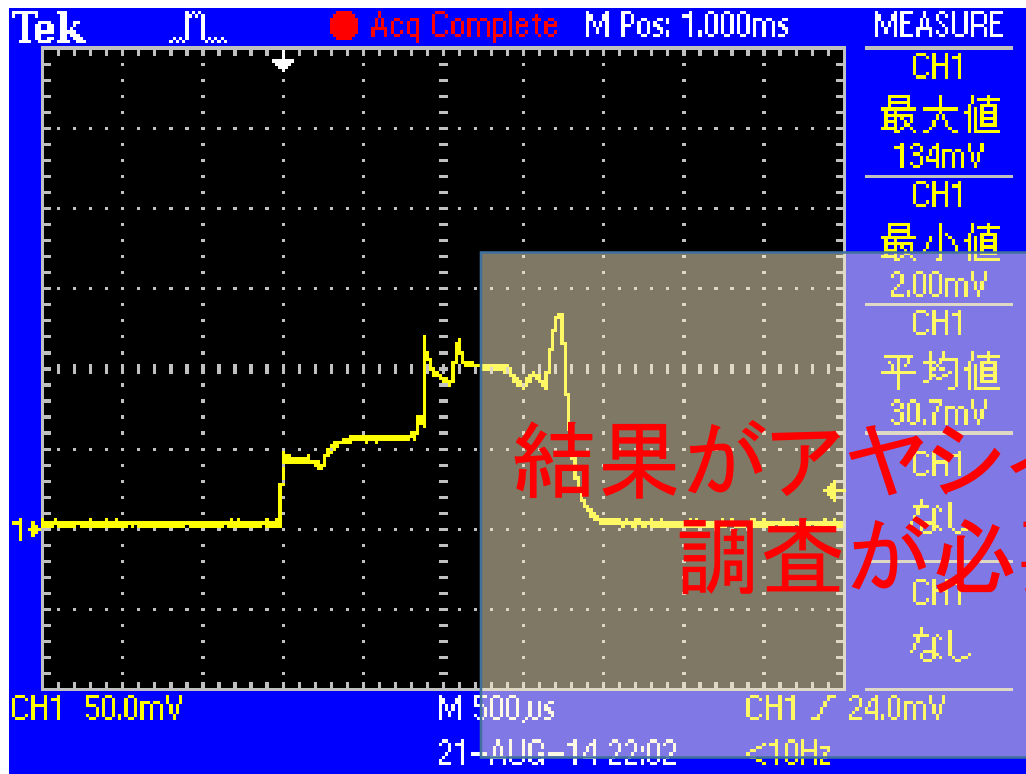
CR2032使用時、**3秒間隔**の送信で**2年**以上。

<http://tocos-wireless.com/jp/products/TWE-001Lite.html#TWE-001-11>



※ 暗電流はデータシート値より計算

電流を測ってみた



- ✓ ToCoNet_bMacTxReq()
- ✓ シヤント抵抗は、10[Ω]

結果がアヤシイので再調査が必要。

だいたい、75[mV]、1.75[ms]

→ 7.5[mA]、1.75[ms]

→ 3秒間隔の場合、 $7.5[\text{mA}] * 1.75[\text{ms}] / 3[\text{s}] = 4.375[\mu\text{A}]$

→ CR2032(220mAh)の場合、 $220[\text{mAh}] / 4.375[\mu\text{A}] = 50285[\text{h}] = 5.7\text{年}$

ToF距離測定

TWE-Liteには電波を使用して距離を測定する際に電波強度(RSSI)を使用する方法に加え**電波の往復時間**による計測距離に対して誤差がほぼ一定な距離測定機能**タイム・オブ・フライト(ToF)**を搭載しております。

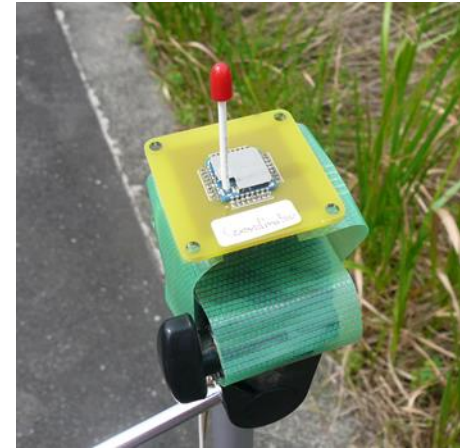
<http://tocos-wireless.com/jp/products/TWE-001Lite.html#TWE-001-1>

ToFを試してみた

名古屋市北区 新川堤防(約600m)



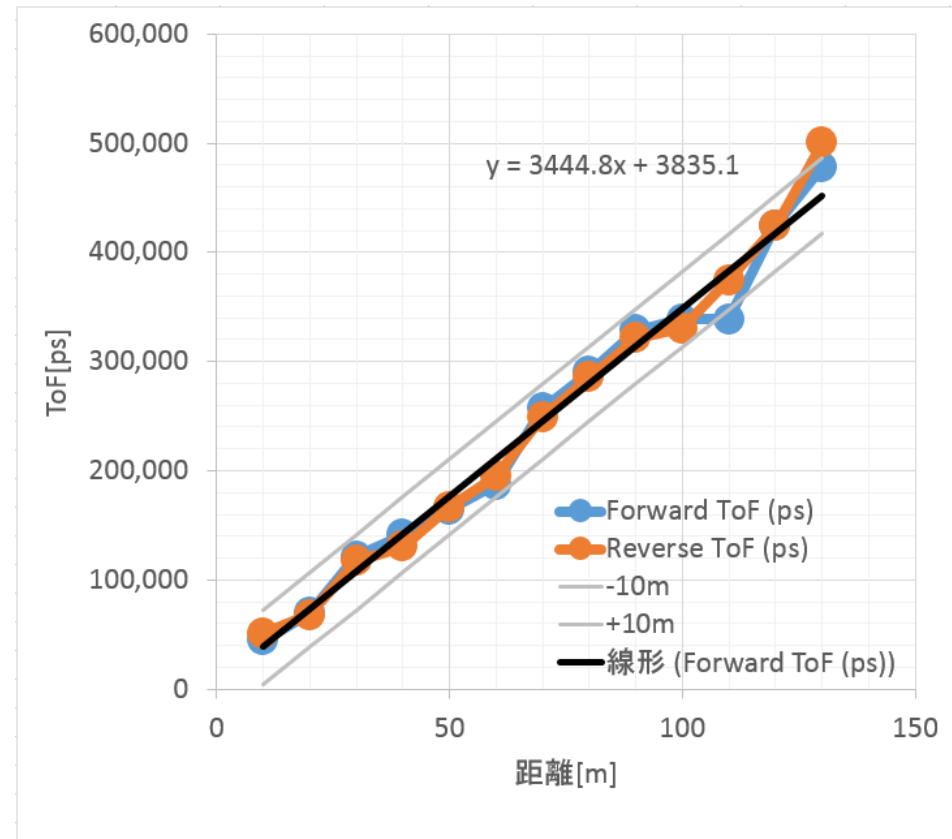
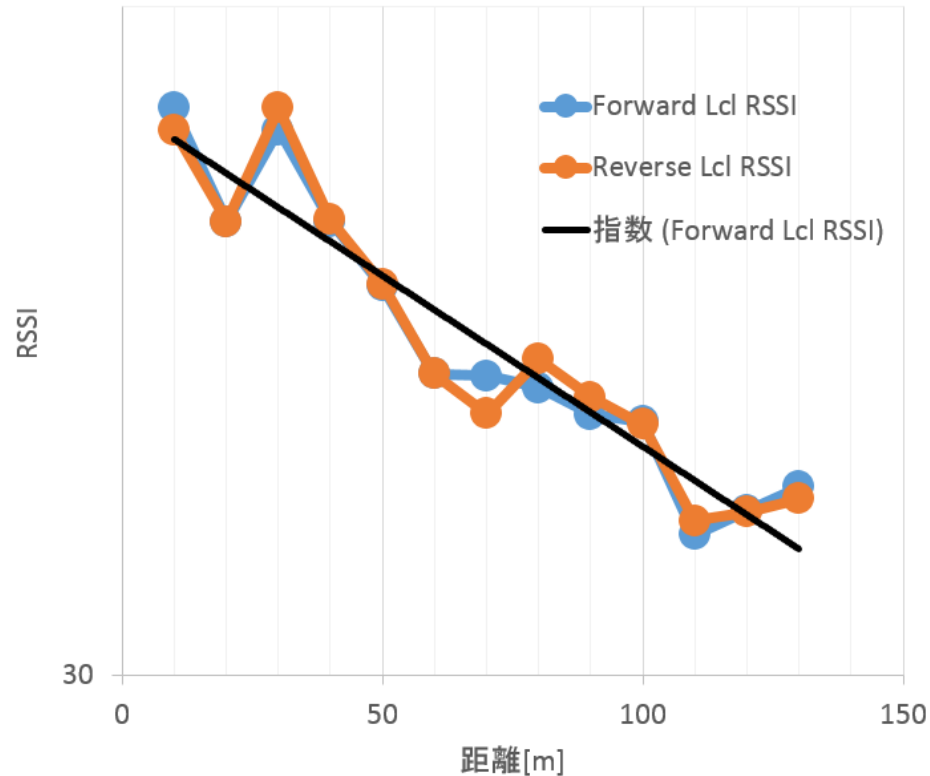
COORDINATOR
TWE-Lite + 単三x2
高さ1m



END DEVICE
ToCoStick
+ パソコン 高さ1m



試した結果

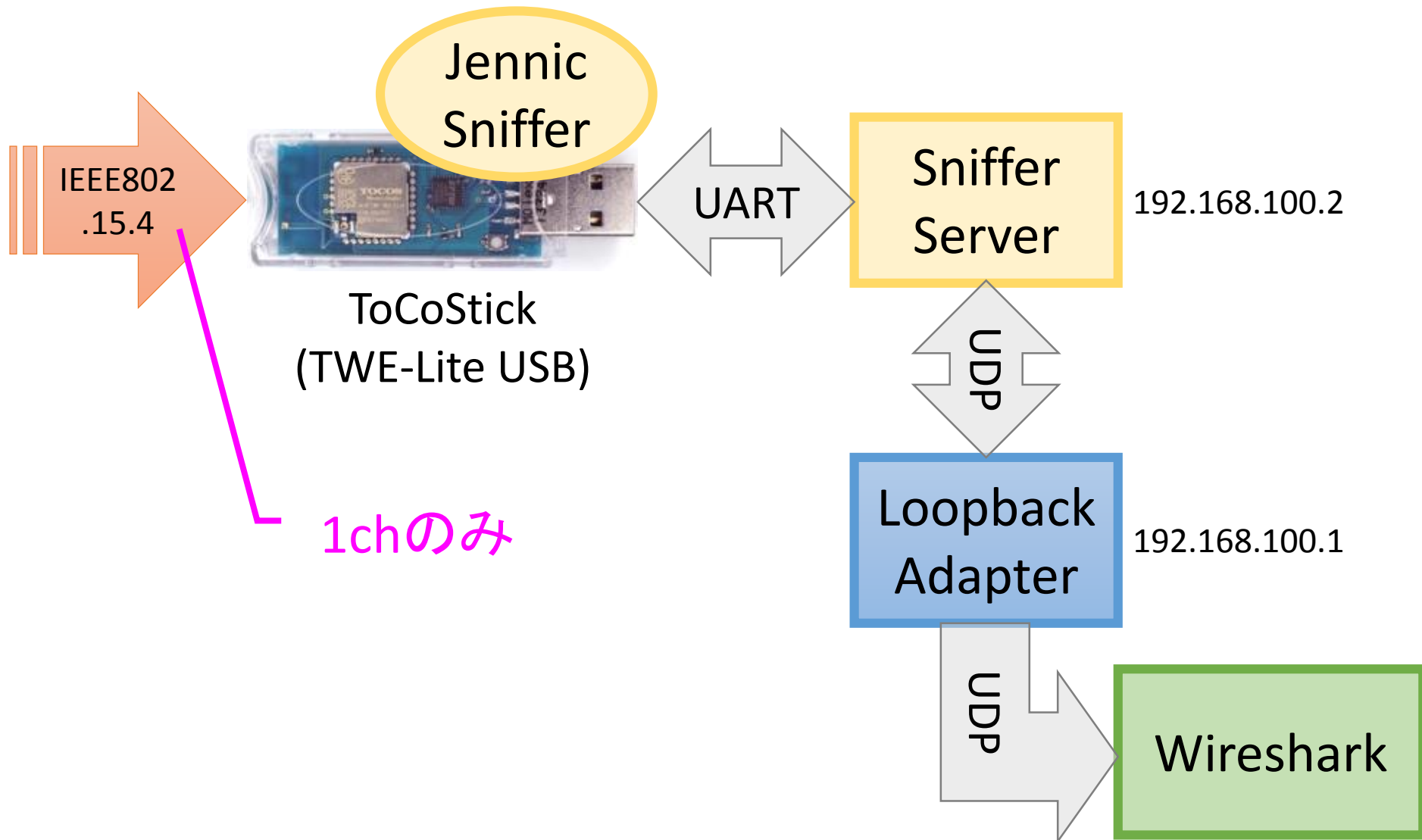


✓ RSSIと比較して、**ToF**は精度が高い。

パケットスニファ

パケットスニファ(パケットアナライザ)は**無線通信内容**を解析するツールで...
ツール上には、時系列に捕捉されたパケットが表示されるため、どの無線モジュールがどの順序で**電波**を出したかを記録できます。動作分析には必須のツールです。

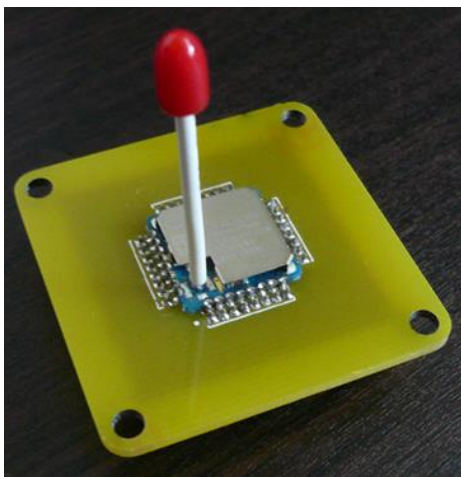
パケットスニファ



パケットスニファ



IEEE802.15.4



Sniffer...

Start Server

Stop Server

Exit

802.15.4 Channel

18

Sniffer ID

Destination IP Address

192.168.100.2

Destination UDP Port

49999

COM Port

☒ COM6

Baud Rate

1000000

Microsoft - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help

Filter: `ip.addr == 192.168.100.2`

No.	Time	Source	Destination	Protocol	Info
293	53.989897	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
295	54.980494	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
296	55.987013	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
297	56.977742	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
298	57.983828	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
299	58.975030	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
310	59.981409	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
311	60.987993	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
323	61.978713	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
324	63.975818	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
325	64.982663	00:1b:c5:01:21:00:49:75	Broadcast	IEEE 802.15.4	Data, Dst: Broadc

Internet Protocol, Src: 192.168.0.3 (192.168.0.3), Dst: 192.168.100.2 (192.168.100.2)

User Datagram Protocol, Src Port: 59839 (59839), Dst Port: 49999 (49999)

Jennic Sniffer Protocol

IEEE 802.15.4 Data, Dst: Broadcast, Src: IeeeReg1_01:21:00:49:75

Frame Control Field: Data (0xc801)

Sequence Number: 194

Destination PAN: 0x5678

Destination: 0xffff

Source PAN: 0x1234

Source: IeeeReg1_01:21:00:49:75 (00:1b:c5:01:21:00:49:75)

Data (5 bytes)

Data: 350000403d

[Length: 5]

0010 00 3b 63 7e 00 00 80 11 f1 dd c0 a8 00 03 c0 a8 :C~... ..4.0

0020 64 02 e9 bf c3 4f 00 27 7e 0e 00 10 e7 34 b0 30 d...o...~...4.0

0030 00 01 c8 c2 78 56 ff ff 34 12 75 49 00 21 01 c5 ...xv...4.uI...!

0040 1b 00 35 00 00 40 3d 19 de :5..@...!

Data (data.data), 5 bytes

Packets: 325 Displayed: 61 Marked: 0 Dropped: 0

Profile: Default

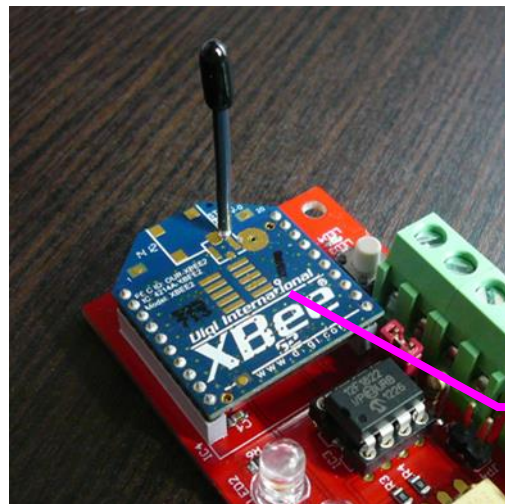
Sniffer
Server

Wireshark

パケットスニファ



IEEE802.15.4



Sniffe... - x

Start Server

Stop Server

Exit

802.15.4 Channel

11

Sniffer ID

Destination IP Address

192.168.100.2

Destination UDP Port

49999

COM Port

☒ COM6

Baud Rate

1000000

Microsoft - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help

Filter: `ip.addr == 192.168.100.2` Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
593	120.480256	0x0000	0x6f17	IEEE 802.15.4	Data, Dst: 0x6f17
594	120.481230			IEEE 802.15.4	Ack
595	121.468757	0x6f17	Broadcast	IEEE 802.15.4	Data, Dst: Broadc
596	121.565011	0x0000	0x6f17	IEEE 802.15.4	Data, Dst: 0x6f17
597	121.566069			IEEE 802.15.4	Ack
598	121.567153	0x6f17	0x0000	IEEE 802.15.4	Data, Dst: 0x0000
599	121.568124			IEEE 802.15.4	Ack
600	121.599608	0x6f17	0x0000	IEEE 802.15.4	Data, Dst: 0x0000
601	121.600670			IEEE 802.15.4	Ack
602	121.602156	0x0000	0x6f17	IEEE 802.15.4	Data, Dst: 0x6f17
603	121.603180			IEEE 802.15.4	Ack

Ethernet II, Src: 28:10:70:03:03:49 (28:10:70:03:03:49), Dst: 01:02:03:01:01:00 (01:02:03:01:01:00)

Internet Protocol, Src: 192.168.0.3 (192.168.0.3), Dst: 192.168.100.2 (192.168.100.2)

User Datagram Protocol, Src Port: 49912 (49912), Dst Port: 49999 (49999)

Jennic Sniffer Protocol

IEEE 802.15.4 Data, Dst: 0x0000, Src: 0x6f17

Data (50 bytes)

Data: 48180000176f1eca8322ac4000a21300401bac4000a21300...

Length: 50

0000 1c b1 7f bf 61 cc 28 18 78 c3 85 49 08 00 45 00 ...a.(.x..I..E.

0010 00 60 6c e5 00 00 80 11 e8 51 c0 a8 00 03 c0 a8 ...l.....Q.....

0020 64 02 c2 f8 c3 4f 00 4c 3b 41 00 0c ac a3 20 30 d.....O.L;A.....0

0030 00 61 88 7b 6d 98 00 00 17 6f 48 18 00 00 17 6f ..a.[m.....O.....0

0040 1e ca 83 22 ac 40 00 a2 13 00 40 1b ac 40 00 a2 ...@.@.....@...

0050 13 00 40 e8 11 00 05 c1 e8 0c 30 30 30 30 30 30 ..@.....00000000

0060 30 30 09 30 30 30 30 30 30 30 0a f2 1b 00.00000 0000...

Data (data.data), 50 bytes

Packets: 607 Displayed: 159 Marked: 0 Dropped: 0

Profile: Default

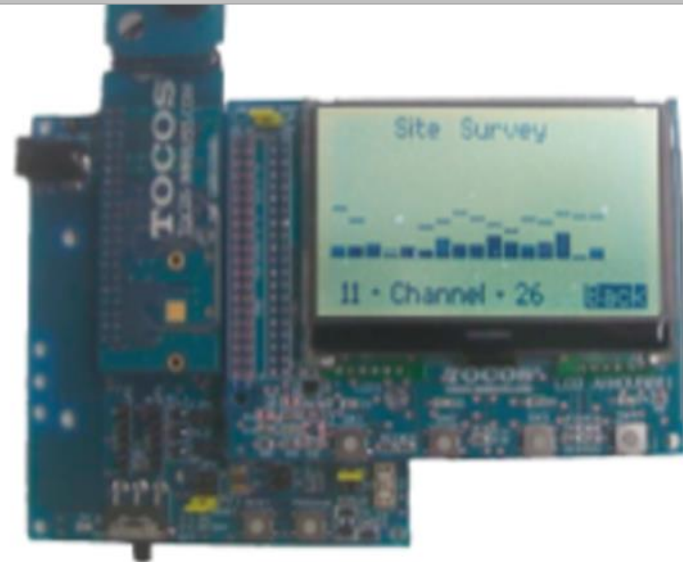
Digi XBeeもキャプチャできた

ネットワークディスカバリー

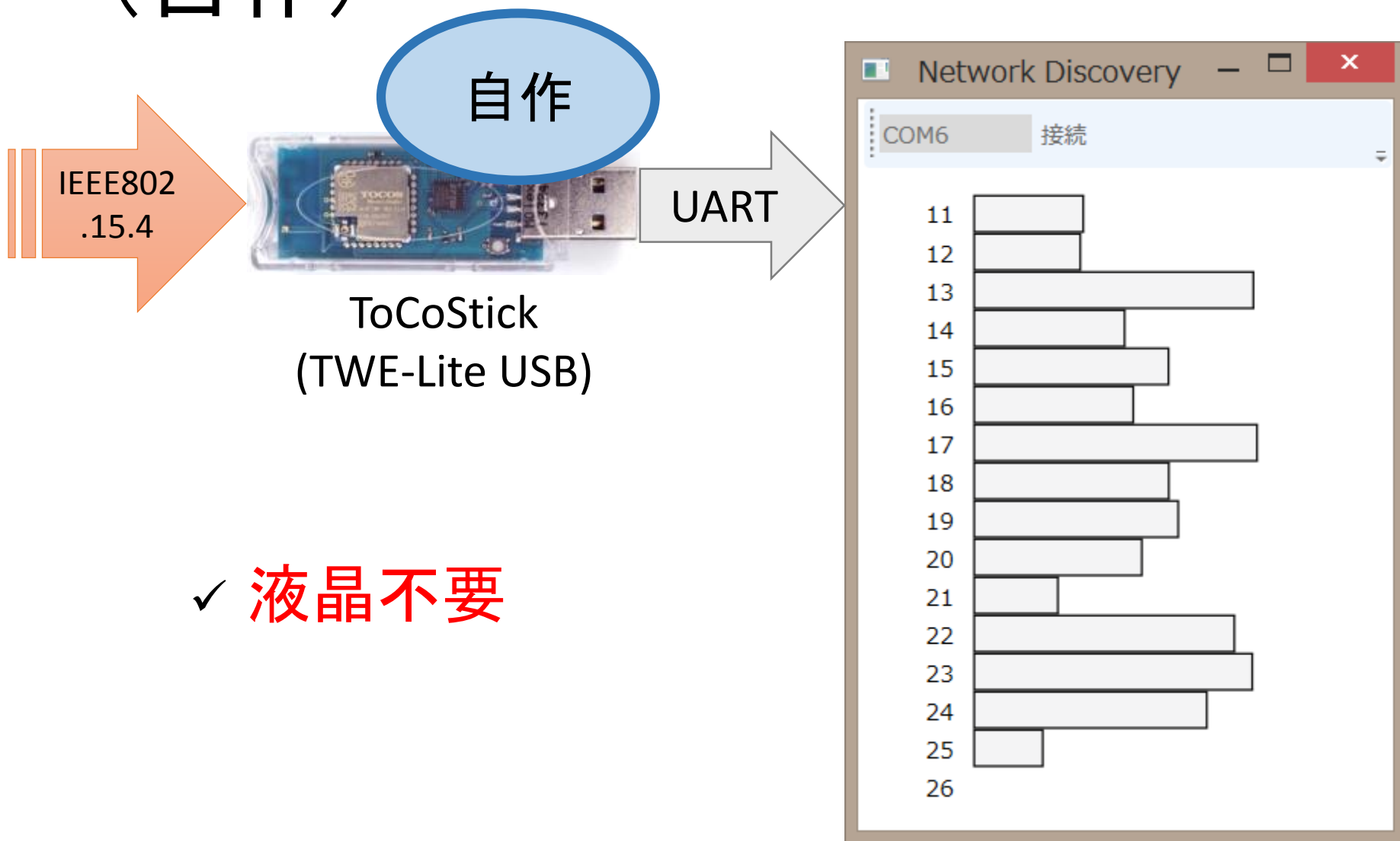
ネットワークディスカバリーツールは**電波環境** (Site Survey: 2.4GHz帯の各チャネルごとの簡易スペアナ) やIEEE802.15.4、JenNet、ZigBee PROネットワークの通信状況を観測するツールとして使用できます。

<http://tocos-wireless.com/jp/products/evalkit3.html>

- ✓ ネットワークディスカバリーツールはキットに付属の無線センサーノード (**液晶付き**) で動作します。



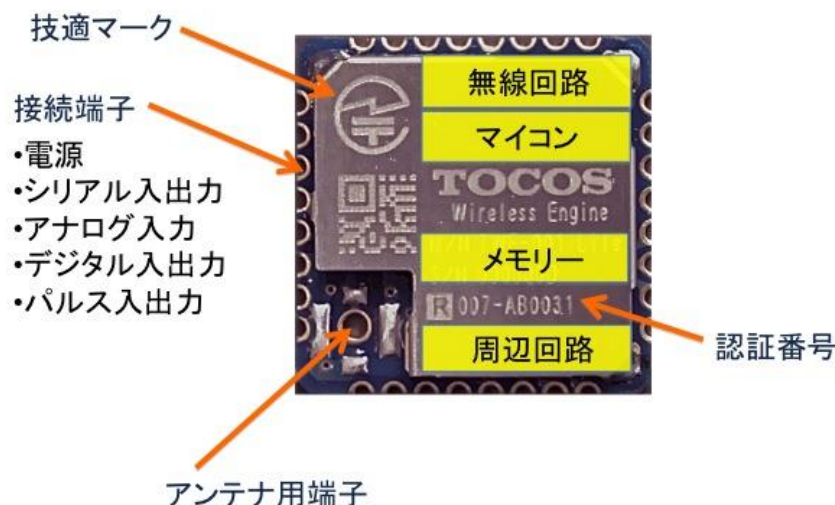
ネットワークディスカバリー (自作)



内蔵マイコンで制御可能

ワイヤレスエンジン TWE-Lite

TOCOS®



<http://tocos-wireless.com/jp/products/TWE-001Lite.html#TWE-001-9>

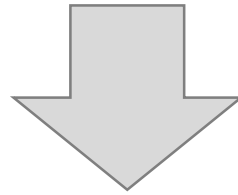
内蔵マイコンをプログラミングすることで、

1. **無線通信**の制御
 2. **I/O**の制御
- が可能！

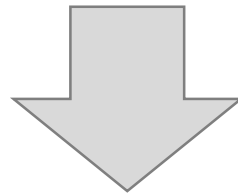
プログラミング作業



雛形となるプログラムをコピー



プログラムを改修



TWE-Liteに書き込み

雛形となるプログラム

✓ TWE-Zeroアプリ

- 超簡単！TWEアプリ
- シリアル通信専用アプリ
- 無線タグ
- リモコン通信専用アプリ
- オーディオ・信号通信専用アプリ

(App_TweLite)
(App_Uart)
(Samp_Monitor)
(App_IO)
(App_Audio)

✓ (分類不明)

- メロディアプリケーション
- 連続的にパケットを送るサンプル
- I2C サンプル
- パケットエラー測定ツール
- PingPong サンプル
- 上り下り送信を行う子機のデモ

(App_Melody)
(Samp_ContTx)
(Samp_I2C)
(Samp_PER)
(Samp_PingPong)
(Samp_Wayback)

TWE-Liteに書き込み

書込手順A



パソコン



TWE-Lite R



TWE-Lite DIP

ToCoNet SDK

Eclipse

TWE-Programmer

書込手順B



TTL-232R-3V3

ちょっと
した
スイッチ

ちょっと
した
配線



TWE-Lite

書込手順A

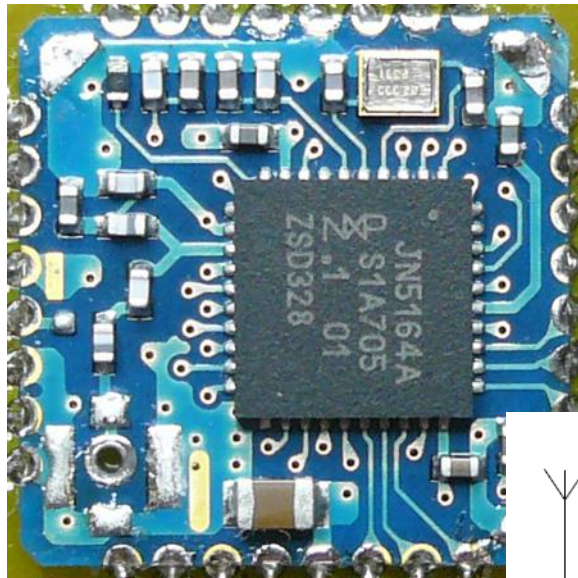


ToCoStick

書込手順A: .binをD&Dするだけ

書込手順B: スイッチをゴニョしてから.binをD&D

TWE-Liteの中身



← JN5164A+XTAL+α

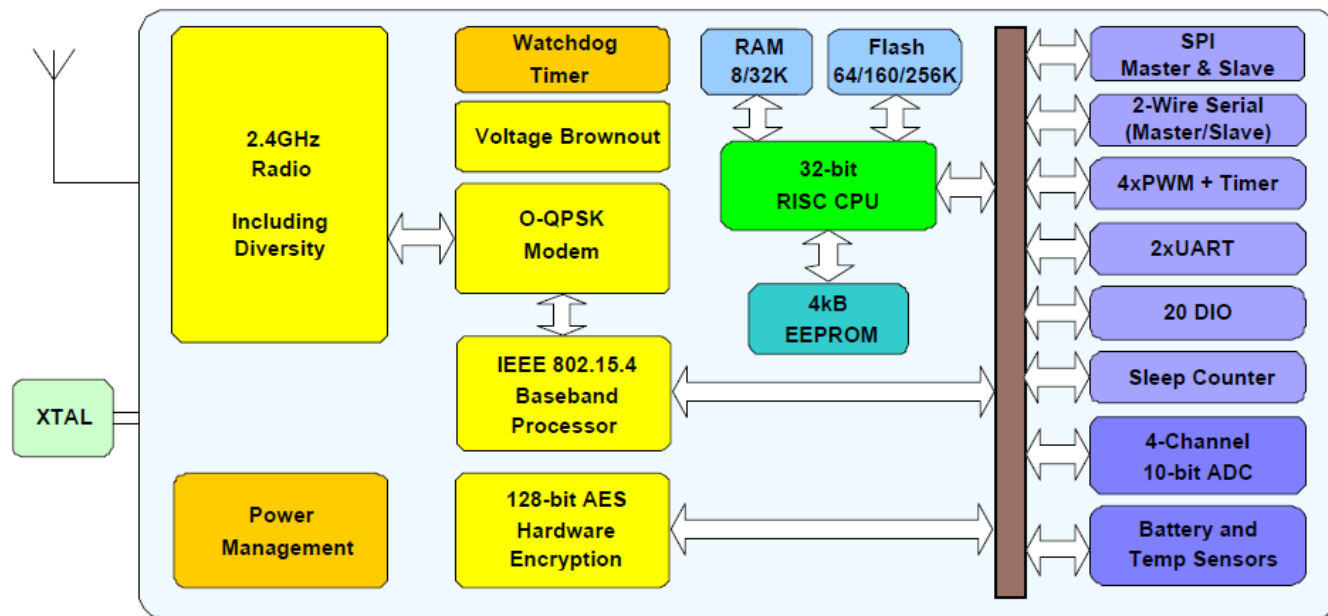
↓ JN516x Block Diagram

JN5164 ...

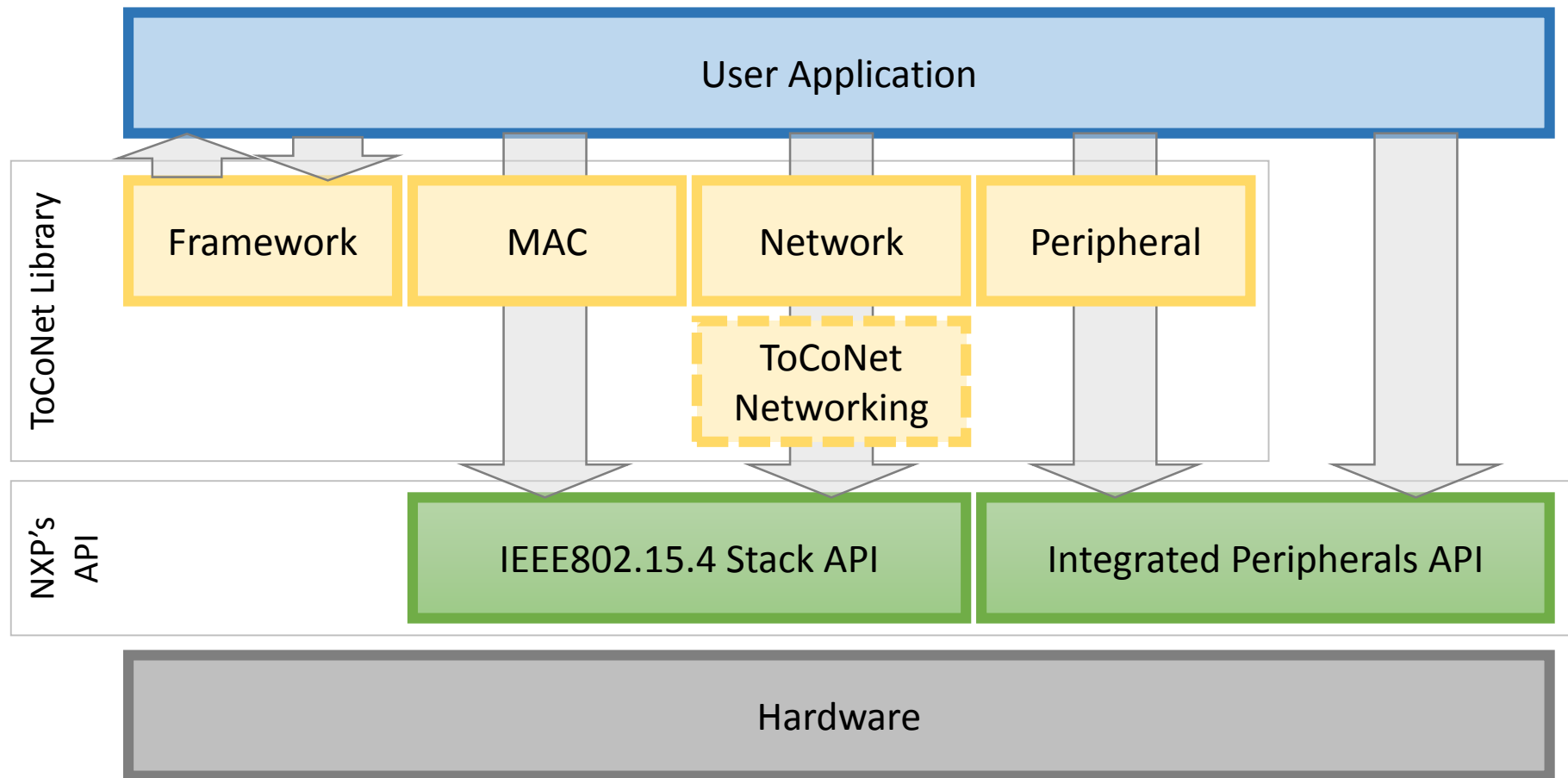
Flash 160kB

RAM 32kB

EEPROM 4kB

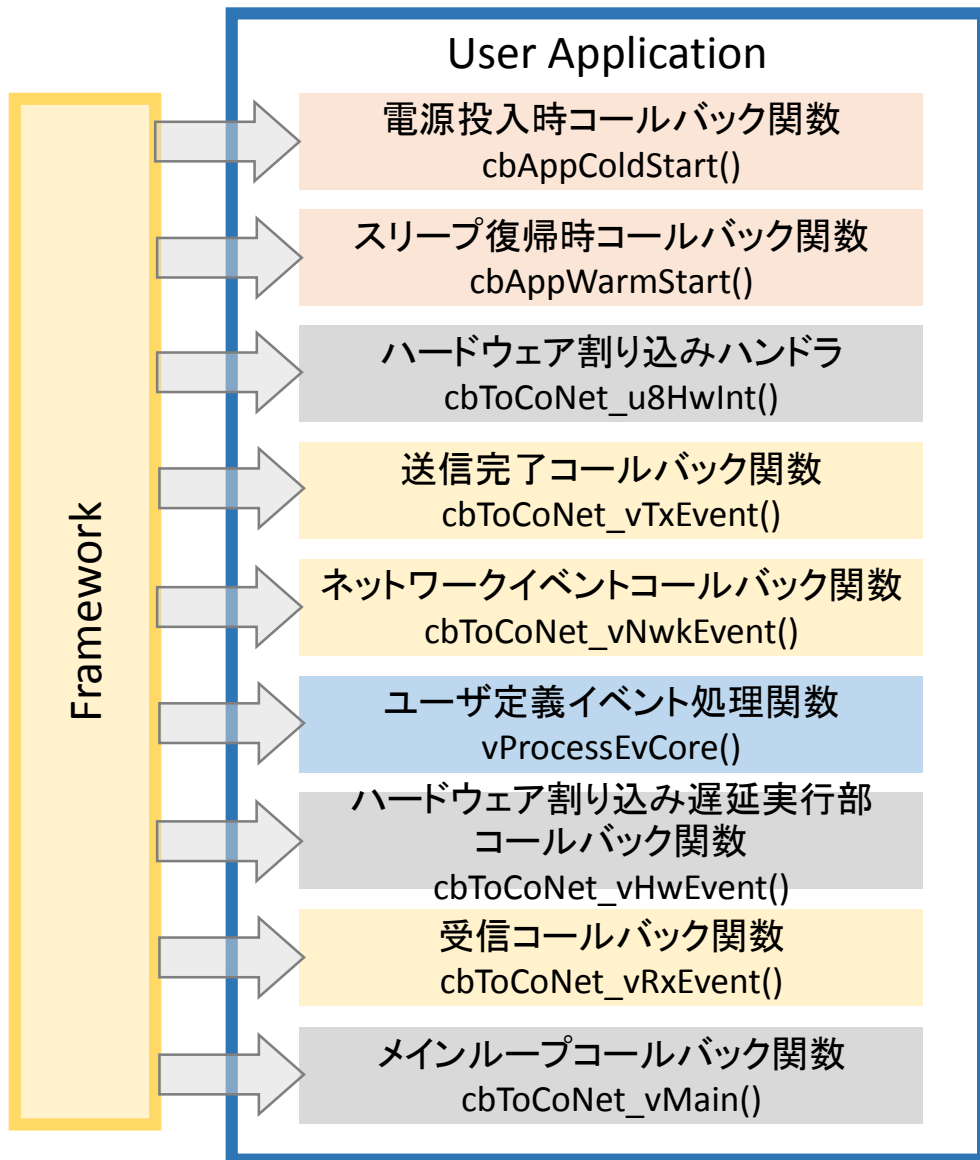


Software Architecture



ToCoNet Framework

Framework



- ✓ イベント駆動型。
- ✓ 予めコールバックする関数名が定義されている。
- ✓ 処理は、状態遷移でユーザー定義イベント処理関数に実装する。

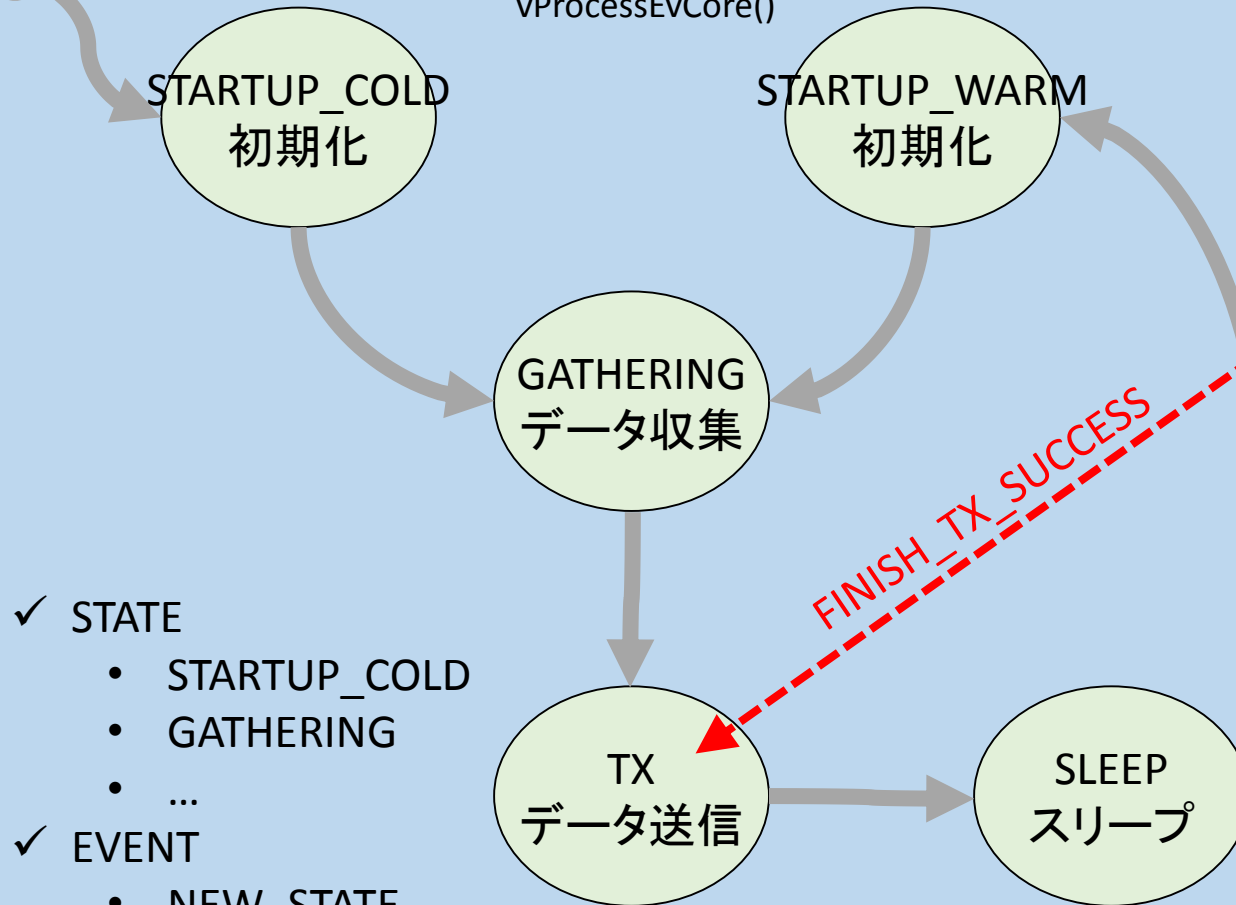
- ✓ 電源ON
 - cdAppColdStart
 - cbAppWarmStart
- ✓ イベントハンドラ
 - vProcessEvCore
- ✓ 送信完了
 - cbToCoNet_vTxEvent
- ✓ 受信
 - cbToCoNet_vRxEvent

ToCoNet Framework

Framework

ユーザ定義イベント処理関数
vProcessEvCore()

送信完了コールバック関数
cbToCoNet_vTxEvent()



✓ STATE

- STARTUP_COLD
- GATHERING
- ...

✓ EVENT

- NEW_STATE
- FINISH_TX_SUCCESS
- ...

ToCoNet Framework

Framework

状態を定義

```
typedef enum
{
    > E_STATE_APP_BASE = ToCoNet_STATE_APP_BASE,
    > E_STATE_APP_STARTUP_GOLD,
    > E_STATE_APP_STARTUP_WARM,
    > E_STATE_APP_GATHERING,
    > E_STATE_APP_TX,
    > E_STATE_APP_SLEEP,
} teStateApp;
```

ユーザー定義イベント処理関数を実装

```
static const tsToCoNet_Event_StateHandler asStateFuncTbl[] =
{
    > PRSEV_HANDLER_TBL_DEF(E_STATE_IDLE),
    > PRSEV_HANDLER_TBL_DEF(E_STATE_APP_STARTUP_GOLD),
    > PRSEV_HANDLER_TBL_DEF(E_STATE_APP_STARTUP_WARM),
    > PRSEV_HANDLER_TBL_DEF(E_STATE_APP_GATHERING),
    > PRSEV_HANDLER_TBL_DEF(E_STATE_APP_TX),
    > PRSEV_HANDLER_TBL_DEF(E_STATE_APP_SLEEP),
};

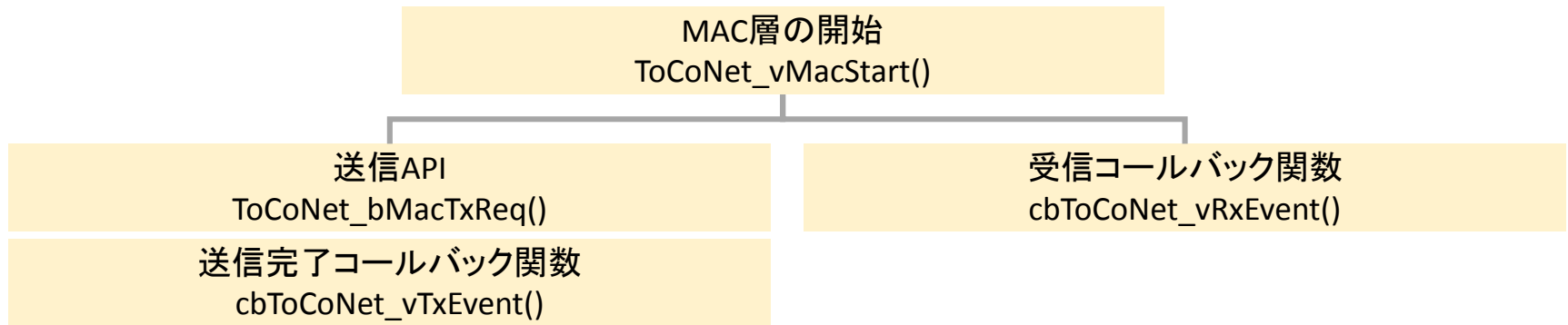
static void vProcessEvCore(tsEvent *pEv, teEvent eEvent, uint32
{
    > ToCoNet_Event_StateExec(asStateFuncTbl, pEv, eEvent, u32eva
}
```

状態毎の処理を実装

```
> PRSEV_HANDLER_DEF(E_STATE_APP_TX, tsEvent *pEv, teEvent
{
    > static uint8 sequence = 0;
    > if (eEvent == E_EVENT_NEW_STATE)
    {
        > > vfprintf(&uart, "HE_STATE_APP_TX"LB);
        > > tsTxDataApp tsTx;
        > > zero_memory(&tsTx, sizeof (tsTx));
        > > tsTx.u8CbId = 0;
        > > //tsTx.u8Retry = 0x80;
        > > tsTx.u8Cmd = TOCONET_PACKET_CMD_APP_DATA;
        > > //tsTx.bAckReq = FALSE;
        > > tsTx.u8Seq = sequence++;
        > > tsTx.u32DstAddr = TOCONET_NWK_ADDR_PARENT;
        > > tsTx.u32SrcAddr = ToCoNet_u32GetSerial();
        > > tsTx.u8Len = 4;
        > > tsTx.auData[0] = U16_LOWER_U8(ModuleTemperature);
        > > tsTx.auData[1] = U16_UPPER_U8(ModuleTemperature);
        > > tsTx.auData[2] = U16_LOWER_U8(ModuleVolt);
        > > tsTx.auData[3] = U16_UPPER_U8(ModuleVolt);
        > > if (!ToCoNet_Nwk_bTx(NwkContext, &tsTx))
        {
            > > vfprintf(&uart, "!Tx."LB);
            > > Reset();
        }
        > > ToCoNet_Tx_vProcessQueue();
    }
    > else if (eEvent == E_EVENT_APP_FINISH_TX_SUCCESS)
```


ToCoNet Wireless

MAC



- ✓ Media Access Controlの略。
- ✓ 1対1の送受信。

ToCoNet Wireless

Network



ToCoNet Peripheral

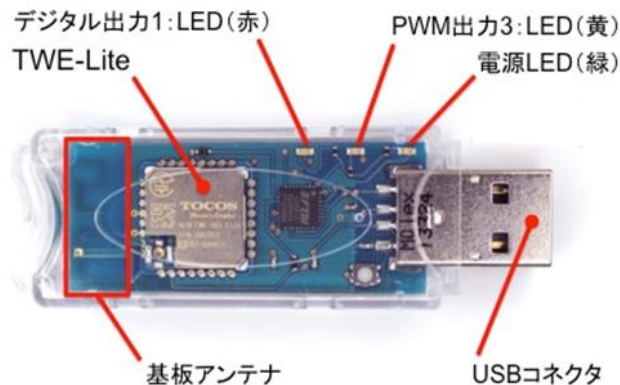
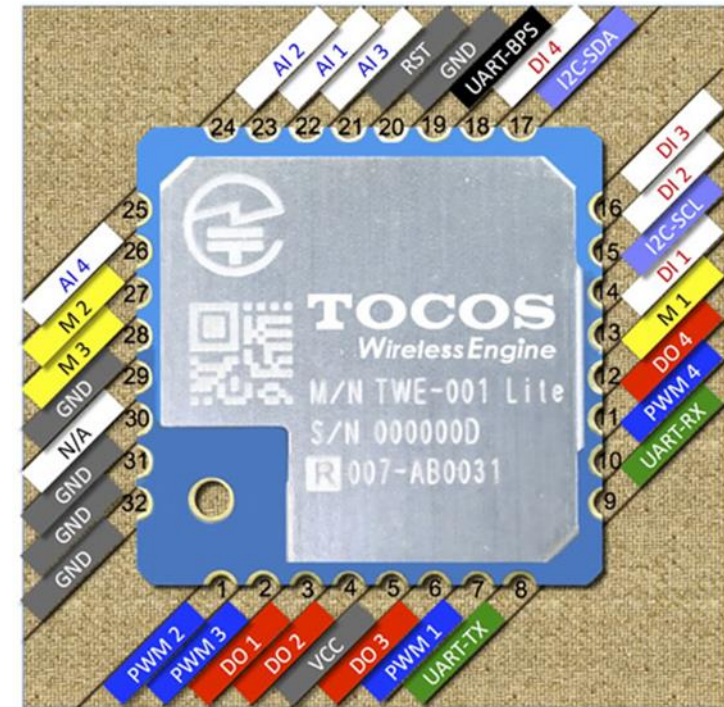
Peripheral

Peripheral	ToCoNet Library	Integrated Peripherals API
SPI		<code>xAHl_SpiXxx()</code>
2-Wire Serial		<code>xAHl_SiXxx()</code>
PWM/Timer		<code>xAHl_TimerXxx()</code>
UART	<code>serial.h</code> <code>serialInputMgr.h</code>	<code>xAHl_UartXxx()</code>
DIO	<code>utils.h</code> <code>btnMgr.h</code>	<code>xAHl_DioXxx()</code> <code>xAHl_DoXxx()</code>
ADC		<code>xAHl_AdcXxx()</code>
EEPROM	<code>eeeprom_6x.h</code>	<code>xAHl_XxxEEPROMxxx()</code>

ToCoNet Peripheral

Peripheral

機能	信号名	シルク	ピン	ピン	シルク	信号名	機能
電源グラウンド	GND	GND	1	28	VCC	VCC	電源(2.3~3.6V)
I2Cクロック	SCL	14	2	27	3	M3	モード設定ビット3
UART受信	RX	7	3	26	2	M2	モード設定ビット2
PWM出力1	PWM1	5	4	25	1	AI4	アナログ入力4
デジタル出力1	DO1	18	5	24	A2	AI3	アナログ入力3
PWM出力2	PWM2	C	6	23	0	A2	アナログ入力2
PWM出力3	PWM3	1	7	22	A1	AI1	アナログ入力1
デジタル出力2	DO2	19	8	21	R	RST	リセット入力
デジタル出力3	DO3	4	9	20	17	BPS	UART速度設定
UART送信	TX	6	10	19	15	SDA	I2Cデータ
PWM出力4	PWM4	8	11	18	16	DI4	デジタル入力4
デジタル出力4	DO4	9	12	17	11	DI3	デジタル入力3
モード設定ビット1	M1	10	13	16	13	DI2	デジタル入力2
電源グラウンド	GND	GND	14	15	12	DI1	デジタル入力1



製品によってピン配置が**違う**！
十分注意が必要。

ToCoNet Peripheral

Peripheral

jenprog	TWE-Lite DIP TWE-001L-DPC-WA			TWE-Lite TWE-001L-NC		ToCoStick TWE-Lite-USB	JN5164A						
Name	Pin#	Silk	Function	Pin#	Name		Pin#	Name	Alt1	Alt2	Alt3	Alt4	Alt5
RXD	1	GND	GND	20,28	GND								
	2	14	SCL	14	SCL		38	DIO14	SIF_CLK	TXD0 TXD1	JTAG_TDO	SPISEL1	SPISEL
	3	7	RX	9	RX	FT233RQ[TXD]	29	DIO7	RXD0	JTAG_TDI	PWM3		
	4	5	PWM1	7	PWM1	FT233RQ[CTS]	27	DIO5	RTS0	JTAG_TMS	PWM1	PC1	
PRG	5	18	DO1	3	DO1		23	DIO18	SPIMOSI				
	6	C	PWM2	1	PWM2		20	DO0	SPICLK			PWM2	
	7	I	PWM3	2	PWM3	FT233RQ[CBUS3]	22	DO1	SPIMISO			PWM3	
	8	19	DO2	4	DO2		24	DIO19	SPISEL0				
TXD	9	4	DO3	6	DO3	FT233RQ[RTS]	26	DIO4	CTS0	JTAG_TCK	TIM0OUT	PC0	
	10	6	TX	8	TX	FT233RQ[RXD]	28	DIO6	TXD0	JTAG_TDO	PWM2		
	11	8	PWM4	10	PWM4		31	DIO8	TIM0CK_GT	PC1	PWM4		
	12	9	DO4	11	DO4		32	DIO9	TIM0CAP	32KXTALIN	RXD1	32KIN	
	13	10	M1	12	M1		33	DIO10	TIM0OUT	32KXTALOUT			
	14	GND	GND	20,28	GND								
	15	12	DI1	13	DI1		36	DIO12	PWM2	CTS0	JTAG_TCK	ADO	SPISMOSI
	16	13	DI2	15	DI2		37	DIO13	PWM3	RTS0	JTAG_TMS	ADE	SPISMISO
	17	11	DI3	16	DI3		34	DIO11	PWM1		TXD1		
	18	16	DI4	18	DI4	LED:赤	1	DIO16	COMP1P	SIF_CLK	SPISMOSI		
	19	15	SDA	17	SDA		40	DIO15	SIF_D	RXD0 RXD1	JTAG_TDI	SPISEL2	SPISEL
RST	20	17	BPS	19	BPS		2	DIO17	COMP1M	SIF_D	SPISMISO		
	21	R	RST	21	RST	FT232RQ[CBUS2]	3	RESETN					
	22	A1	AI1	23	AI1		15	ADC1					
	23	0	AI2	24	AI2		16	DIO0	SPISEL1	ADC3			
	24	A2	AI3	22	AI3		11	ADC2					
	25	1	AI4	25	AI4		17	DIO1	SPISEL2	ADC4	PC0		
	26	2	M2	26	M2		18	DIO2		RFRX	TIM0CK_GT		
	27	3	M3	27	M3		19	DIO3		RFTX	TIM0CAP		
	28	VCC	VCC	5	VCC								

参照すべきドキュメント

Framework

Network

MAC

Peripheral



ToCoNet SDK マニュアル

ToCoNet_SDK_manual_201406.pdf

IEEE802.15.4 Stack API

Integrated Peripherals API



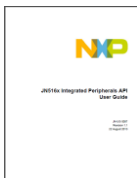
JN516x Integrated Peripherals API User Guide

JN-UG-3087-JN516x-Integrated-Peripherals-API_1v1.pdf



データシート TWE-Lite

TWD-PDS-TWE001L-JP-1.30.pdf



Data Sheet: JN516x
JN-DS-JN516x-1v3.pdf

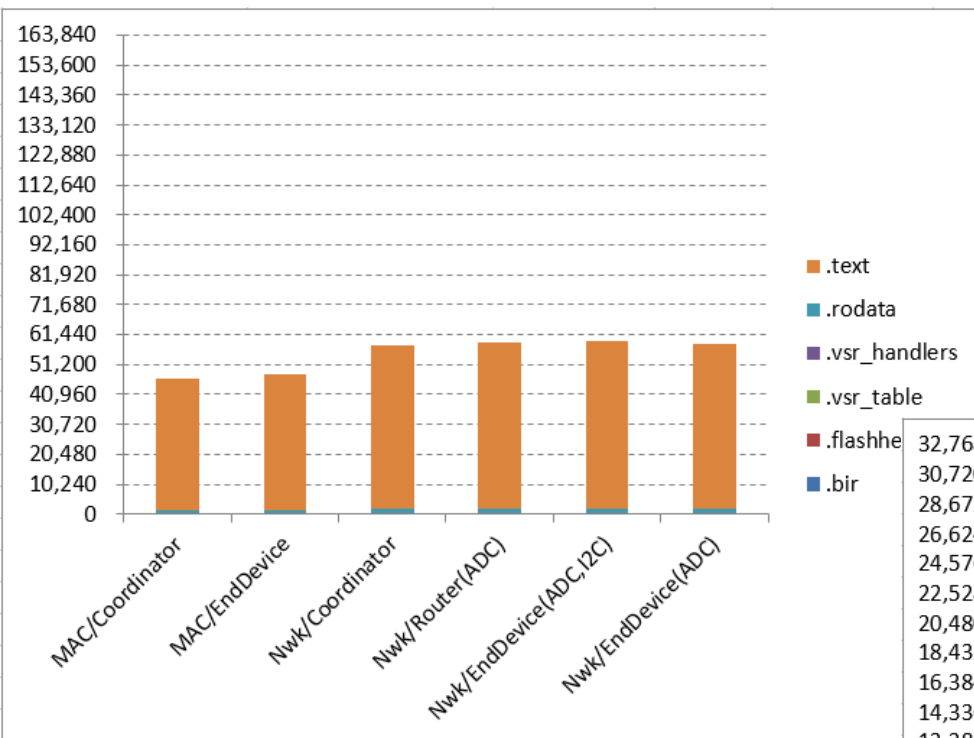


IEEE802.15.4 Stack User Guide

http://www.nxp.com/documents/user_manual/JN-UG-3024.pdf

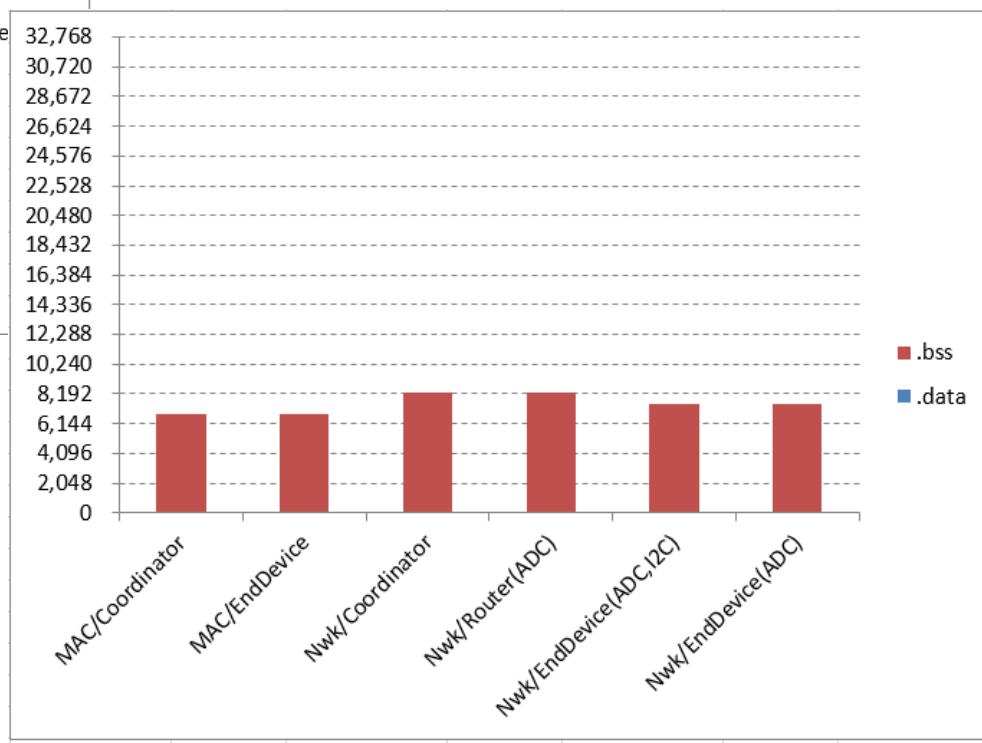
参考

Memory Footprint



↓ RAM
8K used

↑ Flash ROM
60K used



まとめ

- ✓ ボタン電池による運用ができる。
- ✓ 距離を飛ばすのは難しい。
- ✓ 内蔵マイコンのプログラミングはコツがいる。
 - フレームワーク
 - ピン配置
 - 参照ドキュメント

おしまい