

### 【Enddevice と Coordinator 間での文字列送受信を 5 秒間隔に実装】

送信間隔を調整できるようになれば、複数の Enddevice からのデータ受信や短い間隔での Coordinator の受信が実装でき、輻輳状況を評価できる。

sleep 関数を使用できそうだが、コンパイルできなかったため、Enddevice のスリープ機能を活用した。そうすることで、現在 5 秒間隔で文字列を送信できるように実装できた。

スリープ機能とは、Enddevice が特定の時間間隔で Coordinator に送信するデータの有無を確認する ZigBee 特有の機能である。例えば、5 秒間隔であれば、1 ～ 4 秒間はスリープ状態でプログラムが停止し、5 秒になると、Coordinator にデータの確認をする。

```

*****
PWRM_CALLBACK(PreSleep)
{
    DBG_vPrintf(TRACE_APP, "APP: Going to sleep (CB)\n");
    vAppApiSaveMacSettings();
    ZTIMER_vSleep();
}

APP_vSetUpHardware();
ZTIMER_vWake();
SendData(); //文字列送信コード

```

このように ZTIMER\_vSleep（寝ている状態）と ZTIMER\_vWake（起きている状態）の間に、文字列データを送信する SendData 関数を入れた。

今までは、スリープ状態が一定間隔あると、不便であったため、コメントアウトをしていた。またスリープ機能をすると、Coordinator → Enddevice の送信もできるようになる。

## 【複数台の Enddevice からの文字列受信するためのコード整理】

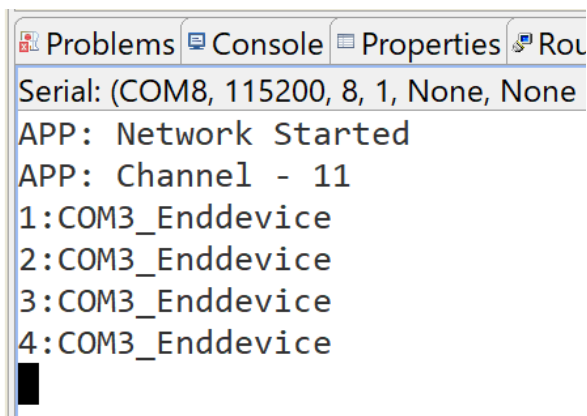
(Enddevice コード)

```
char buffer[16];
buffer[0] = count; // 配列の0番目の要素にcountの値を代入
count++;
strcpy(buffer + 1, "COM3_Enddevice"); // 残りのバッファを 指定の文字列 で初期化
u16Offset += PDUM_u16APduInstanceWriteNBO(hAPduInst, u16Offset, "a\x10", buffer);
DBG_vPrintf	TRACE_APP, "\n");
```

(Coordinator のコード)

```
//文字列実験用コード
char Stringdata[256];
u16bytesread = PDUM_u16APduInstanceReadNBO(sStackEvent.uEvent.sApsDataIndEvent.hAPduInst,0,"a\x10",&Stringdata);
DBG_vPrintf	TRACE_APP, "%d:", Stringdata[0]);
for(i = 1; i < u16bytesread; i++){
    DBG_vPrintf	TRACE_APP, "%c", Stringdata[i]);
}
DBG_vPrintf	TRACE_APP, "\n");
```

(Coordinator での表示結果)



```
Serial: (COM8, 115200, 8, 1, None, None)
APP: Network Started
APP: Channel - 11
1:COM3_Enddevice
2:COM3_Enddevice
3:COM3_Enddevice
4:COM3_Enddevice
```

送信カウント数と、どの Enddevice からのデータなのかを COM で表し、表示させるようにした。

## 【Enddevice を 5 台用意し (COM3, 4, 7, 9, 10), 以下の秒数ごとに受信結果を調査】

(1 秒ごとに各 Enddevice から送信)

結果は,

- COM3\_Enddevice: 57 個 (漏れなし)
- COM4\_Enddevice: 39 個 (漏れなし)
- COM7\_Enddevice: 59 個 (漏れなし)
- COM9\_Enddevice: 51 個 (漏れなし)
- COM10\_Enddevice: 50 個 (漏れなし)

```

APP: Channel - 11
1:COM7_Enddevice
2:COM7_Enddevice
1:COM3_Enddevice
3:COM7_Enddevice
2:COM3_Enddevice
4:COM7_Enddevice
3:COM3_Enddevice
5:COM7_Enddevice
4:COM3_Enddevice
6:COM7_Enddevice
1:COM9_Enddevice
5:COM3_Enddevice
7:COM7_Enddevice
2:COM9_Enddevice
6:COM3_Enddevice
8:COM7_Enddevice
3:COM9_Enddevice
7:COM3_Enddevice
9:COM7_Enddevice
4:COM9_Enddevice
8:COM3_Enddevice
1:COM10_Enddevice
10:COM7_Enddevice
5:COM9_Enddevice
9:COM3_Enddevice
2:COM10_Enddevice
11:COM7_Enddevice
6:COM9_Enddevice
10:COM3_Enddevice
3:COM10_Enddevice
12:COM7_Enddevice
7:COM9_Enddevice
11:COM3_Enddevice
4:COM10_Enddevice
Serial: (COM8, 115200, 8, 1)
43:COM3_Enddevice
45:COM7_Enddevice
40:COM9_Enddevice
33:COM4_Enddevice
37:COM10_Enddevice
44:COM3_Enddevice
46:COM7_Enddevice
41:COM9_Enddevice
34:COM4_Enddevice
38:COM10_Enddevice
47:COM7_Enddevice
45:COM3_Enddevice
42:COM9_Enddevice
35:COM4_Enddevice
39:COM10_Enddevice
48:COM7_Enddevice
46:COM3_Enddevice
43:COM9_Enddevice
36:COM4_Enddevice
40:COM10_Enddevice
49:COM7_Enddevice
47:COM3_Enddevice
44:COM9_Enddevice
37:COM4_Enddevice
41:COM10_Enddevice
50:COM7_Enddevice
45:COM9_Enddevice
48:COM3_Enddevice
38:COM4_Enddevice
42:COM10_Enddevice
51:COM7_Enddevice
46:COM9_Enddevice
49:COM3_Enddevice
39:COM4_Enddevice
47:COM10_Enddevice
51:COM9_Enddevice

```

(0.1 秒ごとに各 Enddevice から送信)

- ・ COM3\_Enddevice: 漏れなし 数値範囲 : 61, 62, 63, ..., 127, -128, -127, ..., -2, -1
- ・ COM4\_Enddevice: 漏れなし 数値範囲 : 52, 53, 54, ..., 127, -128, -127, ..., -2, -1
- ・ COM7\_Enddevice: 漏れなし 数値範囲 : 20, 21, 22, ..., 127, -128, -127, ..., -2, -1
- ・ COM9\_Enddevice: 漏れなし 数値範囲 : 38, 39, 40, ..., 127, -128, -127, ..., -2, -1
- ・ COM10\_Enddevice: 漏れなし 数値範囲 : 16, 17, 18, ..., 127, -128, -127, ..., -2, -1

数値は連続していたが、開始の数値やマイナス数値がおかしい。

理由はカウントする変数 count が符号付きの char 型であるから。そのため、

Enddevice 側では、

char count → unsigned char count

char buffer[16] → unsigned char buffer[16]; に変更した。

Coordinator 側では、

char Stringdata[256]; → unsigned char Stringdata[256]; に変更した。

char 型には, long 修飾子がない。

(0.01 秒ごとに各 Enddevice から送信)

- COM3\_Enddevice: 247 個 数値範囲 33～247
- COM4\_Enddevice: 255 個 数値範囲 7～255
- COM7\_Enddevice: 196 個 数値範囲 1～196
- COM9\_Enddevice: 66 個 数値範囲 1～66
- COM10\_Enddevice: 56 個 数値範囲 1～56

各 COM の数値をみると、連続はしているが、特定の数値範囲が漏れている。

各デバイスが電源に差し込まれるタイミングが異なるが、極端に短い計測ではないため、COM10 など不自然。

0.01s 間隔の送受信を再度計測し直し、デバイス数の影響を調べる。