TB3239

PIC®とAVR®デバイス用JSON復号部

序説

著者: Alexandru Niculae, Microchip Technology Inc.

この資料はPIC®とAVR®マイクロ コントローラのような組み込みデバイスを狙いとしたJSON(JavaScript Object Notation)制限版復号部の使い方と実装を記述します。

復号部はJSONオブジェクトの文字列形式をCデータ構造体表現に変換します。このようにしてプログラム作成者は符牒(Key)と値(Value)の対にアクセスすることができます。JSONオブジェクトの使用は応用の相互連絡を容易にします。

JSON復号部コードはGitHubで入手可能です。

GitHubでコード例を見てください。
貯蔵庫を閲覧するにはクリックしてください。

目次

1. 概要

JSONはｳｪﾌﾞと机上応用で最も使われるﾃﾞｰﾀ直列化形式です。これは事実上全ての最新のﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語で実装されるため、応用間のﾃﾞｰﾀ交換に最適な道具です。これは人にとって読み易く、機械にとって解析と生成が容易な文字形式を持ちます。

JSONｵﾌﾞｼﾞｪｸﾄは符牒(Key)と値(Value)の対の組です。符牒は文字列にだけでき、値は色々にできます。この復号部実装は文字列、数値、ｵﾌﾞｼﾞｪｸﾄ値を支援します。

注: このﾗｲﾌﾞﾗﾘを少ﾒﾓﾘ ﾃﾞﾊﾞｲｽに適合させておくため、実装に対して機能の一部分だけが選ばれ、従って、配列、論理型(ﾌﾞｰﾘｱﾝ)、ﾇﾙ(null)値は支援されません。

以下のｺｰﾄﾞの塊はこの復号部の使い方の例を示します。2. API章はAPI関数を詳細に記述します。

```
#define TEST_JSON "{¥"main¥":{¥"key¥" : 10,¥"foo¥":¥"bar¥"}, ¥"alt¥":2}"

～

jsonNode_t *root, *objmain;
char str[64], foo[10];
int alt;

memcpy(str, TEST_JSON, sizeof(TEST_JSON));

JSON_DECODER_fromString(str)

JSON_DECODER_getRoot(&root);
JSON_DECODER_getObject(root, "main", &objmain);

JSON_DECODER_getNumber(root, "alt", &alt)
JSON_DECODER_getString(objmain, "foo", sizeof(foo), foo)
```

注: JSON内の最も外側のｵﾌﾞｼﾞｪｸﾄがﾙｰﾄ(root)と呼ばれます。

2. API

JSON復号部の公開関数の詳細な説明についてはこの章を参照してください。

```
jsonDecoderStatus_t JSON_DECODER_fromString(char *str)
```

説明

JSON文字列をC表現に解析。文字列、数値、ｵﾌﾞｼﾞｪｸﾄ値を支援します。

ﾊﾟﾗﾒｰﾀ

名前	説明
str	JSONｵﾌﾞｼﾞｪｸﾄの既存文字列表現への位置指示子。解析後、文字列は変更されます。

戻り値

名前	説明
JSON_DECODER_OK	復号は成功しました。
JSON_DECODER_BAD_FORMAT	入力が無効です。

```
JSON_DECODER_getRoot(jsonNode_t **pNode)
```

説明

ﾙｰﾄと呼ばれるJSONの最も外側のｵﾌﾞｼﾞｪｸﾄを探します。この関数はJSON_DECODER_fromString前に呼ばれてはなりません。

ﾊﾟﾗﾒｰﾀ

名前	説明
pNode	jsonNode_tへの位置指示子。

戻り値

名前	説明
JSON_DECODER_OK	pNodeは今やﾙｰﾄを指示します。

```
jsonDecoderStatus_t JSON_DECODER_getObject(jsonNode_t *current, char *key, jsonNode_t **pNode)
```

説明

別のJSONｵﾌﾞｼﾞｪｸﾄで符牒によってJSDNｵﾌﾞｼﾞｪｸﾄを探します。この関数はJSON_DECODER_fromString前に呼ばれてはなりません。

ﾊﾟﾗﾒｰﾀ

名前	説明
current	jsonNode_tへの位置指示子。
key	見つけるｵﾌﾞｼﾞｪｸﾄの符牒。
pNode	見つかった場合、そのｵﾌﾞｼﾞｪｸﾄへの位置指示子。

戻り値

名前	説明
JSON_DECODER_OK	ｵﾌﾞｼﾞｪｸﾄが見つかり、今やpNodeはそれを指示します。
JSON_DECODER_KEY_NOT_FOUND	指定した符牒が存在しません。

```
jsonDecoderStatus_t JSON_DECODER_getString(jsonNode_t *current, char *key, uint8_t size, char *pVal)
```

説明

JSONｵﾌﾞｼﾞｪｸﾄで符牒によって文字列を探します。この関数はJSON_DECODER_fromString前に呼ばれてはなりません。

ﾊﾟﾗﾒｰﾀ

名前	説明
current	検索するJSONｵﾌﾞｼﾞｪｸﾄ。
key	探す文字列の符牒。
size	文字列の最大の大きさ。pVal緩衝部の長さより大きくてはなりません。
pVal	見つかった場合、その文字列への位置指示子。

戻り値

名前	説明
JSON_DECODER_OK	文字列が見つかり、今やpNodeはそれを指示します。
JSON_DECODER_KEY_NOT_FOUND	指定した符牒が存在しません。

```
jsonDecoderStatus_t JSON_DECODER_getNumber(jsonNode_t *current, char *key, int *pVal)
```

説明

JSONｵﾌﾞｼﾞｪｸﾄで符牒によって数値を探します。この関数はJSON_DECODER_fromString前に呼ばれてはなりません。

ﾊﾟﾗﾒｰﾀ

名前	説明
current	検索するJSONｵﾌﾞｼﾞｪｸﾄ。
key	探す数値の符牒。
pVal	見つかった場合、その数値への位置指示子。

戻り値

名前	説明
JSON_DECODER_OK	数値が見つかり、今やpNodeはそれを指示します。
JSON_DECODER_KEY_NOT_FOUND	指定した符牒が存在しません。

3. 実装

本章はｺｰﾄﾞのより深い理解を促進する目的で実装の詳細を記述します。更に、ﾒﾓﾘ使用と実装安全性を最適化させるいくつかの判断を記述します。例えば、再帰の使用が推奨されないので、算法の一部が内部整理されました。

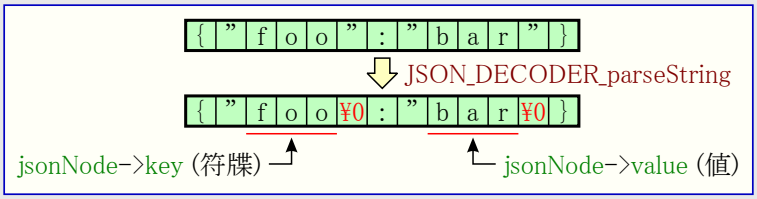
3.1. 入力緩衝部

入力緩衝部はMQTTﾗｲﾌﾞﾗﾘ、UARTﾄﾞﾗｲﾊﾞ、または他の通信方法によってJSON文字列を受け取る緩衝部です。この緩衝部はJSON_DECODER_parseString関数を使ってJSON復号部に渡されます。この関数が戻る時に緩衝部の内容が変更されています。

可能な限りﾒﾓﾘを少なく使うため、内部JSONｵﾌﾞｼﾞｪｸﾄ表現はどの文字列も保持しませんが、元々入力緩衝部に置かれた文字列への位置指示子(ﾎﾟｲﾝﾀ)を保持します。従って、文字列を分離するためその緩衝部内にﾇﾙ終了子が挿入されます。全ての符牒と文字列が入力緩衝部内です。

重大な影響はJSONﾃﾞｰﾀが使われる前に緩衝部を書き換えないように注意が払われなければならないことです。緩衝部がJSON_DECODE_parseStringとJSON構造体ｱｸｾｽ間に書き換えられる場合、これはﾃﾞｰﾀを不正にします。

図3-1. 文字列緩衝部



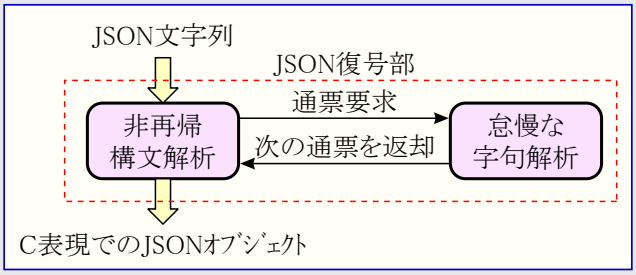
3.2. 復号

復号はJSONｵﾌﾞｼﾞｪｸﾄの文字列形式からCﾃﾞｰﾀ構造体表現への変換を意味します。これは文法解析問題で、通常、字句解析と構文解析の2段階処理で解決されます。

字句解析は空白のような余分な文字を取り除くだけでなく、入力文を通票(ﾄｰｸﾝ)に分離します。通票は単一文字(中括弧、ｺﾝﾏ、ｺﾛﾝ)と複数文字(文字列、数値)の両方です。この実装は全ての通票の一覧を作成する代わりを意味する「怠慢な字句解析」を使い、これらは構文解析によって必要とされる時に1つづつ取得されます。

構文解析は通票の流れがJSON文法の脈絡で意味を成すことを確認します。内部表現をCﾃﾞｰﾀ構造体に構築します。構文解析は入れ子にされたJSONｵﾌﾞｼﾞｪｸﾄを処理するため、本来、本質的に再帰的です。再帰(とｽﾀｯｸ溢れ異常の可能性)を避けるため、算法は実際のｽﾀｯｸの代わりにﾋｰﾌﾟ ｽﾀｯｸとWhile繰り返しを使うように内部整理されました。

図3-2. JSON復号部HLD

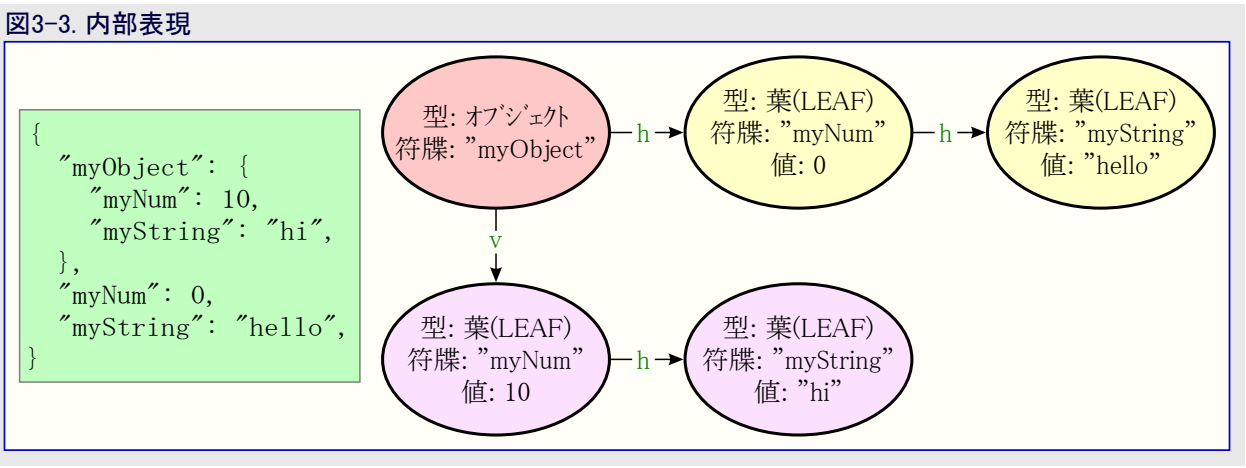


3.3. 内部表現

文字列形式構文解析後、JSONｵﾌﾞｼﾞｪｸﾄは樹状様の構造体としてﾒﾓﾘに格納されます。API関数使用で、この木構造は符牒名によって値を取得するために走査されます。各節点は符牒に対応し、(文字列または数値を持つ)葉(LEAF)節点またはｵﾌﾞｼﾞｪｸﾄ節点のどちらかであり得ます。

図3-3.はJSONｵﾌﾟｼﾞｪｸﾄとjsonNode_t構造体の木構造としてのそれの表現を示します。節点構造体は型領域、符牒領域、値領域と隣接節点への2つの位置指示子領域を持ちます。現在の節点が型ｵﾌﾞｼﾞｪｸﾄの場合、h位置指示子は同じJSONｵﾌﾞｼﾞｪｸﾄ内の次の符牒を指示し、v位置指示子は自身の最初の符牒を指示します。

図3-3. 内部表現



注: JSONｵﾌﾞｼﾞｪｸﾄが復号されるために持ち得る符牒の最大数は静的に定義されたjsonNode_t配列の長さによって制限されます。組み込み系では動的ﾒﾓﾘ配置が避けられるべきです。

注: 入れ子にされたｵﾌﾞｼﾞｪｸﾄの最大数は構文解析部によって使われるﾋｰﾌﾟの大きさによって決められます。

3.4. 符号化

例えこの実装が対応する符号化を含んでいなくても、これはprintf/sprintf関数を使い、単にこれらの関数の形式(ﾌｫｰﾏｯﾄ)ﾊﾟﾗﾒｰﾀを使うことによって達成することが容易です。以下のｺｰﾄﾞ行は文字列形式でJSONｵﾌﾞｼﾞｪｸﾄを作成します。これは測定したﾃﾞｰﾀを報告するのにMQTT/UART等を渡って送ることができます。

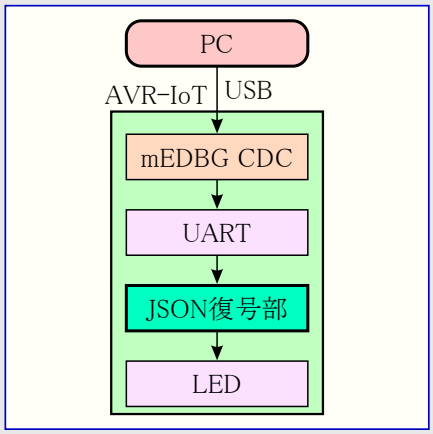
```
char json[30];
sprintf(json, "{¥"Light¥":%d,¥"Temp¥":%d}", 10, 25);
```

4. 実演：PCからAVR-IoT上の4つのLEDを制御

本章は簡単なMCCﾌﾟﾛｼﾞｪｸﾄにJSON復号部を追加する方法とTera Term端末模倣器から送るJSON形式化された命令を送ることによってAVR-IoT-WG基板上の4つのLEDを制御する方法を説明します。

この作業に使うJSONｵﾌﾞｼﾞｪｸﾄは同時に1、2、3、または4つのLEDを制御することができる明確な形式と処理論理を持つ柔軟な単一命令をもたらします。

図4-1. 簡単なJSON復号部使用事例



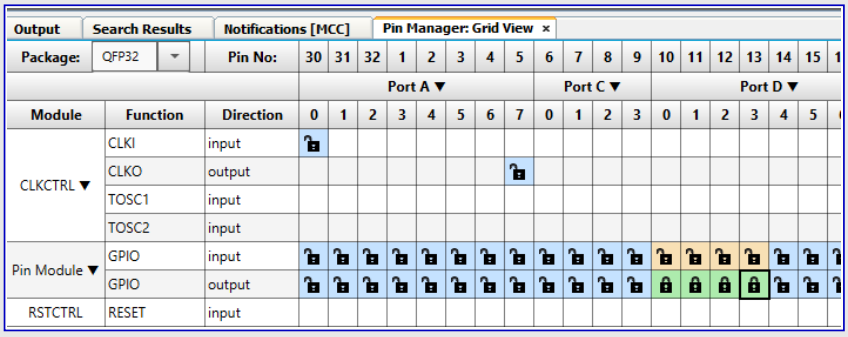
4.1. 手順

AVR-Iot基板、MPLAB® X 5.25、AVR® MCU周辺機能ﾗｲﾌﾞﾗﾘ2.0.2(より古い版も動くかもしれません)を持つMCC 3.85.1、それとGitはこれらの項での手順に従うことが必要とされます。

4.1.1. MCCでのﾌﾟﾛｼﾞｪｸﾄ自動生成

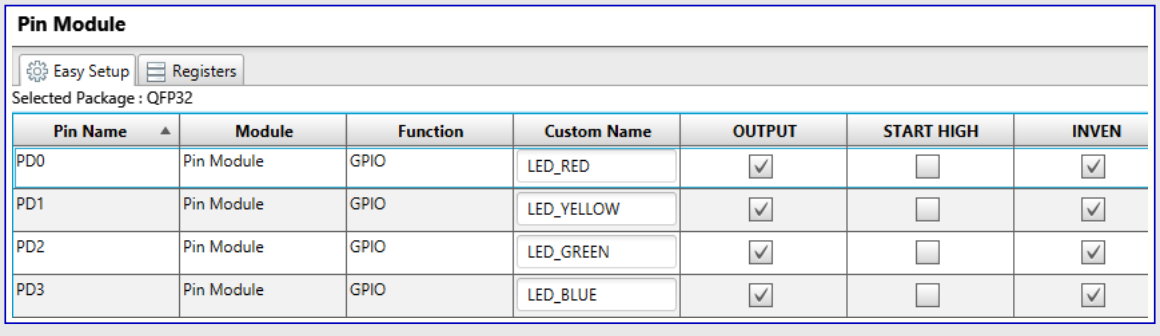
1. ATmega4808ﾃﾞﾊﾞｲｽ用の新しいMPLAB Xﾌﾟﾛｼﾞｪｸﾄを作成してください。
2. ﾌﾟﾛｼﾞｪｸﾄの構成設定を開始するためにMCCｱｲｺﾝをｸﾘｯｸしてください。
3. Pin Manager: Grid View(ﾋﾟﾝ管理部: 格子表示)ｳｨﾝﾄﾞｳでPD0～PD3のﾋﾟﾝを出力として記してください。

図4-2. ﾋﾟﾝ管理部: 格子表示部(Pin Manager: Grid View)



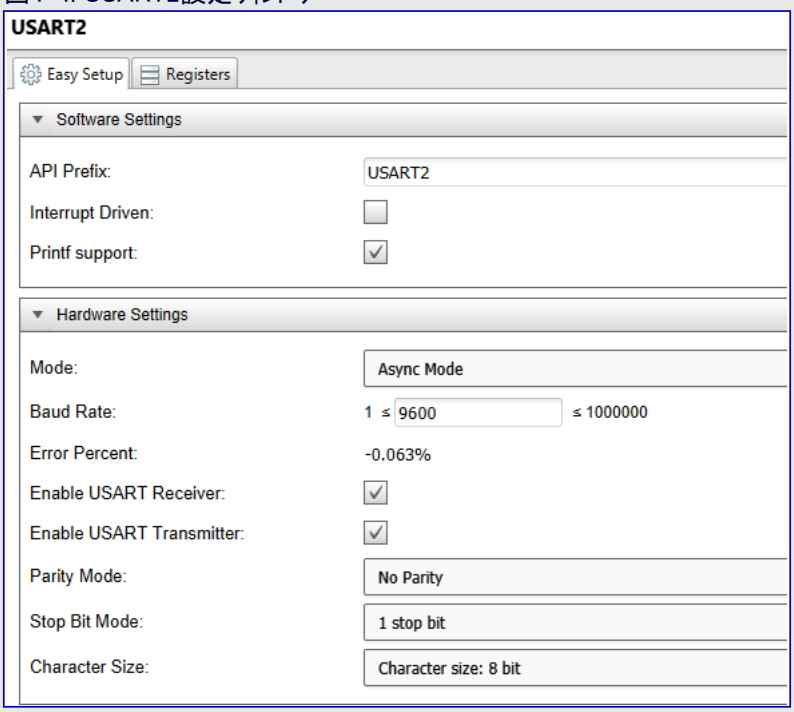
4. Pin Module(ﾋﾟﾝ単位部)ｳｨﾝﾄﾞｳでﾋﾟﾝをLED_RED、LED_YELLOW、LED_GREEN、LED_BLUEに改名し、その後にそれら全てに対してINVEN(反転I/O許可)をﾁｪｯｸしてください。

図4-3. ﾋﾟﾝ単位部(Pin Module)ｳｨﾝﾄﾞｳ



5. Device Resources(ﾃﾞﾊﾞｲｽ資源)からﾌﾟﾛｼﾞｪｸﾄにUSART2を追加してください。
6. Pin Manager: Grid View(ﾋﾟﾝ管理部: 格子表示)ｳｨﾝﾄﾞｳでUSART2がPF0とPF1を使うことを確実にしてください。
7. USART2設定ｳｨﾝﾄﾞｳでPrintf support(printf支援)がﾁｪｯｸされ、他の全ての設定が既定であることを確実にしてください。

図4-4. USART2設定ｳｨﾝﾄﾞｳ

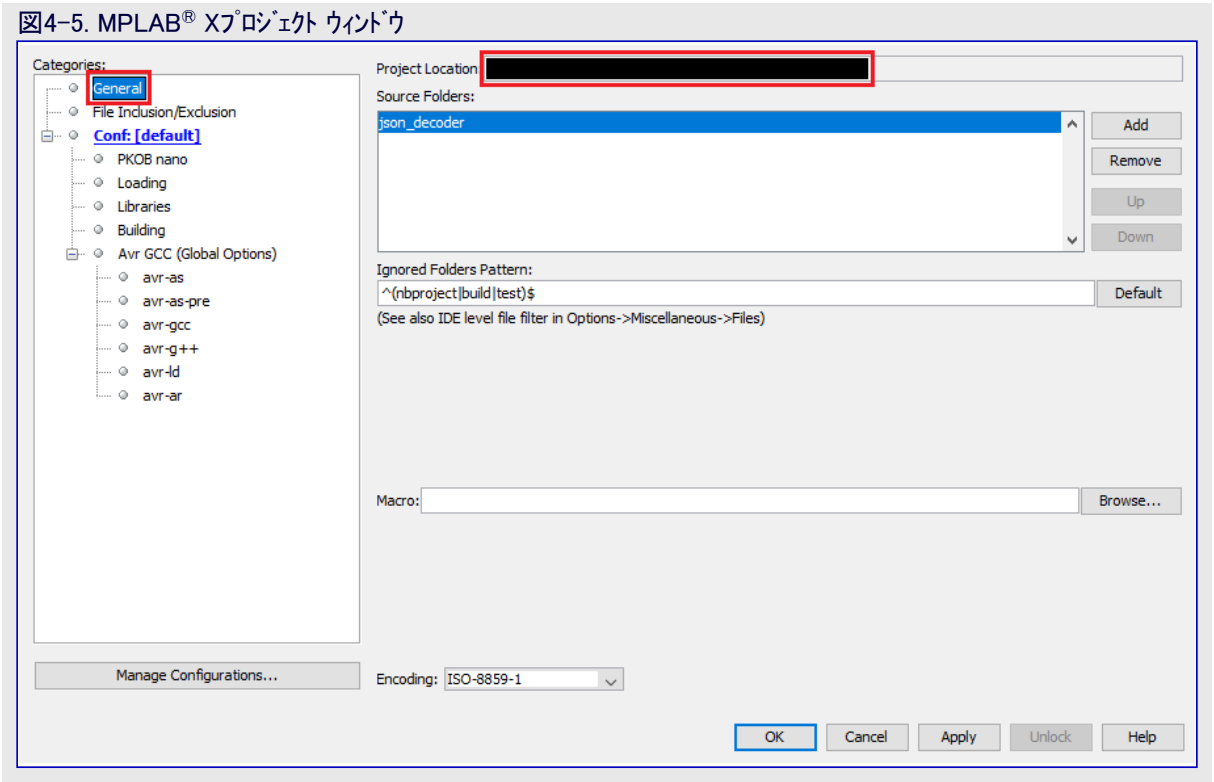


8. generateをｸﾘｯｸしてください。

4.1.2. ﾌﾟﾛｼﾞｪｸﾄにjson_decoderを追加

1. MPLAB Xﾌﾟﾛｼﾞｪｸﾄ ｳｨﾝﾄﾞｳでそれを右ｸﾘｯｸすることによってﾌﾟﾛｼﾞｪｸﾄ ﾊﾟｽを見つけ、その後にProperties(ﾌﾟﾛﾊﾟﾃｨ)をｸﾘｯｸしてください。General(全般)ﾀﾌﾞの最初の要素がProject Location(ﾌﾟﾛｼﾞｪｸﾄ位置)です。

図4-5. MPLAB® Xﾌﾟﾛｼﾞｪｸﾄ ｳｨﾝﾄﾞｳ



2. cmdまたはPower Shellのような命令行を使ってﾌﾟﾛｼﾞｪｸﾄのﾙｰﾄに誘導してください。
3. 以下の命令を使ってﾌﾟﾛｼﾞｪｸﾄにjson_decoder保管場所を複製してください。

```
$ git clone https://github.com/MicrochipTech/json_decoder.git
```

4. ﾌﾟﾛｼﾞｪｸﾄ ｳｨﾝﾄﾞｳの現在のﾌﾟﾛｼﾞｪｸﾄでSource Files(ｿｰｽ ﾌｧｲﾙ)を右ｸﾘｯｸしてNew logical folder(新しい論理ﾌｫﾙﾀﾞ)を選んでください。
5. 作成したﾌｫﾙﾀﾞをjson_decoderに改名してください。
6. json_decoderﾌｫﾙﾀﾞを右ｸﾘｯｸしてAdd Existing Item ...(既存項目追加...)を選んでください。json_decoder複製ﾌｧﾙﾀﾞから全ての.cﾌｧｲﾙを追加してください。
7. Header files(ﾍｯﾀﾞ ﾌｧｲﾙ)下に新しい論理ﾌｫﾙﾀﾞを追加し、それをjson_decoderに改名してjson_decoder複製ﾌｧﾙﾀﾞから全ての.hﾌｧｲﾙを追加してください。
8. main.cでjson_decoder/json_decore.hをｲﾝｸﾙｰﾄﾞしてください。

```
#include  “json_decoder/json_decoder.h”
```

9. Clean and Build Main Project(主ﾌﾟﾛｼﾞｪｸﾄを解消して構築)ｱｲｺﾝをｸﾘｯｸしてください。ﾌﾟﾛｼﾞｪｸﾄが成功裏に構築するでしょう。

4.1.3. 簡単な応用を作成

1. main.cでMAX_COMMAND_LENを定義してください。

```
#define MAX_COMMAND_LEN 64
```

2. while繰り返しの前にmain(主)関数で以下の変数を定義してください。

```
char command[MAX_COMMAND_LEN];
uint8_t index = 0;
char c;
```

3. while繰り返しに以下のｺｰﾄﾞを追加してください。

```
c = USART2_Read();
if(c != '¥n' && c != '¥r')
{
    command[index++] = c;
    if(index > MAX_COMMAND_LEN)
    {
        index = 0;
    }
}

if(c == '¥n')
{
    command[index] = '¥0';
    index = 0;
    executeCommand(command);
}
```

注: USARTについてもっと学ぶには「TB3216 - USARTでの開始に際して (DS90003216B)」をご覧ください。

4. main(主)関数の上にvoid executeCommand(char *command)関数を定義してください。
5. 関数の内側に於いて命令をJSONｵﾌﾞｼﾞｪｸﾄに復号してﾙｰﾄを取得してください。

```
jsonNode_t *root = 0;
jsonDecoderStatus_t ret;
char state[5];

ret = JSON_DECODER_fromString(command);
if(JSON_DECODER_OK != ret)
{
    printf("Invalid JSON string.¥r¥n");
}

JSON_DECODER_getRoot(&root);
```

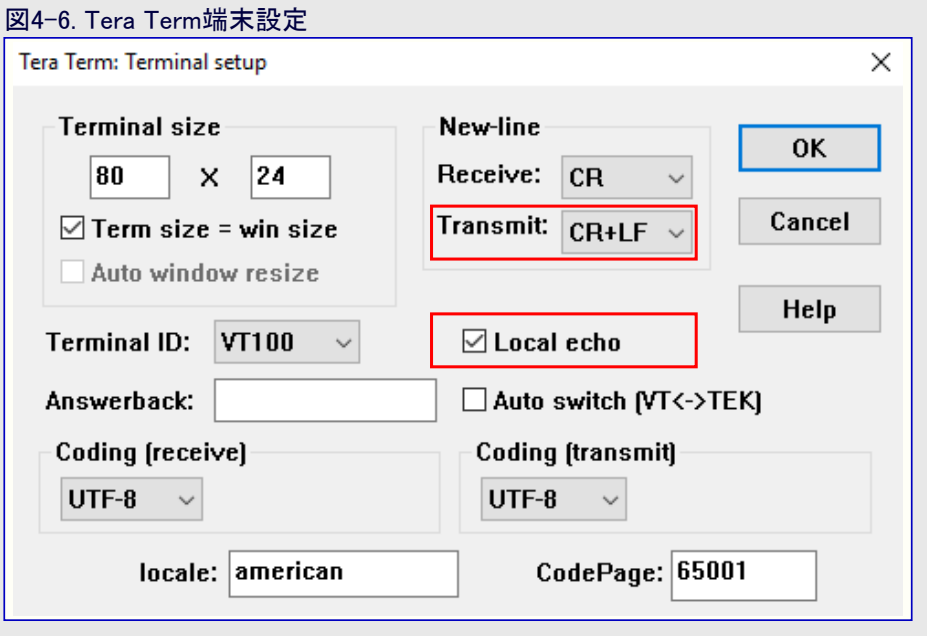
7. 他の3つのLEDに対して同様に行ってください。

   注: executeCommand関数に対する完全なｺｰﾄﾞは6. 追補です。

4.1.4. 応用を試験

1. Tera Termのような端末模倣器を開いてください。
2. ｺﾝﾋﾟｭｰﾀに接続されたAVR-IoT-WGのCOMﾎﾟｰﾄを選んでください。
3. Tera TermのﾒﾆｭｰでSetup(準備設定)⇒Terminal(端末)へ行き、New-line(新規行)のTransmit:(送信:)をCR+LFに設定し、その後にLocal echo(送信も表示)を許可してください。

図4-6. Tera Term端末設定



4. 以下のようなJSON文字列命令を送ってください。

```
{" red" :" on" }
```

- 赤色LEDがONに切り替わります。

```
{" blue" :" on" }
```

- 青色LEDがONに切り替わります。

```
{" yellow" :" on" ," blue" :" off" }
```

- 黄色LEDがONに切り替わります。
- 青色LEDがOFFに切り替わります。

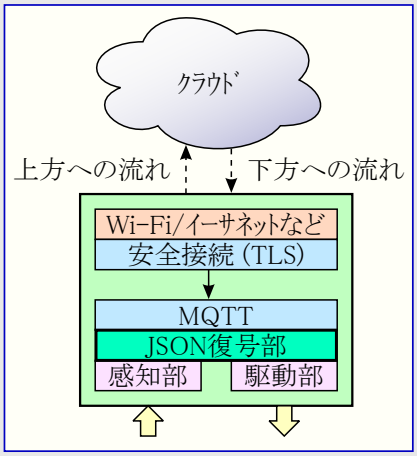
```
{" yellow" :" on" ," blue" :" off" }
```

- 黄色と赤色のLEDがOFFに切り替わります。
- 青色と緑色のLEDがONに切り替わります。

5. 実演：IoT節点との汎用ｸﾗｳﾄﾞ通信

IoT節点との代表的なｸﾗｳﾄﾞ通信は安全なTLS通信を渡って起こります。様々な応用規約がありますが、この筋書きで最も広く使わるのがMQTT(Message Queueing Telemetry Transport)です。実際のﾃﾞｰﾀが一般的にJSONｵﾌﾞｼﾞｪｸﾄです。従って、JSON復号単位部はIoT節点での不可欠な部分です。右の画像はこのような応用の全体的な高位基本構造を説明します。

図5-1. JSON復号部の高度な例



6. 追補


```
void executeCommand(char *command)
{
    jsonNode_t *root = 0;
    jsonDecoderStatus_t ret;
    char state[5];

    ret = JSON_DECODER_fromString(command);
    if(JSON_DECODER_OK != ret)
    {
        printf("Invalid JSON string.¥r¥n");
    }

    JSON_DECODER_getRoot(&root);

    ret = JSON_DECODER_getString(root, "red", sizeof(state), state);
    if(JSON_DECODER_OK == ret)
    {
        if(strcmp(state, "on") == 0)
        {
            LED_RED_SetHigh();
        }
        else if(strcmp(state, "off") == 0)
        {
            LED_RED_SetLow();
        }
    }

    ret = JSON_DECODER_getString(root, "yellow", sizeof(state), state);
    if(JSON_DECODER_OK == ret)
    {
```

```
        if(strcmp(state, "on") == 0)
        {
            LED_YELLOW_SetHigh();
        }
        else if(strcmp(state, "off") == 0)
        {
            LED_YELLOW_SetLow();
        }
    }

    ret = JSON_DECODER_getString(root, "green", sizeof(state), state);
    if(JSON_DECODER_OK == ret)
    {
        if(strcmp(state, "on") == 0)
        {
            LED_GREEN_SetHigh();
        }
        else if(strcmp(state, "off") == 0)
        {
            LED_GREEN_SetLow();
        }
    }

    ret = JSON_DECODER_getString(root, "blue", sizeof(state), state);
    if(JSON_DECODER_OK == ret)
    {
        if(strcmp(state, "on") == 0)
        {
            LED_BLUE_SetHigh();
        }
        else if(strcmp(state, "off") == 0)
        {
            LED_BLUE_SetLow();
        }
    }
}
```

Microchipｳｪﾌﾞ ｻｲﾄ

Microchipはhttp://www.microchip.com/で当社のｳｪﾌﾞ ｻｲﾄ経由でのｵﾝﾗｲﾝ支援を提供します。このｳｪﾌﾞ ｻｲﾄはお客様がﾌｧｲﾙや情報を容易に利用可能にするのに使われます。利用可能な情報のいくつかは以下を含みます。

- **製品支援** – ﾃﾞｰﾀｼｰﾄと障害情報、応用記述と試供ﾌﾟﾛｸﾞﾗﾑ、設計資源、使用者の手引きとﾊｰﾄﾞｳｪｱ支援資料、最新ｿﾌﾄｳｪｱ配布と保管されたｿﾌﾄｳｪｱ
- **全般的な技術支援** – 良くある質問(FAQ)、技術支援要求、ｵﾝﾗｲﾝ検討ｸﾞﾙｰﾌﾟ、Microchip設計協力課程会員一覧
- **Microshipの事業** – 製品選択器と注文の手引き、最新Microchip報道発表、ｾﾐﾅｰとｲﾍﾞﾝﾄの一覧、Microchip営業所の一覧、代理店と代表する工場

製品変更通知ｻｰﾋﾞｽ

Microchipの製品変更通知ｻｰﾋﾞｽはMicrochip製品を最新に保つのに役立ちます。加入者は指定した製品系統や興味のある開発ﾂｰﾙに関連する変更、更新、改訂、障害情報がある場合に必ず電子ﾒｰﾙ通知を受け取ります。

登録するにはhttp://www.microchip.com/pcnへ行って登録指示に従ってください。

お客様支援

Microchip製品の使用者は以下のいくつかのﾁｬﾈﾙを通して支援を受け取ることができます。

- 代理店または販売会社
- 最寄りの営業所
- 組み込み解決技術者(ESE:Embedded Solutions Engineer)
- 技術支援

お客様は支援に関してこれらの代理店、販売会社、またはESEに連絡を取るべきです。最寄りの営業所もお客様の手助けに利用できます。営業所と位置の一覧はこの資料の後ろに含まれます。

技術支援はhttp://www.microchip.com/supportでのｳｪﾌﾞ ｻｲﾄを通して利用できます。

Microchipﾃﾞﾊﾞｲｽ ｺｰﾄﾞ保護機能

Microchipﾃﾞﾊﾞｲｽでの以下のｺｰﾄﾞ保護機能の詳細に注意してください。

- Microchip製品はそれら特定のMicrochipﾃﾞｰﾀｼｰﾄに含まれる仕様に合致します。
- Microchipは意図した方法と通常条件下で使われる時に、その製品系統が今日の市場でその種類の最も安全な系統の1つであると考えます。
- ｺｰﾄﾞ保護機能を破るのに使われる不正でおそらく違法な方法があります。当社の知る限りこれらの方法の全てはMicrochipのﾃﾞｰﾀｼｰﾄに含まれた動作仕様外の方法でMicrochip製品を使うことが必要です。おそらく、それを行う人は知的財産の窃盗に関与しています。
- Microchipはそれらのｺｰﾄﾞの完全性について心配されているお客様と共に働きたいと思います。
- Microchipや他のどの半導体製造業者もそれらのｺｰﾄﾞの安全を保証することはできません。ｺｰﾄﾞ保護は当社が製品を”破ることができない”として保証すると言うことを意味しません。

ｺｰﾄﾞ保護は常に進化しています。Microchipは当社製品のｺｰﾄﾞ保護機能を継続的に改善することを約束します。Microchipのｺｰﾄﾞ保護機能を破る試みはﾃﾞｼﾞﾀﾙ ﾐﾚﾆｱﾑ著作権法に違反するかもしれません。そのような行為があなたのｿﾌﾄｳｪｱや他の著作物に不正なｱｸｾｽを許す場合、その法律下の救済のために訴権を持つかもしれません。

法的通知

ﾃﾞﾊﾞｲｽ応用などに関してこの刊行物に含まれる情報は皆さまの便宜のためにだけ提供され、更新によって取り換えられるかもしれません。皆さまの応用が皆さまの仕様に合致するのを保証するのは皆さまの責任です。**Microchipはその条件、品質、性能、商品性、目的適合性を含め、明示的にも黙示的にもその情報に関連して書面または表記された書面または黙示の如何なる表明や保証もしません。**Microchipはこの情報とそれの使用から生じる全責任を否認します。生命維持や安全応用でのMicrochipﾃﾞﾊﾞｲｽの使用は完全に購入者の危険性で、購入者はそのような使用に起因する全ての損害、請求、訴訟、費用からMicrochipを擁護し、補償し、免責にすることに同意します。他に言及されない限り、Microchipのどの知的財産権下でも暗黙的または違う方法で許認可は譲渡されません。

商標

品質管理ｼｽﾃﾑ

Microchipの品質管理ｼｽﾃﾑに関する情報についてはhttp://www.microchip.com/qualityを訪ねてください。

日本語© *HERO* 2020.

本技術概説はMicrochipのTB3239技術概説(DS90003239A-2020年1月)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。

世界的な販売とサービス

米国	亜細亜/太平洋	亜細亜/太平洋	欧州
本社 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 技術支援: http://www.microchip.com/support ｳｪﾌﾞｱﾄﾞﾚｽ: http://www.microchip.com	**ｵｰｽﾄﾗﾘｱ - ｼﾄﾞﾆｰ** Tel: 61-2-9868-6733 **中国 - 北京** Tel: 86-10-8569-7000 **中国 - 成都** Tel: 86-28-8665-5511 **中国 - 重慶** Tel: 86-23-8980-9588 **中国 - 東莞** Tel: 86-769-8702-9880 **中国 - 広州** Tel: 86-20-8755-8029 **中国 - 杭州** Tel: 86-571-8792-8115 **中国 - 香港特別行政区** Tel: 852-2943-5100 **中国 - 南京** Tel: 86-25-8473-2460 **中国 - 青島** Tel: 86-532-8502-7355 **中国 - 上海** Tel: 86-21-3326-8000 **中国 - 瀋陽** Tel: 86-24-2334-2829 **中国 - 深圳** Tel: 86-755-8864-2200 **中国 - 蘇州** Tel: 86-186-6233-1526 **中国 - 武漢** Tel: 86-27-5980-5300 **中国 - 西安** Tel: 86-29-8833-7252 **中国 - 廈門** Tel: 86-592-2388138 **中国 - 珠海** Tel: 86-756-3210040	**ｲﾝﾄﾞ - ﾊﾝｶﾞﾛｰﾙ** Tel: 91-80-3090-4444 **ｲﾝﾄﾞ - ﾆｭｰﾃﾞﾘｰ** Tel: 91-11-4160-8631 **ｲﾝﾄﾞ - ﾌﾟﾈｰ** Tel: 91-20-4121-0141 **日本 - 大阪** Tel: 81-6-6152-7160 **日本 - 東京** Tel: 81-3-6880-3770 **韓国 - 大邱** Tel: 82-53-744-4301 **韓国 - ｿｳﾙ** Tel: 82-2-554-7200 **ﾏﾚｰｼｱ - ｸｱﾗﾙﾝﾌﾟｰﾙ** Tel: 60-3-7651-7906 **ﾏﾚｰｼｱ - ﾍﾟﾅﾝ** Tel: 60-4-227-8870 **ﾌｨﾘﾋﾟﾝ - ﾏﾆﾗ** Tel: 63-2-634-9065 **ｼﾝｶﾞﾎﾟｰﾙ** Tel: 65-6334-8870 **台湾 - 新竹** Tel: 886-3-577-8366 **台湾 - 高雄** Tel: 886-7-213-7830 **台湾 - 台北** Tel: 886-2-2508-8600 **ﾀｲ - ﾊﾞﾝｺｸ** Tel: 66-2-694-1351 **ﾍﾞﾄﾅﾑ - ﾎｰﾁﾐﾝ** Tel: 84-28-5448-2100	**ｵｰｽﾄﾘｱ - ヴｪﾙｽ** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **ﾃﾞﾝﾏｰｸ - ｺﾍﾟﾝﾊｰｹﾞﾝ** Tel: 45-4450-2828 Fax: 45-4485-2829 **ﾌｨﾝﾗﾝﾄﾞ - ｴｽﾎﾟｰ** Tel: 358-9-4520-820 **ﾌﾗﾝｽ - ﾊﾟﾘ** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **ﾄﾞｲﾂ - ｶﾞﾙﾋﾝｸﾞ** Tel: 49-8931-9700 **ﾄﾞｲﾂ - ﾊｰﾝ** Tel: 49-2129-3766400 **ﾄﾞｲﾂ - ﾊｲﾙﾌﾞﾛﾝ** Tel: 49-7131-72400 **ﾄﾞｲﾂ - ｶｰﾙｽﾙｰｴ** Tel: 49-721-625370 **ﾄﾞｲﾂ - ﾐｭﾝﾍﾝ** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **ﾄﾞｲﾂ - ﾛｰｾﾞﾝﾊｲﾑ** Tel: 49-8031-354-560 **ｲｽﾗｴﾙ - ﾗｰﾅﾅ** Tel: 972-9-744-7705 **ｲﾀﾘｱ - ﾐﾗﾉ** Tel: 39-0331-742611 Fax: 39-0331-466781 **ｲﾀﾘｱ - ﾊﾟﾄﾞﾊﾞ** Tel: 39-049-7625286 **ｵﾗﾝﾀﾞ - ﾃﾞﾙｰﾈﾝ** Tel: 31-416-690399 Fax: 31-416-690340 **ﾉﾙｳｪｰ - ﾄﾛﾝﾊｲﾑ** Tel: 47-72884388 **ﾎﾟｰﾗﾝﾄﾞ - ﾜﾙｼｬﾜ** Tel: 48-22-3325737 **ﾙｰﾏﾆｱ - ﾌﾞｶﾚｽﾄ** Tel: 40-21-407-87-50 **ｽﾍﾟｲﾝ - ﾏﾄﾞﾘｰﾄﾞ** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **ｽｳｪｰﾃﾞﾝ - ｲｪｰﾃﾎﾞﾘ** Tel: 46-31-704-60-40 **ｽｳｪｰﾃﾞﾝ - ｽﾄｯｸﾎﾙﾑ** Tel: 46-8-5090-4654 **ｲｷﾞﾘｽ - ｳｫｰｷﾝｶﾞﾑ** Tel: 44-118-921-5800 Fax: 44-118-921-5820