



μ C3/Standard + μ Net3 評価版ガイド

OMAP-L1、AM18xx 編

2016 年 03 月 イー・フォース株式会社

1. はじめに

このたびは、μ C3/Standard + μ Net3 評価版をお試しいただきまして、ありがとうございます。本書では評価版パッケージのインストール手順、パッケージの概要、サンプルプログラム実行方法などを説明いたします。

なお、μ C3/Standard (RTOS) 、μ Net3(TCP/IP プロトコルスタック)の詳細については、評価版のインストール後、Document フォルダにインストールされる各ユーザーズガイドを参考にして下さい。

制限事項

本評価版は OMAP-L1xx パッケージ評価での試用を想定しています。添付しているプログラムを製品評価以外の目的で使用することはできません。本評価版は製品版とは異なり、RTOS、プロトコルスタックのソースコードが含まれないほか、下記の機能制限がなされています。

- RTOS の ID 数の制限

- ・タスクの個数：7（製品版では 255）
- ・その他のオブジェクト個数：それぞれ 10（製品版では 999）

※. 参考情報：μ C3/Standardユーザーズガイド「3.3.1 オブジェクトのID番号上限
のコンフィグレーション情報」

- TCP/IP プロトコルスタックの制限

- ・TCPのソケットの個数：3個以下
- ・TCP以外のソケットの個数：3個以下
- ・マルチキャスト：使用不可
- ・IP Reassembly：使用不可
- ・TCP/IPプロトコルスタックの設定：変更不可
- ・MACアドレスの設定：変更不可
- ・イーサネットドライバの設定：変更不可

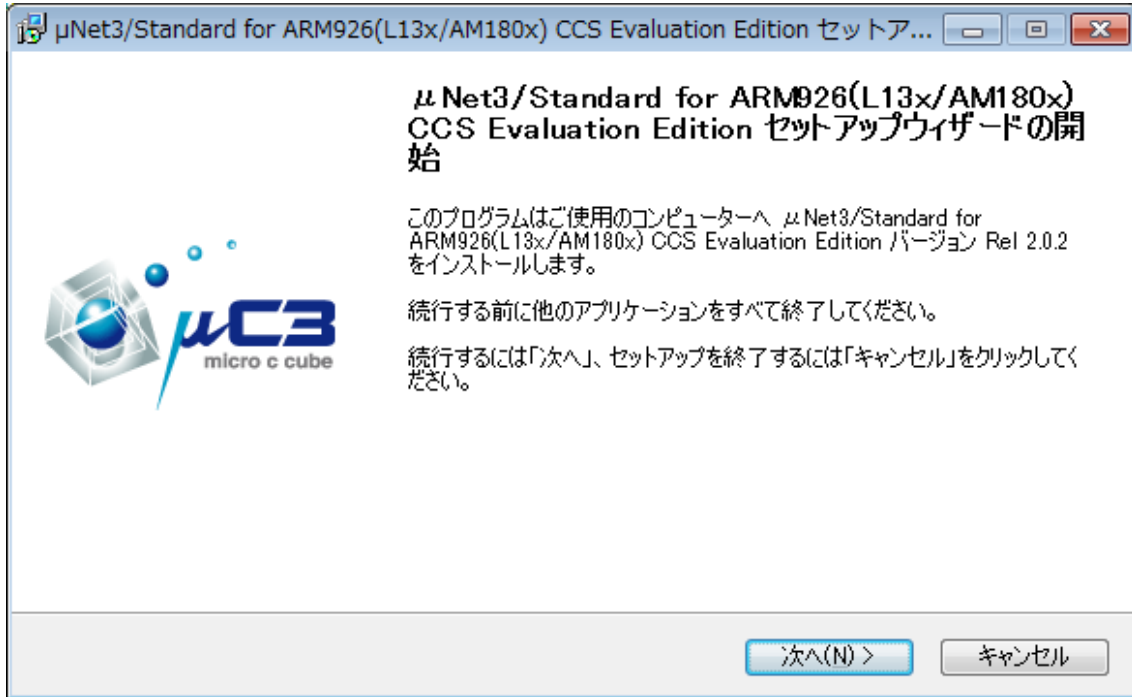
本評価版を使用するにはテキサス・インスツルメンツ社の統合開発環境が別途必要です。なお、統合開発環境の評価版は、テキサス・インスツルメンツ社のホームページから入手が可能です。

2. パッケージの概要

インストーラ

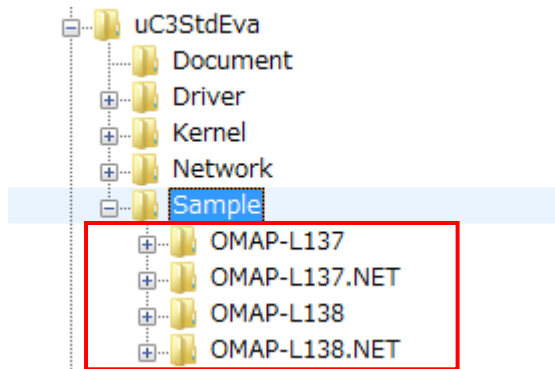
μ C3/Standard + μ Net3 評価版では、インストーラが用意されています。

uNet3s_omap11x_ccs_eva.exe を起動すると下記インストール画面が表示されるので、インストーラのメッセージに従い、評価版パッケージをインストールしてください。



フォルダ構成

以下ストールが完了すると、評価版のパッケージは、以下の構成となります。



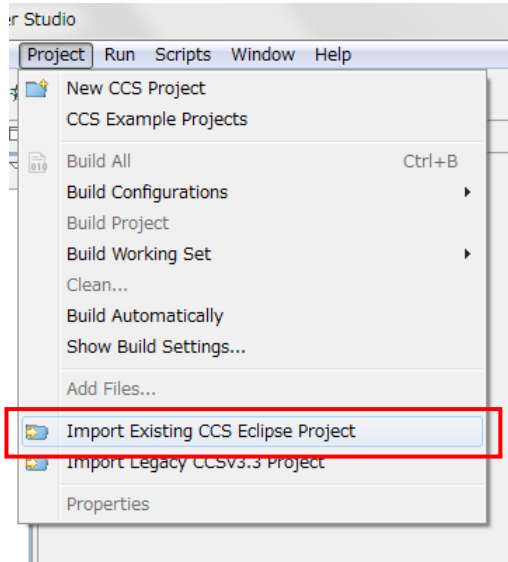
サンプルプログラムが上記のフォルダに含まれています。

次ページ以降に、評価版パッケージ内のサンプルプログラムの実行方法を説明します。

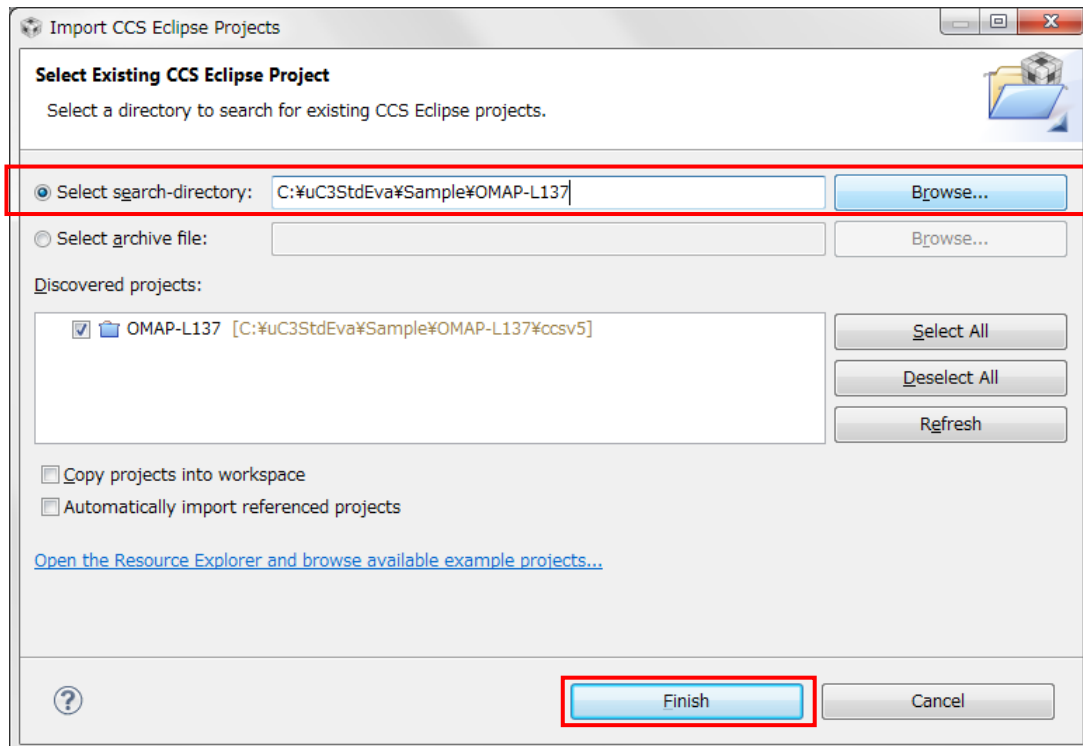
- OMAP-L137 (Spectrum Digital OMAP-L137 評価ボード)
μ C3/Standard(RTOS)上で、LED 点滅と UART 通信を実施するプログラムです。
UART 通信では、μ C3/Standard の標準 COM ポートドライバを使用しています。
- OMAP-L137.NET (Spectrum Digital OMAP-L137 評価ボード)
μ C3/Standard + μ Net3 上で、ネットワーク機能を実施するプログラムです。
- OMAP-L138 (Logic PD Zoom OMAP-L138 評価ボード)
μ C3/Standard(RTOS)上で、LED 点滅と UART 通信を実施するプログラムです。
UART 通信では、μ C3/Standard の標準 COM ポートドライバを使用しています。
- OMAP-L138.NET (Logic PD Zoom OMAP-L138 評価ボード)
μ C3/Standard + μ Net3 上で、ネットワーク機能を実施するプログラムです。

3. サンプルプログラム

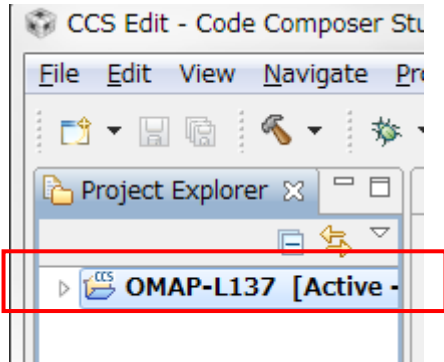
はじめにCode Composer Studio (以下CCS)にサンプルのプロジェクトを認識させる手順を説明します。Code Composer Studio (以下CCS)を起動してください。CCSメニュー「Project」→「Import Existing CCS Eclipse Project」を選択します。



「Select search-directory」欄にサンプルフォルダのパスを入れ、「Browse」を押下します。
「Discovered projects」欄にプロジェクト名が表示されるので「Finish」を押下します。



CCS の「Project Explorer」 ウィンドウにサンプルプロジェクトが追加されます。

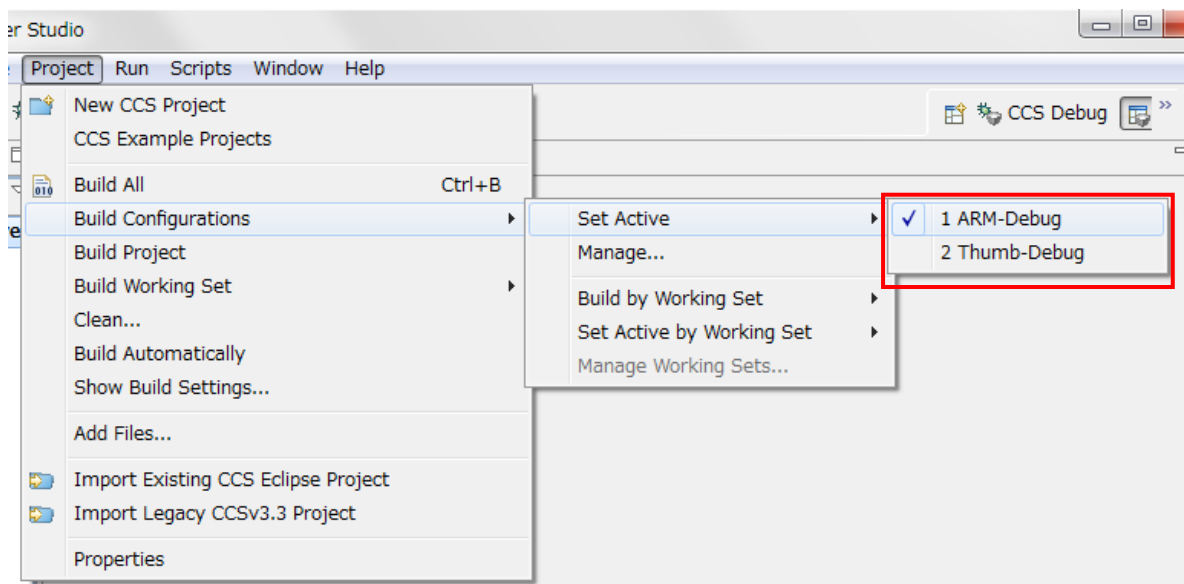


次に、2つのサンプルプログラムの実行例を説明します。

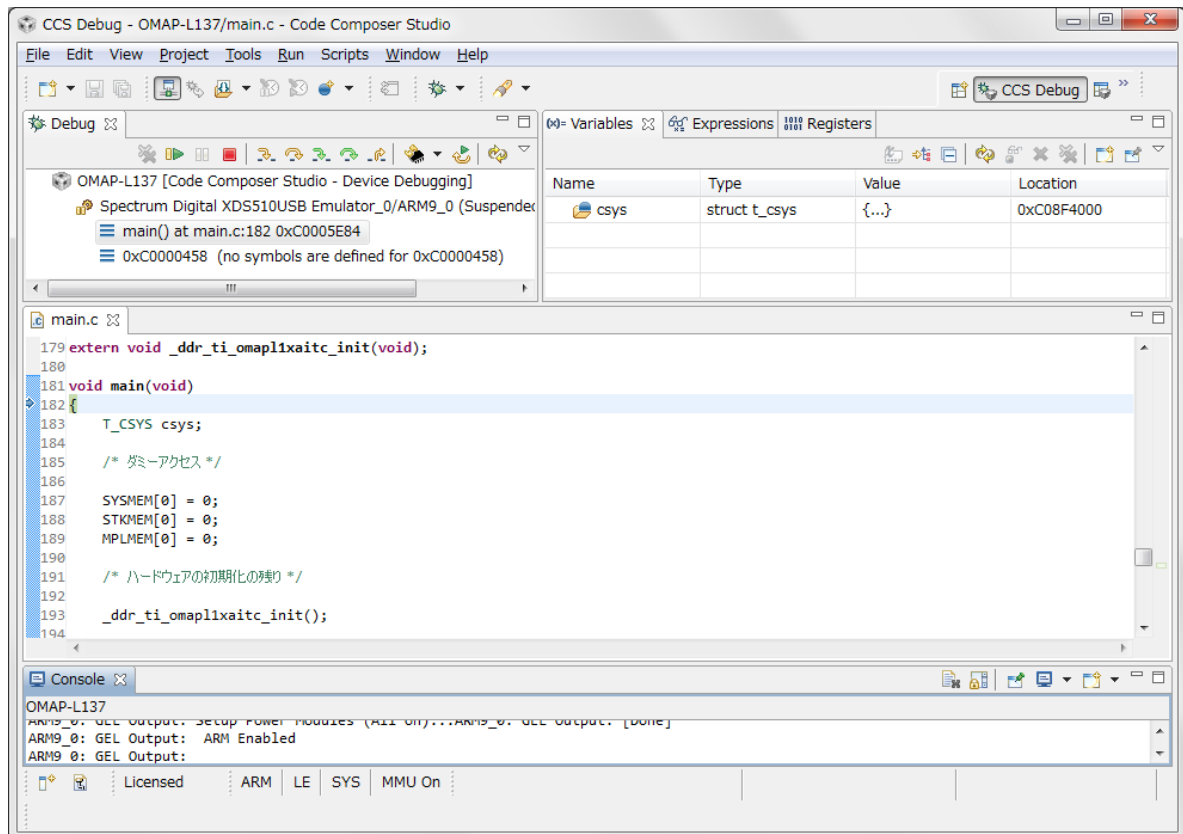
OMAP-L137

フォルダ内には、Readme.txt もありますので、こちらも合わせて参照してください。以下動作例を示します。


ワークスペースには、デバッグ用に2つのプロジェクトが作成されています。Thumb ステートを使用した"ThumbLittle"と ARM ステートのみの"ARMLittle"です。実行したいプロジェクトを、CCS メニュー 「Project」 → 「Build Configurations」 → 「Set Active」 メニューから選択してください。



その後、 「Build」 ボタンでプログラムのコンパイルを実行します。



プログラムを動作させるには、まず  「Debug」 ボタンを押下すると上記のデバッグ画

面に切り替わります。  「Resume」 ボタンを押下すると動作を開始します。

このサンプルでは、オンボードの LED 点滅と、UART からメッセージの出力の後、一文
字ずつ受信と送信を繰り返します。

UART の通信設定は下記となります。

| ボーレート | データ | パリティ | ストップビット | フロー制御 |
|-------|-------|------|---------|----------|
| 57600 | 8 bit | なし | 1 bit | Xon/Xoff |

OMAP-L137.NET

フォルダ内には、Readme.txt もありますので、こちらも合わせて参照してください。サンプルプログラムでは、DHCP クライアント・HTTP サーバ・DNS クライアントの機能を試すことができます。

また、本プログラムでは、DHCP サーバから IP アドレスを取得します。DHCP サーバがない環境で動作させる場合は、net_cfg.c、net.c 内の記述を以下のように変更してください。

変更 [net.c]

```
/* Configuration */
#define DHCP_ENB 0 /* Enable DHCP */
```

DHCP_ENB の値で DHCP 使用を指定します。1:使用、0:未使用

変更 [net_cfg.c]

```
/******
```

```
Define Local IP Address
```

デバイスに設定する IP Address, Gateway Address, Subnet Mask の必要な値に変更してください。

```
*****
```

```
T_NET_ADR gNET_ADR[] = {
```

```
{
```

```
0x0, /* Reserved */
```

```
0x0, /* Reserved */
```

```
0xC0A80A67, /* IP address (192.168. 1.103) */
```

```
0xC0A80A01, /* Gateway (192.168. 1. 1) */
```

```
0xFFFFF00, /* Subnet mask (255.255.255. 0) */
```

```
}
```

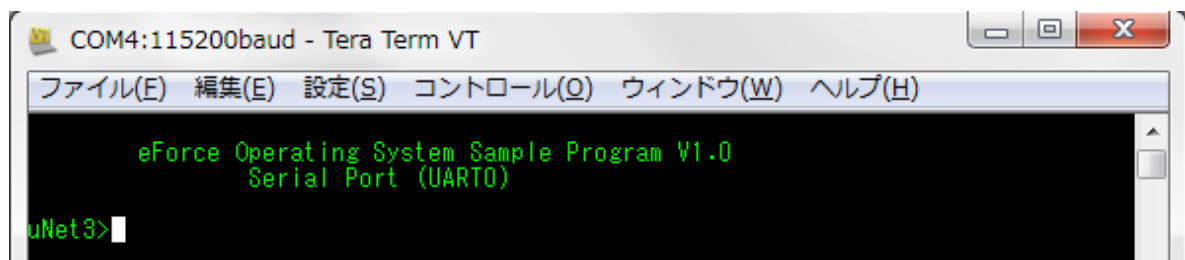
```
};
```

ビルド～プログラム実行までの流れは前項サンプル「OMAP-L137」と同様です。

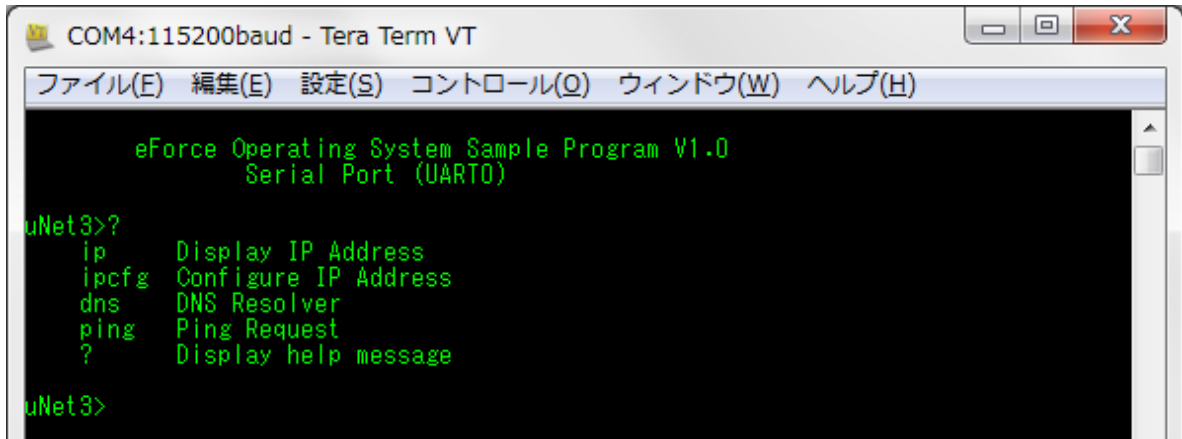
UART の通信設定は下記となります。

| ボーレート | データ | パリティ | ストップビット | フロー制御 |
|--------|-------|------|---------|-------|
| 115200 | 8 bit | なし | 1 bit | なし |

サンプルプログラムが起動すると、ターミナルエミュレータにメッセージが表示され、入力待ち状態となります。

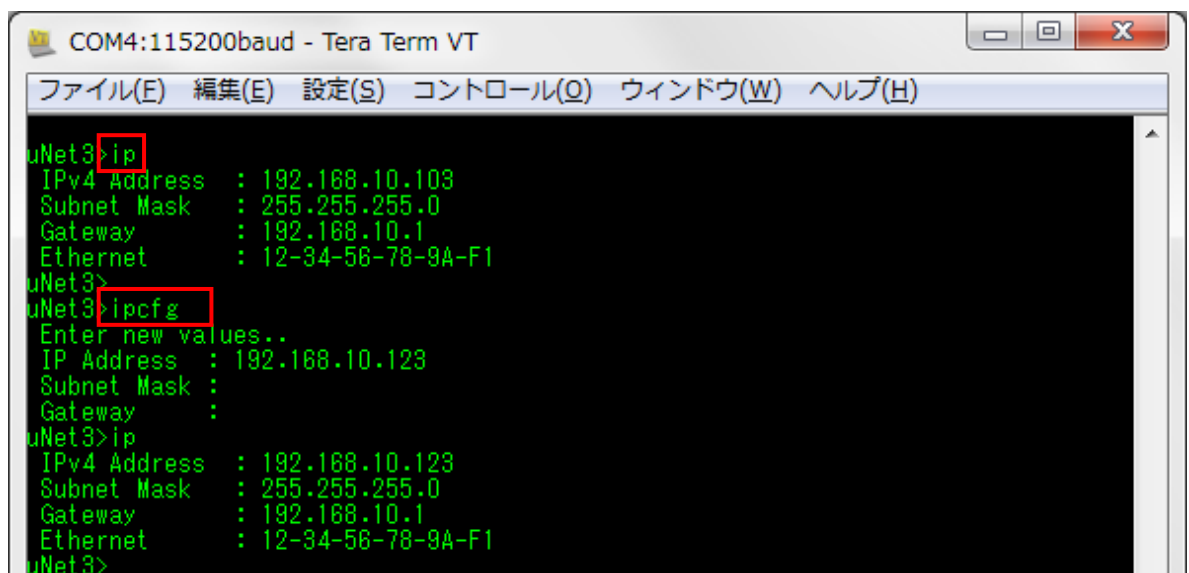


「?」を入力すると、下記のように使用できるコマンドが表示されます。



① IP アドレスの表示／変更

ターミナルエミュレータで「ip」を入力すると、デバイスに設定された IP アドレスを確認することが出来ます。また、「ipcfg」を入力後、必要な情報を入力すると現在の IP アドレスを変更できます。



DHCP の機能を使用している場合は、デバイスに割り当てられた IP Address の確認となります。DHCP の機能を使用していない場合、設定した IP Address が表示されます。以降の確認では、このアドレスを使用します。

デバイスとの通信確認は、PC の ping で確認することが出来ます。

② HTTP サーバ

ウェブブラウザからデバイスにアクセスできます。サンプルは CGI 上から入力した値をターミナルエミュレータ上に出力するプログラムとなります。アクセスする IP Address は、「①IP アドレスの表示／変更」を参考にして下さい。



上記は、ウェブブラウザからデバイスにアクセスした例を示します。ここに「1」を入力して「Set」ボタンをクリックすると、CGI のメッセージとして出力されます。

③ DNS クライアント

FQDN の A クエリを発行して IP アドレスの解決を実施します。

以下に実行例を示します。

ターミナルエミュレータで「dns (DNS サーバの IP Address) (ホスト名)」を入力すると、ホスト名の IP Address の解決を実施して、その IP Address を表示します。

4. μC3/Standard のシステムコール

μ C3/Standard は 32 ビットプロセッサが搭載された組込システム向けのリアルタイム OS です。高性能プロセッサが、より高度なリアルタイム制御に耐えられるよう、割込み禁止区間を極力なくし、割込み応答性を重視して設計しています。

μ C3/ Standard は μ ITRON4.0 仕様のフルセットを基本とし使われる可能性の少ないと思われる下記の機能を省いています。

- 静的 API と、これを解釈するコンフィグレータ
- タスク例外処理機能
- サービスコール管理機能

システムコール一覧

○：使用可

×：使用不可

△：プロセッサ依存

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|--------------------|-----|-------------|---------------------|---------------------|
| A) タスク管理機能 | | | | |
| cre_tsk / acre_tsk | ○ | ○ | × | × |
| del_tsk | ○ | × | × | × |
| act_tsk / iact_tsk | ○ | ○ | ○ | ○ |
| can_act | ○ | ○ | ○ | × |
| sta_tsk | ○ | ○ | ○ | ○ |
| ext_tsk | ○ | × | × | × |
| exd_tsk | ○ | × | × | × |
| ter_tsk | ○ | × | × | × |
| chg_pri | ○ | ○ | ○ | × |
| get_pri | ○ | ○ | ○ | × |
| ref_tsk | ○ | ○ | ○ | × |
| ref_tst | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|----------------------|-----|-------------|---------------------|---------------------|
| B) タスク付属同期 | | | | |
| slp_tsk | ○ | × | × | × |
| tslp_tsk | ○ | × | × | × |
| wup_tsk/iwup_tsk | ○ | ○ | ○ | ○ |
| can_wup | ○ | ○ | ○ | × |
| rel_wai/irel_wai | ○ | ○ | ○ | ○ |
| sus_tsk | ○ | ○ | ○ | × |
| rsm_tsk | ○ | ○ | ○ | × |
| frsm_tsk | ○ | ○ | ○ | × |
| dly_tsk | ○ | × | × | × |
| C) タスク例外処理 | | | | |
| def_tex | × | × | × | × |
| ras_tex | × | × | × | × |
| iras_tex | × | × | × | × |
| dis_tex | × | × | × | × |
| ena_tex | × | × | × | × |
| sns_tex | × | × | × | × |
| ref_tex | × | × | × | × |
| D) 同期・通信 セマフォ | | | | |
| cre_sem/acre_sem | ○ | ○ | × | × |
| del_sem | ○ | × | × | × |
| sig_sem/isig_sem | ○ | ○ | ○ | ○ |
| wai_sem | ○ | × | × | × |
| pol_sem | ○ | ○ | ○ | × |
| twai_sem | ○ | × | × | × |
| ref_sem | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|-------------------------|-----|-------------|---------------------|---------------------|
| E) 同期・通信 イベントフラグ | | | | |
| cre_flg／acre_flg | ○ | ○ | × | × |
| del_flg | ○ | × | × | × |
| set_flg／iset_flg | ○ | ○ | ○ | ○ |
| clr_flg | ○ | ○ | ○ | × |
| wai_flg | ○ | × | × | × |
| pol_flg | ○ | ○ | ○ | × |
| twai_flg | ○ | × | × | × |
| ref_flg | ○ | ○ | ○ | × |
| F) 同期・通信 データキュー | | | | |
| cre_dtq／acre_dtq | ○ | ○ | × | × |
| del_dtq | ○ | × | × | × |
| snd_dtq | ○ | × | × | × |
| psnd_dtq／ipsnd_dtq | ○ | ○ | ○ | ○ |
| tsnd_dtq | ○ | × | × | × |
| fsnd_dtq／ifsnd_dtq | ○ | ○ | ○ | ○ |
| rcv_dtq | ○ | ○ | ○ | × |
| prcv_dtq | ○ | ○ | ○ | × |
| trcv_dtq | ○ | × | × | × |
| ref_dtq | ○ | ○ | ○ | × |
| G) 同期・通信 メールボックス | | | | |
| cre_mbx／acre_mbx | ○ | ○ | × | × |
| del_mbx | ○ | × | × | × |
| snd_mbx | ○ | ○ | ○ | × |
| rcv_mbx | ○ | × | × | × |
| prcv_mbx | ○ | ○ | ○ | × |
| trcv_mbx | ○ | × | × | × |
| ref_mbx | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|-----------------------------|-----|-------------|---------------------|---------------------|
| H) 拡張同期・通信 ミューテック | | | | |
| cre_mtx/acre_mtx | ○ | ○ | × | × |
| del_mtx | ○ | × | × | × |
| loc_mtx | ○ | × | × | × |
| ploc_mtx | ○ | × | × | × |
| tloc_mtx | ○ | × | × | × |
| unl_mtx | ○ | × | × | × |
| ref_mtx | ○ | ○ | ○ | × |
| I) 拡張同期・通信 メッセージバッファ | | | | |
| cre_mbf/acre_mbf | ○ | ○ | × | × |
| del_mbf | ○ | × | × | × |
| snd_mbf | ○ | × | × | × |
| psnd_mbf | ○ | ○ | ○ | × |
| tsnd_mbf | ○ | × | × | × |
| rcv_mbf | ○ | × | × | × |
| prcv_mbf | ○ | ○ | ○ | × |
| trcv_mbf | ○ | × | × | × |
| ref_mbf | ○ | ○ | ○ | × |
| J) 拡張同期・通信 ランデブ | | | | |
| cre_por/acre_por | ○ | ○ | × | × |
| del_por | ○ | × | × | × |
| cal_por | ○ | × | × | × |
| tcal_por | ○ | × | × | × |
| acp_por | ○ | × | × | × |
| pacp_por | ○ | × | × | × |
| tacp_por | ○ | × | × | × |
| fwd_por | ○ | × | × | × |
| rpl_rdv | ○ | × | × | × |
| ref_por | ○ | ○ | ○ | × |
| ref_rdv | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|------------------------------|-----|-------------|---------------------|---------------------|
| K) メモリプール管理 固定長メモリプール | | | | |
| cre_mpf/acre_mpf | ○ | ○ | × | × |
| del_mpf | ○ | × | × | × |
| get_mpf | ○ | × | × | × |
| pget_mpf | ○ | ○ | ○ | × |
| tget_mpf | ○ | × | × | × |
| rel_mpf | ○ | ○ | ○ | × |
| ref_mpf | ○ | ○ | ○ | × |
| L) メモリプール管理 可変長メモリプール | | | | |
| cre_mpl/acre_mpl | ○ | ○ | × | × |
| del_mpl | ○ | × | × | × |
| get_mpl | ○ | × | × | × |
| pget_mpl | ○ | ○ | ○ | × |
| tget_mpl | ○ | × | × | × |
| rel_mpl | ○ | ○ | ○ | × |
| ref_mpl | ○ | ○ | ○ | × |
| M) 時間管理システム時刻管理 | | | | |
| set_tim | ○ | ○ | ○ | × |
| get_tim | ○ | ○ | ○ | × |
| isig_tim | × | × | × | ○ |
| N) 時間管理周期ハンドラ | | | | |
| cre_cyc/acre_cyc | ○ | ○ | × | × |
| del_cyc | ○ | × | × | × |
| sta_cyc | ○ | ○ | ○ | × |
| stp_cyc | ○ | ○ | ○ | × |
| ref_cyc | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|-------------------------|-----|-------------|---------------------|---------------------|
| O) 時間管理アラームハンドラ | | | | |
| cre_alm/acre_alm | ○ | ○ | × | × |
| del_alm | ○ | × | × | × |
| sta_alm | ○ | ○ | ○ | × |
| stp_alm | ○ | ○ | ○ | × |
| ref_alm | ○ | ○ | ○ | × |
| P) 時間管理オーバランハンドラ | | | | |
| def_ovr | ○ | ○ | × | × |
| ivsig_ovr | × | × | × | ○ |
| sta_ovr | ○ | ○ | ○ | × |
| stp_ovr | ○ | ○ | ○ | × |
| ref_ovr | ○ | ○ | ○ | × |
| Q) システム状態管理 | | | | |
| rot_rdq/irot_rdq | ○ | ○ | ○ | ○ |
| get_tid/iget_tid | ○ | ○ | ○ | ○ |
| loc_cpu/iloc_cpu | ○ | ○ | ○ | ○ |
| unl_cpu/iunl_cpu | ○ | ○ | ○ | ○ |
| dis_dsp | ○ | × | × | × |
| ena_dsp | ○ | × | × | × |
| sns_ctx | ○ | ○ | ○ | ○ |
| sns_loc | ○ | ○ | ○ | ○ |
| sns_dsp | ○ | ○ | ○ | ○ |
| sns_dpn | ○ | ○ | ○ | ○ |
| ref_sys | ○ | ○ | ○ | × |

| システムコール名 | タスク | 初期化 ハンドラ | タイム イベント ハンドラ | 割込み サービス ルーチン |
|-----------------------------|-----|-------------|---------------------|---------------------|
| R) 割込み管理 | | | | |
| def_inh | ○ | ○ | ○ | × |
| cre_isr/acre_isr | ○ | ○ | × | × |
| del_isr | ○ | × | × | × |
| ref_isr | ○ | ○ | ○ | × |
| dis_int | △ | △ | △ | △ |
| ena_int | △ | △ | △ | △ |
| chg_ims | ○ | ○ | ○ | ○ |
| get_ims | ○ | ○ | ○ | ○ |
| S) サービスコール管理機能 | | | | |
| def_svc | × | × | × | × |
| cal_svc | × | × | × | × |
| T) システム構成管理機能 | | | | |
| def_exc | △ | △ | △ | △ |
| ref_cfg | ○ | ○ | ○ | ○ |
| ref_ver | ○ | ○ | ○ | ○ |
| U) 独自機能・デバイスドライバ管理機能 | | | | |
| vdef_dev | ○ | ○ | × | × |
| vctr_dev | ○ | × | × | × |

5. μNet3 の API

μ Net3 は弊社 リアルタイム、オペレーティング、システム μ C3 向けに実装された TCP/IP プロトコルスタックです。

μ Net3 はワンチップマイコン向けに最適化されたコンパクトな TCP/IP プロトコルスタックとなっており、また、導入を容易にするために、わかりやすい独自 API を採用しています。

主な機能

- IPv4、ARP、ICMP、IGMPv2、UDP、TCP プロトコルをサポート
- DHCP クライアント、DNS クライアント、FTP サーバー、HTTP サーバー機能が利用可能
- コンフィグレータによる TCP/IP の設定が可能(Compact 版)
- TCP 高速再送/高速復帰アルゴリズムサポート
- IP 再構築とフラグメンテーションサポート
- 複数のネットワーク・インタフェースをサポート

API 一覧

| API 名 | |
|-------------------|------------------------|
| A) ネットワーク・インタフェース | |
| net_ini | TCP/IP プロトコルスタックの初期化 |
| net_cfg | ネットワーク・インタフェースのパラメータ設定 |
| net_ref | ネットワーク・インタフェースのパラメータ参照 |
| B) ネットワーク・デバイス制御 | |
| net_dev_ini | ネットワーク・デバイスの初期化 |
| net_dev_cls | ネットワーク・デバイスの解放 |
| net_dev_ctl | ネットワーク・デバイスの制御 |
| net_dev_sts | ネットワーク・デバイスの状態取得 |

| API 名 | |
|--------------------------|---|
| C) ソケット | |
| cre_soc | ソケットの生成(Standard 版のみ) |
| del_soc | ソケットの削除(Standard 版のみ) |
| con_soc | ソケットの接続 |
| cls_soc | ソケットの切断 |
| snd_soc | データの送信 |
| rcv_soc | データの受信 |
| cfg_soc | ソケットのパラメータ設定 |
| ref_soc | ソケットのパラメータ参照 |
| abt_soc | ソケット処理の中止 |
| D) ネットワークアプリケーション | |
| dhcp_client | DHCP クライアントの開始 |
| ftp_server | FTP サーバーの開始 |
| http_server | HTTP サーバーの開始 |
| CgiGetParam | CGI 引数の解析 |
| HttpSendText | テキストコンテンツの送信 |
| dns_get_ipaddr | ホスト名から IP アドレスの取得 |
| dns_get_name | IP アドレスからホスト名の取得 |
| E) その他 | |
| ip_aton | ドット表記の IPv4 アドレス文字列を 32 ビット値に変換 |
| ip_ntoa | 32 ビット値の IPv4 アドレスをドット表記の IPv4 アドレス文字列に変換 |
| ip_byte2n | IPv4 アドレスの配列を 32 ビット値に変換 |
| ip_n2byte | IPv4 アドレスの 32 ビット値を配列に変換 |
| htons | 16 ビット値をネットワークバイトオーダーへ変換 |
| ntohs | 16 ビット値をホストバイトオーダーへ変換 |
| htonl | 32 ビット値をネットワークバイトオーダーへ変換 |
| ntohl | 32 ビット値をホストバイトオーダーへ変換 |