R8C_M用 64K以上も使えるヘキサダンプ作成

今回は、前回の続きで、64Kbyte以上のメモリ空間も アクセスできるサブルーチンを使ってヘキサダンププログラムを作成します。

その前に、ヘキサダンプって何。？ と思われる初心者の方もいると思いますので、少し説明しておきます。 ヘキサは 16進数の事です。マイコンに興味を持つ方であれば、16進数を知らない方は いないと思いますが、一応 16進数についても説明しておきます。

通常、人間世界では、10進数を 使います。ところが、コンピュータ内部は デジタル回路の世界で 1 か 0 の 2進数の世界です。この2進数は、各桁は 0 か 1 しかありませんが、必要な数値を表わす場合は、やたら桁が 長くなります。 10進数で 100 を、2進数で表現すると 1100100 に なります。

10進数で 1000 であれば、2進数表現では 1111101000 になります。1 と 0 の長い羅列で、分かりにくいですよね。 それを 見やすくしたのが、8進数、または 16進数という事になります。

8進数は 昔ミニコンの時代に 使われてましたが、最近あまり見なくなってきました。

8進数は、2進数の 数値 1を 表す bitから 値が大きくなる方向に、3bit 刻みで、読み上げた数値です。 10進数 1000 の 1111101000 を 3bit刻みにすると、1,111,101,000 で、8進数表現では 1750oに なります。 o は 8進数オクタルを意味します。

10進数 1000 を 16進数で読み上げる場合は、2進数の数値 1の bitから 4bit刻みで区切ると 11,1110,1000 になります。 16進表現では、3E8h になります。最後の h は 16進数を意味します。

尚、8進数は、3bitで表す事の出来る数値
000 ～ 111 を そのまま 0 ～ 7 で表現しています。

16進数は、4bitで 表す事の出来る数値
0000 ～ 1111 を、10進数の 0 ～ 9 と、10以上の数値を、A ～ F の アルファベットで 表現しています。

0000 --> 0h 、 1000 --> 8h
0001 --> 1h 、 1001 --> 9h
0010 --> 2h 、 1010 --> Ah
0011 --> 3h 、 1011 --> Bh
0100 --> 4h 、 1100 --> Ch
0101 --> 5h 、 1101 --> Dh
0110 --> 6h 、 1110 --> Eh
0111 --> 7h 、 1111 --> Fh

16進数は、理解いただけたでしょうか。

あと、ヘキサダンプの ダンプという言葉を ダンプ単体で調べると、ダンプカーが 出てきますので、ここでは、ヘキサダンプで調べた意味では加工しないで、そのまま出力する。 というような意味らしいです。 ヘキサダンプは、主に プログラムのデバッグを 行う時に使用します。

用途として、メモリ内容のヘキサダンプが ありま す。 メモリの特定の アドレスから 128byte または256byte 単位で ダンプしていきます。

今回のダンプルーチンは、R8Cマイコンのメモリダンプ専用です。

あと、パソコン上では 出力したファイル内容を調べるために、ファイル内容をダンプする場合もあります。 または、ローレイヤー用途では、HDDや SDカードの内容を セクタ番号指定でダンプする場合も あります。 という事で、ダンプルーチンの概要が、理解できましたでしょうか。

今回作成したダンプ処理の出力フォーマット

ヘキサダンプの 出力フォーマットですが、各行の最初に メモリアドレスを 16進表示します。
今回は、R8Cマイコンの拡張アドレス対応なので、16進 5桁になります。 次に 16byteの データを 1byte 単位に、データを 16進ダンプします。 その右に、16byteの データを、ASCII文字として表示します。 文字として表示出来ないコードは、"." に 置き換えます。
では、実際の出力フォーマットのサンプルを お見せします。 文字列表示データの所に、
プログラム内の文字列 ＊R8C/M ExtMemory Dump routine. （ v_1.0 ）が、表示されてます。

アドレス	16進数表示のデータ	文字列表示データ

```
08080    75 C3 00 00  7C E8 75 CF  7A 04 82 05  75 CF 7C 04    u...|.u.z...u.|.
08090    80 00 EB 70  00 00 FD E4  80 00 FE FF  FB 44 00 41    ...p.........D.A
080A0    53 20 00 3E  00 2A 20 52  38 43 2F 4D   20 20 45 78    S .>.* R8C/M  Ex
080B0    74 4D 65 6D  6F 72 79 20  44 75 6D 70   20 72 6F 75    tMemory Dump rou
080C0    74 69 6E 65  2E 20 28 20  76 5F 31 2E   30 20 29 00    tine. ( v_1.0 ).
080D0    20 20 00 20  20 20 00 FF  00 00 00 00  00 00 00 00      .    ..........
080E0    00 00 00 00  F5 3B 00 FD  3A 87 00 75  C2 40 00 75    .....;..:..u.@.u
080F0    C1 0C 03 FD  8B 89 00 75  C1 A5 80 FD  78 84 00 D9    .......u....x...
```

今回の ダンプ プログラムの コマンド

今回は、余分な機能を入れずに最低限のコマンドだけにしました。 コマンドは 2つだけです。

① AS コマンド： 開始アドレス設定コマンド

例 1） AS 08000[cr] これは、開始アドレス 08000h を設定しています。

例 2） AS 10000[cr] これは、開始アドレス 10000h を設定しています。

② D コマンド： ダンプコマンド

ASコマンドで設定した開始アドレスからダンプ表示を始めます。 再度、D コマンドを 入力すると、前の Dコマンドの アドレスの続きを表示します。

という事で、使い方は簡単です。

このサンプルでは、青文字が コマンドです。

```
* R8C/M  ExtMemory Dump routine. ( v_1.0 )
>as 08000
>d
08000   EB 40 82 05  C7 02 0A 00
08010   80 00 EB 50  02 05 EB 60
08020   D8 FE B4 AA  00 03 75 C3
08030   75 C3 00 00  7C EA B4 AA
08040   B4 AA 80 04  75 C3 02 00
08050   AA 00 03 75  C3 00 00 7C
08060   00 03 75 C3  00 00 7C E8
08070   03 75 C3 0C  00 7C E8 A2
>d
08080   75 C3 00 00  7C E8 75 CF
08090   80 00 EB 70  00 00 FD E4
080A0   53 20 00 3E  00 2A 20 52
080B0   74 4D 65 6D  6F 72 79 20
080C0   74 69 6E 65  2E 20 28 20
080D0   20 20 00 20  20 20 00 FF
080E0   00 00 00 00  F5 3B 00 FD
080F0   C1 0C 03 FD  8B 89 00 75
```

ここより右側は 切り取っています

今回の ダンプ プログラム使う前の 準備

話が、前後して申し訳ありませんが、R8C/Mマイコンに 今回のダンププログラム 109_ExMemDump.mot を 書き込みます。
R8C/M110AN 、R8C/M120AN 共通で 使用できます。 書き込み後は、マイコンを 実行モードにしておいて下さい。

パソコン側は、Tera Term を 使用します。
シリアル通信モードで、オープンします。
マイコンを パソコンに 接続します。
通信ポートは、各実行環境により異なるので マイコンと通信するポート番号を設定して下さい。 それと、ボーレイトは、38400 bps を 設定して下さい。 あとは、デフォルトのままで OKです。 どれか、パソコンの キーを 押すと オープニングのメッセージと、プロンプトの > が 出てきます。

R8C/Mシリーズのメモリマップ

秋月電子で販売している R8C/Mマイコンを 想定して話をします。 秋月電子のサイトでは

プログラムメモリ: 2KB
データ用 EEP-ROM : 2KB
RAM : 256B と なっています。
これは メーカーのデータシートに このように 記載されているからと思います。

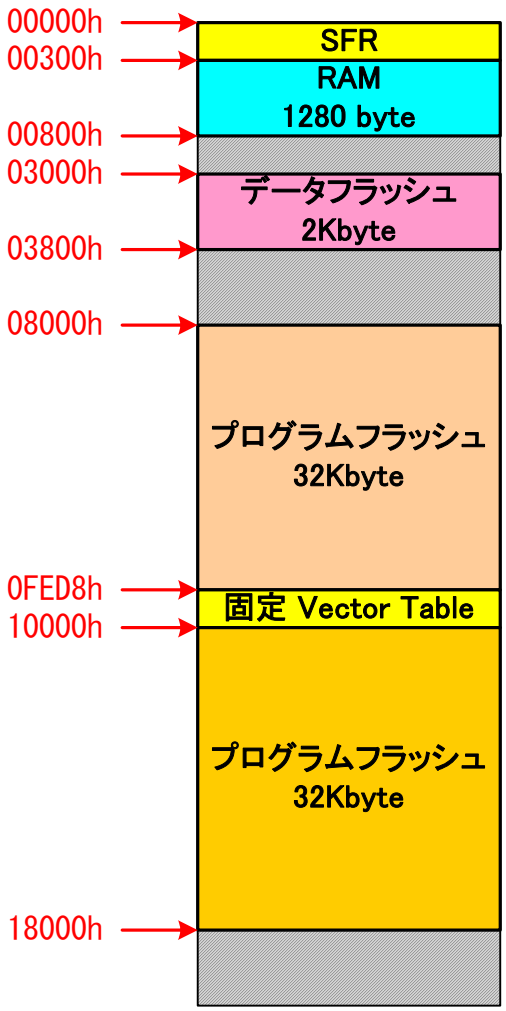
でも 実際は
プログラムメモリ: 32KB ＋ 32KB
データ用 EEP-ROM : 2KB
RAM : 1280Byte
(ネット上には、RAM容量を 1280Byteより 若干多い値で書いている方も おられます。 私は、未確認なので、1280Byteとします。)
プログラムメモリは、茶色で表示している 32KB 10000h ～ 18000h に 存在します。

10000h から存在する 32KBの プログラムメモリですが、64KB内に存在する 32KBの プログラムメモリは、8000h から始まります。

であれば、8000h から始まる 32KBの プログラムメモリと、10000hから始まる 32KBの プログラムメモリは、連続しているように見えるので 8000hから始まる 64KBの プログラムメモリとして使えそうに見えますが、実際は 途切れている箇所があります。 僅かですが、0FED8hからハードウェアベクトルテーブル等があります。

その関係で、連続した 64KBのプログラム書き込みエリアとしては、使えません。32KBのフラッシュROMが 2つ有ると解釈した方がよさそうです。 10000hから 始まるフラッシュROMは、固定的な データテーブルを格納する用途で、使うのがよさそうです。

で、実際のメモリマップは、以下の様になります。

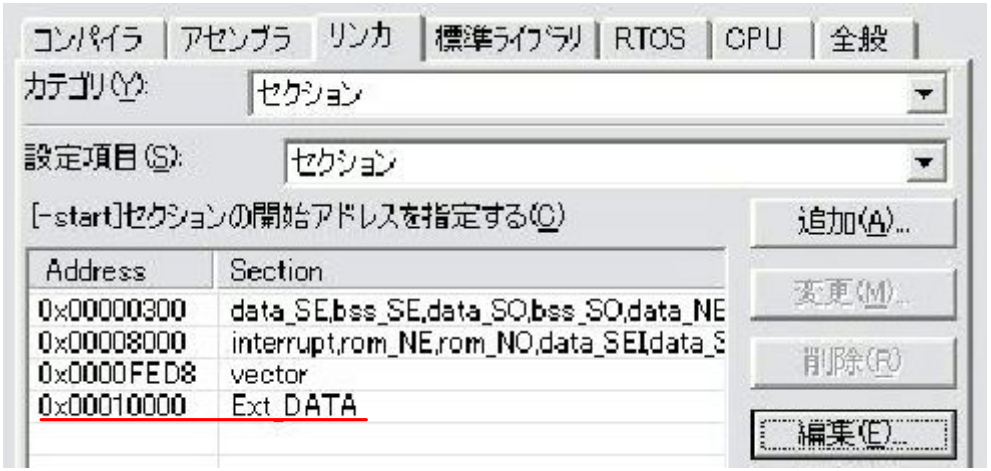


字が、小さくてすみません。

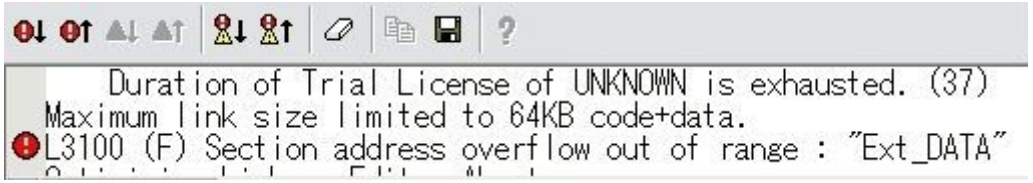
では、プログラムを書き込んで、Tera Tarmを起動して、メモリダンプを 行ってみます。

アドレス64K以上に データを書き込むには

64Kbyte以上のメモリアドレスに、データを書き込むには、64Kbyteから始まる セクション情報を設定する必要が ありま す。 試しに Ext_DATAというセクション名で Address 10000h を、設定してみました。 セクションの設定は、出来ました。 赤線部分です。



ところが、そのセクション名 Ext_DATAを アセンブラで宣言すると、エラー L3100 が 出ました。



Section address overflow out of range Ext_DATA

が、出ました。残念。 やはり何らかの 64Kbyteの制限が あるようです。

まあ、正攻法では、出来ないみたいですが、別の 何らかの方法を 検討してみます。

R8Cマイコン用の gccのコンパイラを 使うと出来るような事を書いてあるサイトは、見た事があります。 試しに、Windows10上にて、仮想Linux環境を実現する Microsoft純正の WSL2という環境を使って R8Cのクロスコンパイラを ダウンロードビルドする試みを行いましたが、途中でエラーで止まりました。 原因不明。?

他のサイトにて、WSL1で gccコンパイラが、使えていたのに、WSL2に バージョンアップして使えなくなった。 というような記事を見ました。

何と言うか、マイクロソフト あるある、な感じが します。 あと やるのであれば、Windows上の仮想Linux環境では なくて ubuntu とかメジャーな Linux OS を 新たに パソコンのHDDに インストールして、R8C用の gccをインストール、ビルドして試す事になります。

この際、とことん追求してみたい気もするのですが、初心者の方の敷居が、だんだん高くなるようで、初心者の方に申し訳ない気がします。

あと、もう一つ 64Kの壁を越える方法として邪道ですが、MOTファイル(ヘキサファイル)を編集して、2つの モジュールを連結する方法も考えられます。 ちょっと泥臭い作業ですし、人に勧められる方法とは、言い難いですね。

という事で、R8C/Mマイコンの 10000hから始まる フラッシュROMに関しては、バックグランドで、気が向いた時に やる事にします。

拡張領域の ROMを 使わなくても 32Kbyteのプログラムフラッシュがあるので、アマチュアであれば、結構使えると思います。

R8Cマイコンは、ディスコン品で、いずれなくなるでしょうが、50円で 32Kbyteのマイコンは二度と出て来ないと思います。 足ピンが 少ない等の制限は ありますが、制限のある中でアプリケーションの可能性を考えるのも面白いと思います。 初心者に分かりやすい形で、アプリケーション寄りというか、応用を 楽しく紹介して行ければと 思います。