

機能を拡張するシリアル通信と コマンド処理

これまでに制作したデジタル・クロックに通信機能を付けます。USBまたはRS-232Cでデジタル・クロックとパソコンを接続し、パソコンからWindowsハイパーターミナルなどのターミナル^①・ソフトを使って、デジタル・クロックにコマンドを与えて制御できるようにします。シリアル通信機能を活用することで、PICの応用範囲は格段に広がります。

このアイコンは、章末に用語解説があります

9-1 仕様、機能の検討

ここでは、パソコンから指示を出すことで、時計合わせや予約プログラムの設定、予約プログラム内容の一覧表示ができるようにします。パソコン側には専用アプリケーションを用意しなくても済むように、汎用の通信ソフト(Windowsハイパーターミナルなど)を使います。そのため、通信で流れるデータはアルファベットや数字などの文字といくつかの制御コードです。

コマンド・フォーマット

処理を簡単にするために、コマンド・コードはアルファベット1文字(大文字、小文字の区別なし)、コマンド・パラメータは0~9までの数値(桁数は可変)とします。コマンド・パラメータをもたないコマンドもあります。

制御コマンドは次の四つを用意します。

- ▶ 時刻設定「T」
- ▶ 全予約プログラムの消去「A」
- ▶ 予約プログラム設定「P」
- ▶ 予約プログラム一覧表示「D」

コマンドのフォーマットを次のように定義します。

▶ 時刻表示(T)コマンド

Thhmmss

hhは2桁の時(00~23)、mmは2桁の分(00~59)、ssは2桁の秒(00~59)

デジタル・クロックの時刻を設定します。パラメータは6桁固定です。

▶ 全予約プログラムの消去(A)コマンド

A

パラメータなし

登録されているすべての予約プログラムを一度に消去します。

▶ 予約プログラム設定(P)コマンド

Pnn

nnは予約プログラム番号(1~10)

1件分の予約プログラムを登録します。このコマンドは次のように1項目ずつ入力内容を聞いてくる対話形式になっています。

Time?(HHMM) hhmm[Enter]

Port?(1-4) p[Enter]

ON/OFF?(1/0) n[Enter]

Repeat?(1/0) r[Enter]

アンダラインのついた文字部分はユーザが入力する部分で、hhは2桁の「時」、mmは2桁の「分」、pは1~4のポート番号、nとrは1または0の数字、[Enter]はキーボードのEnterキー押下を表しています。途

リスト9-1 シリアル通信コマンドの操作例
パソコンのターミナル・ソフトで操作しているときのコマンド入力と、デジタル・クロックの応答出力の表示例。

```
a [Enter]
OK

t120000 [Enter]
OK

p4 [Enter]
Time (HHMM)? 1700 [Enter]
Port? (1-4) 2 [Enter]
ON/OFF? (1/0) 1 [Enter]
Repeat? (1/0) 1 [Enter]
OK

d [Enter]
No. Time PT SW Rep
01 ----- - - - - -
02 ----- - - - - -
03 ----- - - - - -
04 17:00 P2 ON R
05 08:00 P2 OFF R
06 ----- - - - - -
07 ----- - - - - -
08 ----- - - - - -
09 ----- - - - - -
10 ----- - - - - -
```

☑はパソコンのキーボードでのEnterキー入力を表す。
アンダラインの付いた文字はキーボードからの入力文字、それ以外はデジタル・クロックからの応答を表す

中で入力を打ち切りたいときは、スペース・キーを押します。

▶ 予約プログラム一覧表示 (D) コマンド

D

パラメータなし

10件分の予約プログラムを一覧で表示します。


実際にコマンドをターミナルから入力して、その応答が返ってきたときの表示はリスト9-1のようになります。

9-2 プロジェクトの複製，プログラム拡張

プロジェクトの複製


既存のプロジェクトを複製して、そこに手を加えていくことにします。まず、これまでに作った“Clock”プロジェクトをフォルダごとWindowsエクスプローラでコピーします。コピーしたフォルダは“Clock2”という名前に変更します。

次にWIZ-Cでこのプロジェクトを開きます。ファイル・メニューから[File]-[Close All Pages]をクリックして、エディタ・ウィンドウにあるすべてのソース・ファイルをいったん閉じます。


アプリケーション・デザイナに移り、 (Generate Application) ボタンをクリックしてコードを再構築します。これは、プロジェクト・フォルダを変更したことを反映させるための処置で、“Clock_Main.c”と“Clock_User.c”でインクルードしているヘッダ・ファイル“Clock_Auto.h”の絶対パスを補正しています。その他、独自に作成したファイルの中に絶対パスで指定したヘッダ・ファイルをもつものは、手作業でフォルダ名を修正する必要があります。

次に新規にファイルを作成して、そのファイルをプロジェクトに追加します。ファイル名は“SerCmd.c”とします。その後、このファイルにコーディングしていきます。

エレメント，Occurrence関数の追加

シリアル通信のドライバを追加します。アプリケーション・デザイナ上で (Interrupt Driven Serial Interface) エレメントを追加し、プロパティを次のように設定します。

- ▶ Serial Bit Rate 19200
- ▶ Use XON/XOFF flow control OFF
- ▶ Receive Buffer Size 32
- ▶ Transmit Buffer Size 16

次に、同エレメントのOccurrence “RxByte”に“GetRx”関数を登録します。ここで再び (Generate Application) ボタンをクリックしてコードを再構築しておきますが、まだ“GetRx”関数は存在しないので、コンパイル・エラーが発生します。これは無視してかまいません。

既存ソース・ファイルの変更

すでにコーディングしてあるファイルに変更を加えます。

“Clock_User.c”と“Menu.c”，“Global.h”ファイルをリスト9-2のように修正します。

リスト9-2 既存ソース・ファイルの変更

```
0050: void UserInitialise()
0051: {
0052:     int i;
0053:     Secc = 0; // 25進ソフトウェア・カウンタ
0054:     TF1Sec = false; // 1秒経過フラグ
0055:
0056:     LCDClear(); // LCDの表示をクリア
0057:     LCDString("Hello"); // ウェイクアップ・メッセージを表示
0058:     InitDatas(); // メニュー処理関係の変数初期化
0059:     InitSCmd(); // シリアル通信関係の変数初期化

0088: void UserLoop()
0089: {
0090:     TskClockUpdate(); // 1秒ごとに時計更新, 予約プログラムのチェック
0091:
0092:     TskClearAll(); // 全プログラムのクリア・タスク
0093:     TskSetProg(); // 1プログラム書き込みタスク
0094:     TskSelClr(); // 実行済みプログラムのクリア・タスク
0095:     TskDumpProg(); // 予約プログラムのいち覧表示タスク
0096: }

0120: //
0121: // シリアル・データ 1バイト受信
0122: //
0123: void GetRx() {
0124:     BYTE ch;
0125:     if(GetRxSize() > 0) {
0126:         // 受信データがあるとき
0127:         ch = WaitRx(); // 受信データを1バイト取り出す
0128:
0129:         CmdProc(ch);
0130:     }
0131: }
```

(a) "Clock_User.c"にグレーの濃い部分の行を追加

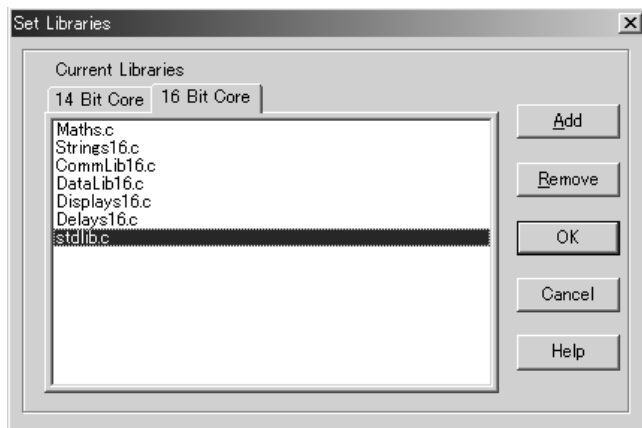


図9-1 拡張ライブラリの追加
拡張ライブラリ・ファイルのstdlib.cを[16 Bit Core]のページへ追加する。[Add]ボタンをクリックしてライブラリ・ファイルを指定する。

リスト9-2 既存ソース・ファイルの変更(つづき)

```
0013: // 予約プログラムのカレント・データ(このファイル内のみ有効なグローバル変数)
0014: static BYTE PrgHour, PrgMin;
0015: static BYTE PrgNum;
0016: static BYTE PrgPort;
0017: static BYTE PrgOnOff;
0018: static BYTE PrgRep;
```

(b) "Menu.c"の上記の変数をstatic変数に変更

```
0078: // ***** SerCmd.c 定義関数,変数 *****
0079: void InitSCmd(void); // 変数初期化
0080: void TxStr(char *str); // 文字列をシリアル・ポートへ出力
0081: BOOL SetCmdBuf(char ch); // シリアル・コマンド・バッファへ入力文字を挿入する
0082: void CmdProc(char ch); // コマンド処理
0083: void ExecCmd(void); // コマンドを解析して実行
0084:
0085: // タスク
0086: // 全予約プログラム内容一覧表示 タスク
0087: void TrgDumpProg(void); // 起動関数
0088: void TskDumpProg(void); // タスク本体
0089:
0090: extern char CmdBuf[8]; // コマンド・バッファ
0091: extern BYTE CPtr; // コマンド・バッファの挿入位置
```

(c) "Global.h"に上記のプロトタイプ宣言, extern宣言を追加

WIZ-Cライブラリの追加

“SerCmd.c”のなかで, atoi という, 文字列を整数値に変換するライブラリ関数を使用していますが, これは標準ではインストールされていません. この関数を使用するには, WIZ-CのCD-ROMから LibraryExtensionsをインストールしなければなりません.

まず, CD-ROM内の“Libraries¥LibExt”フォルダにある“SETUP.EXE”を実行して“stdlib.c”と“stdlib.h”ファイルをインストールしてください. インストール先は, 標準的な環境では“C:¥Program Files¥FED¥PIXIE¥Libs”フォルダとなります.

次に, WIZ-Cメイン・ウィンドウのメニュー [File]-[Libraries]をクリックして, “Set Libraries”ウィンドウを開き, [16bit core]のタブをクリックした上で, [Add] ボタンをクリックします. ファイル・オープンダイアログ・ボックスが開くので, 今インストールした“stdlib.c”を追加します. これで“Set Libraries”ウィンドウを閉じれば, stdlibが使えるようになります(図9-1).

ここでは, ターゲットが18F452なので, [16bit core]のほうに登録しましたが, 16F877や16F88などの14ビット・コアのCPUで使いたい場合は, [14bit core]のほうにも登録します.

拡張プログラムの概要

拡張するシリアル通信のプログラムは, 大きく分けて次の二つで構成されています.

- ▶ シリアル・コマンドを受信し, それを解析して実行する
- ▶ コマンドの応答をパソコンへ送信する

▶ コマンド解析

パソコンのキーボードから1文字ずつ入力され、シリアル通信でデジタル・クロックへ送られてきた文字は、いったんクロック側のコマンド・バッファ `CmdBuf` (配列変数)へ蓄えられます。そして、パソコン側でEnterキーが押されてC/Rコード(‘`␣`’)が送られてくると、それまでに蓄えられた文字列がコマンド文字列として確定します。

Enterキーが押される前にBS(バックスペース)キーが押されて、デジタル・クロック側にBSコード(‘`␣`’)が送られてくると、直前に入力した1文字をコマンド・バッファ `CmdBuf` から削除し、パソコン側にも削除のコードを送信して、ターミナル上からも1文字消去します。

また、スペース・キーが押されてSPコード(0x20)が送られてくると、それまでの文字入力は破棄されてコマンド・バッファは空になります。

確定したコマンド文字列は1文字目がコマンド・コード、それに続く数字がコマンド・パラメータと見なされます。まず、コマンド・コードを調べて、該当するコマンドがあれば、続けてパラメータ部分を数値に変換します。この数値は設定すべき変数に代入されます。

コマンドによってはパラメータをもたないものもあります。この場合は、それに応じた処理を直ちに実行します。

▶ コマンド応答

PICでは、メモリ容量の関係で、連続した大きなメモリ領域を確保することができません。そのため、送信バッファもそれ相応のものになります。

一度に送信するデータの量が送信バッファ・サイズより多いときは、それをいくつかのブロックに分けて送信しなければなりません。今回このケースにあたるのが、予約プログラム一覧表示(D)コマンドです。このコマンドでは、10件の予約プログラムの内容を一覧で表示するために、多くのデータを送信する必要があります。

そこで、データを分割して送信するために、EEPROMの書き込みの際に使ったようなタスク機構を使います。

1サイクルに送信できる文字数(送信バッファ・サイズ)は最大16文字としています。送信するデータは最大16バイトのブロックに分割し、1ブロックの送信が終わったら次のブロックを送信するというように、ブロックを順次送信するようにします。

プログラムの説明

▶ 変数類の宣言

リスト9-3の行は、ヘッダ・ファイルのインクルード宣言です。0008行目の“`stdlib.h`”は、文字列からint型整数へ変換する`atoi`ライブラリ関数を使うために、また、0010行目の“`ctype.h`”は、大文字に変換する`toupper`マクロを使うためにインクルードしています。

リスト9-4の部分で予約プログラムの一時的記憶変数を定義しています。

これらは外部変数(グローバル変数)ですが、同名の外部変数が“`Menu.c`”にもあります。そのため、`static`宣言しています。こうすることにより、これらの変数は“`SerCmd.c`”の中だけで有効なグローバル変数(ファイル・ローカル)となります。ただし、“`Menu.c`”内の同名の変数も同じように`static`宣言を付けるのを忘れないでください。同名の変数を使うことは混乱のもとでもあり、あまりお勧めできませんが、ここでは、使用例として使っています。