読者の広場

ウィンドウズ環境における計測プログラム

1. はじめに

私が所属する研究室では，画像計測における必要性から，Windows 環境で C 言語による計測プログラミングを開始した．そして，現在までの経験を通じて MS-DOS 環境から Windows 環境に移行する際の「障壁（メッセージ，スレッドについての理解）」を認識した．本稿では，この二点についての解説を行い，さらにサンプルプログラム（ソースコードは，私が運営する「計測プログラミングのホームページ」[1]よりダウンロード可能），および参考文献[2-6]を示す事にする．これらの情報に加えて，適切なホームページを検索・参照する事で，一般的な C 言語の知識から出発して Windows プログラミングを行う事が出来る筈である．この点について，卒業研究を通じて既に確認済みであるので，意欲ある読者諸氏は大いに心強くして挑んでいただきたい．

2. Windows プログラムの特徴

1：基本構成（メッセージの役割） 一般的な Windows プログラムの根幹は，WinMain 関数とウィンドウのコールバック関数（ウィンドウプロシージャと呼ばれる）である[2,5,6]．

全ての Windows プログラムの実行は，WinMain 関数の呼び出しで始まる．この関数の役割は，まずウィンドウを生成することである．計測プログラミングの場合，「ボタン」等を使用するために，「ダイアログボックス」[2,5,6]と呼ばれる特殊なウィンドウを生成させることが多い（図 1 のサンプルプログラム画面もダイアログボックスである）．

そして，もう一つの WinMain 関数の重要な役割は，オペレーティングシステム（OS）からのメッセージの受け取りとその処理である．OS より送られるメッセージは，何らかのイベント（キーを押した，ウィンドウ中のボタンが押された等）に対応している．WinMain 関数は，受け取ったメッセージを適宜加工し，OS を介してウィンドウプロシージャに伝える．

ウィンドウプロシージャはコールバック関数の一種であり，OS がメッセージを伝える必要があるときに呼び出される．そしてウィンドウプロシージャは，内部に記述された switch 文に応じて処理を判定し実行する．この処理後にウィンドウプロシージャは終了し，次の呼び出しに備える．

ウィンドウプロシージャが行う実際の処理について，図 2 の例を基に説明する（この例は，図 1 に示したサンプルプログラムの中から選んだ）．図中の「IDC_BUTTON1」は，図 1 の「表示」ボタンが押された事に対応するメッセージである．これに対応して測定用スレッド関数（詳しくは，2 の「スレッド」の項を参照）が起動し，データ表示が行われる．そして，「IDC_BUTTON2」は「記録」ボタンが押された事に対応し，同様に測定用スレッド関数が起動し，データの表示と記録が行われる．最後に「IDCANCEL」は，ウィンドウ右肩の×印（閉じる）を押された事に対応し，終了処理を行った後にプログラム終了へと向かう．

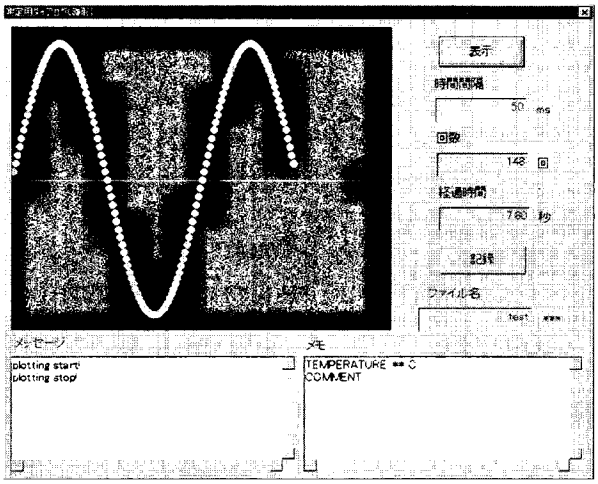

図 1　サンプルプログラムの実行画面．

```
case WM_COMMAND:
  switch(LOWORD(wp)){
  case IDC_BUTTON1:
    スレッド関数（表示）の起動
    return TRUE;
  case IDC_BUTTON2:
    スレッド関数（表示・記録）の起動
    return TRUE;
  case IDCANCEL:
    各種終了処理
    return TRUE;
  }
```

図 2　ダイアログプロシージャ（ダイアログボックス用のウィンドウプロシージャ）内の記述例．

2：スレッド 上では「受け取ったメッセージに応じて，ウィンドウプロシージャが実際の処理を行う」と述べた．しかし，その処理が複雑な場合がある（計測プログラミン

グは，正にその好例である)．このとき，その処理をウィンドウプロシージャ内に全て記述してしまうと，この処理を行っている間プログラムはメッセージに応答できなくなる（ボタンを押しても反応しない等)．

このような場合には，スレッド関数に処理すべき内容を記述し，ウィンドウプロシージャではその関数を呼び出すことだけを行う．スレッドとは，実行中のプログラムの中の1つの処理の流れのことであり，Windows環境では1つの実行プログラムの中に複数の流れ（マルチスレッド）を持つ事ができる[2-6]．ウィンドウプロシージャは，スレッド関数を呼び出した後に速やかに終了できるので，次のOSの呼び出しに備えることができる．一方，スレッド関数は，これを呼び出したウィンドウプロシージャが終了した後も，独立して測定を続けることが可能である．このスレッド関数は，内部に適切な終了条件を設定しておくことで，ユーザーの希望する機会に終了させることができる(終了条件の実装については，サンプルプログラムのソースコードを参照のこと)．

3. サンプルプログラムの概要

サンプルプログラムは，Visual C++ 6.0環境でコーディングした．クラスライブラリは使用せず，Win32 API (Application Programming Interface) 関数とC言語の標準ライブラリ関数を使用した．

プログラムの主な機能は，①「表示ボタン」を押すと正弦波形を計算し描画する，②「記録ボタン」を押すとこれらに加えて描画したデータをファイルに記録する，の二つである．「正弦波形の計算コード」を，各自が必要とする「計測機器の制御（データ取得）コード」に置き換えてやる事で，独自の計測プログラムに書き換えることができるので，大いに利用して頂きたい．また，その他に以下の機能も実装している．

- 高性能パフォーマンスカウンタを利用した計時機能
- プログラムからの情報をテキスト表示する「メッセージ画面（デバッグ，エラー発生時に有用)」
- 書き込んだテキストをファイル保存する「メモ画面（計測条件などの保存に利用可能)」

サンプルプログラムの使用方法については，冒頭で紹介したホームページより圧縮ファイル（measurement 0.2.zip）をダウンロードし，これを解凍して得られるテキストファイル（readme.txt）を参照して頂きたい．実行可能ファイル（measurement.exe）の動作とソースコードを比較する事で，Windowsプログラミングに関する知識が深まるものと期待している．

4. おわりに

最後のまとめに替えて，冒頭で紹介した参考文献の概要を説明しておく．文献2は，今回のプログラム作成で一番参考にした本である．また文献3は，関数の戻り値や引数の記述が見易く，スレッド・同期の記述が充実しており，文献2を補う形で参考にした．ファイル入出力，通信については，文献4に詳しい記述があり，この本と文献2，3を揃えれば，出発点として充分である．その後には，Windowsプログラミングの一般的知識ならば文献5，画像の取り扱いならば文献6を参照すればよい．

Windows環境においてC言語で自由に計測プログラムを構築する能力は，独自の新しい計測システムを開発するための，ひとつの大きな武器となるはずである．

参考文献

1) http://regulus.mtrl1.im.hiroshima-cu.ac.jp/~fujiwara/measurement/index.html
2) 土井滋貴，那須靖弘，上田悦子：Win32 API 完璧マスタ（CQ出版，東京，2001)．
3) 北山洋幸：技術者のための Visual C++実践プログラミング技法（技術評論社，東京，1999)．
4) Marshall Brain, Ron Reeves（訳：ドキュメントシステム)，Win32システムサービスプログラミング（ピアソン・エデュケーション，東京，2002)．
5) Herbert Schildt（監訳：山本信雄)，Windows2000プログラミング標準講座（翔泳社，東京，2000)．
6) Charles Petzold（訳：株式会社ロングテール，長尾高弘)，プログラミング Windows 第5版（上・下巻)，(アスキー，東京，2000)．

（広島市立大学情報科学部　藤原久志）