（仮称）十進BASICのLinuxへの移植

白　石　和　夫*

Porting Decimal BASIC to GNU／Linux

Kazuo SHIRAISHI

数学教育での利用を目的とし，Windows上のアプリケーションとして配布してきた(仮称)十進BASICをGNU／Linuxに対応させた．

Borland Delphiにより作成されたものであったので，Borland社のKylixを利用することでLinux版を作成することが期待されていた．しかし，Kylixの構造上，移植は単純作業ではない．まず，CLXアプリケーションに移行した上で，Windows APIを代替機能で置き換えなければならない．また，LinuxにはWindows Helpに相当する標準機能がないため，独自にヘルプ機能を用意する必要があった．

CLXアプリケーションへの移行で最も大きな影響を受けたのは，実時間マウス入力の機能である．また,WindowsのRich Editコントロールに依存した編集機能も影響を受けた．再帰処理のためにWindows APIを用いて独自に作成して用いていたメモリマネージャは，Linuxのmmapを利用することで代替とすることができた．

結果として，機能の制約はあるものの，教育用として十分使用に耐えるものが得られた．

１．はじめに

筆者は，マイクロソフトWindows上で動作する日本工業規格（JIS）Full BASIC準拠の言語処理系を作成し，数学教育での利用を主たる目的として配布してきた[i)ii)iii)]．筆者が知る限りにおいて，筆者が作成した処理系が最もJIS規格に近いものである．しかし，Full BASICの言語仕様のうち中核機能単位について“ほぼ完全”という状態になっているが，規格合致処理系ではない．完全なものが完成したときには何かよい名称を考えるつもりで，とりあえず「(仮称）十進BASIC」と名付けて配布してきたが，新課程の高校数学教科書の一部にも採用されるなど，利用が広がり，逆に名前の変更が難しくなってしまった．

（仮称）十進BASICはPC-9801 MS-DOS版から始まり,Windows3.1版を経て,現在の32ビットWindows版に至っている．本体はBorland社のTurbo PascalおよびDelphiで作成されている．Turbo PascalおよびDelphiはPascal系の言語であるが，アセンブリ語のコードを直接プログラム中に記述することが可能で，要所をアセンブラで記述することで処理速度をかせぐことができる．

近年，MS-Windowsに代えてOSにLinux（正式にはGNU／Linux）を採用する動きがでてきた．PC本体は，Windowsが動作するものと同じインテル486系のCPUを用いる場合がほとんどである．Borland社がDelphiのLinux版に相

* しらいし かずお　文教大学教育学部学校教育課程

当するKylixを出したことで，（仮称）十進BASICのLinuxへの移植の期待が高まってきた．（仮称）十進BASIC自体のバグもほぼ出尽くし安定してきたことと，Kylixも第3版となりKylixの側の不具合も解消されているであろうとの期待から，Kylixを利用して（仮称）十進BASICをLinuxに移植することを試みた．

なお，本稿では，特に断らないかぎり，「(仮称)十進BASIC」は，Delphi3で作成された32ビットWindows版を指す．

２．移植作業の実際

⑴　概　要

CPUはインテル486互換を前提とするので，KylixがDelphiと完全互換であれば，Windows APIを利用している部分を書き換えるだけで動作するはずであるが，実際にはそうではない．Borland社は，Delphi 6で，従来からのVCL（Visual Component Library）と呼ばれるアプリケーションフレームワークに加えて，Kylixと共通のCLX（Component Library for Cross Platform）と呼ぶアプリケーションフレームワークを追加した．CLX上のアプリケーションであれば，Kylixと互換性を持つというわけである．

移植作業はおよそ４つの段階に分けて実行した．第1段階は，Delphi 3のアプリケーションであったものをDelphi 6のアプリケーションに移行すること，第2段階は，CLXアプリケーションへの移行で，第3段階はWindows APIの削除，第４段階はKylixを利用したLinuxへの移植である．ただし，実際には，この４段階を行ったり来たりしての試行を何度も繰り返している．

⑵　Delphi 6への移行

（仮称）十進BASICは長期にわたりDelphi 3での開発で通してきた．利用するコンパイラのバージョンが変わることによる非互換が原因で思わぬ不具合が入り込むのを恐れたためである．しかし，CLXに移行するためには，まず，Delphi 6への移行というステップを置くことが望ましいと判断した．実際には，Delphi5への移行の後，Delphi 6への移行というステップになった．Delphi 5への移行の段階で問題が発生した．Delphiのオブジェクト内での変数の配置の仕方が変更になったため，アセンブリ語で書かれた処理が正しく実行できない不具合が出た．具体的には，複素数を拡張精度で保持するオブジェクト型の宣言にPackedを追加し，Delphi 3互換の配置となるように修正する必要があった．

なお，現在，公開されている(仮称)十進BASICの最新版（ver.5.1.12）はDelphi 6を用いたものになっている．

⑶　CLXアプリケーションへの移行

VCLアプリケーションからCLXアプリケーションへの移行は，フォームをテキスト形式で保存して拡張子を変更し，PASファイルでTで始まるユニット名のうちCLXで使えないものをQで始まるユニット名に修正する程度の書き換えで可能な場合がある．

しかし，（仮称）十進BASICは，プログラムの編集および保持の目的でDelphiのTRichEditを用いている．TRichEditはWindowsのRichEditコントロールのラッパーで，(仮称)十進BASICでは，Delphiが用意していない機能も含めてRichEditの機能を多く利用している．しかし，TRichEditはCLXでは省かれていた．そのため，VCLアプリケーションからCLXアプリケーションへの移行で最も面倒な作業となったのはエディタまわりの移植であった．エディタをTMemoに変更し，RichEdit独自の機能を用いている部分は削除した．たとえば，(仮称)十進BASICはデバッグ時に次に実行する行に色をつけて表示しているが，この機能は削除の対象となった．ここで，大きな問題となったのは，Windowsの編

集コントールでは文字位置をバイト単位で数えるのに対し,CLXでは文字位置を文字単位で数える場合があることであった．

もうひとつの大きな問題はマウスである．VCLアプリケーションではマウス・イベントは任意の時点で発生する．（仮称）十進BASICでは，それを利用してget-point文と，MOUSE POLLを実現している．MOUSE POLLはJISにはない独自の機能で，この文を実行するとその時点でのマウスの状態と位置を引数に指定した変数に返す．MOUSE POLLの機能は，描画ウィンドウのonMouseMoveのイベント処理で，マウスボタンの状態とマウスカーソル位置を記録し，MOUSE POLLの実行時にそのデータを読み出すことで実現している．CLXアプリケーションでは，マウスをクリックしないとMouseMoveのイベントが発生しないため，MOUSE POLLの機能が実現できない．また，get-point文においても，プログラム実行開始後，１回目のマウスクリックが無視される不具合がある．完全な対応のためには，BASICのプログラムの実行をメインスレッドから切り離して別スレッドにすればよさそうに思うが，CLXはスレッドセーフではないので，それを実行するのは容易ではない．

⑷ Windows APIの除去

Windows版(仮称)十進BASICは，一部にWindows APIを利用している．CLXアプリケーションではWindows APIも利用できるが，そのままではLinuxに移植できない．

Windows APIのうちのいくつかは，操作性を向上させるために利用している．この分については移植を断念してもやむを得ない．しかし，BASIC言語の動作のために不可欠なAPIもある．

（仮称)十進BASICは，漢字を含む文字列をプログラムに書くこと，また，文字列変数に漢字を含めることを許している．OS標準のテキストを利用するため，文字コードはいわゆるShift－JISであり，テキスト処理においては先頭から順に各バイトが多バイト文字の1バイト目であるかどうかを判断しながら読んでいく必要がある．多バイト文字の１バイト目であるかどうかの判断にWindows APIの関数isDBCSLeadByteを利用している．Delphi 6で追加された定数LeadBytesを利用して次の関数を定義することで，全体にわたるプログラムの書き換えを避けた．

```
function
isDBCSLeadByte(ch:byte):boolean;
  begin
     result:＝char(ch) in LeadBytes;
  end;
```

⑸ メモリー管理

（仮称)十進BASICは，利用者定義関数や副プログラムの実行時に局所変数を確保する．それらは，Delphiのオブジェクトであるが，通常の方法で初期化すると変数ごとにメモリーを確保するので効率が悪い．特に，配列変数の場合，それは顕著である．（仮称)十進BASICでは，配列変数のメモリー確保を合理化するために，独自のメモリーマネージャーを用意し，クラス関数NewInstanceとFreeInstanceをオーバーライドしてメモリー確保を合理化している．このメモリーマネージャーは，最後に確保したメモリーから順に解放されるという性質を満たす計算ルーチン，処理ルーチンにおいて共通に利用している．

メモリーマネージャー自体は単に上端アドレスを要求量に応じて加減算するだけのものであるが，アドレスが連続した領域であることが前提となっている．そのため，あらかじめ十分な大きさのアドレス空間を予約しておき，実際にメモリーが必要になった時点で実領域を確保する（コミットする）という構造になっている．これは，Windowsの持つ機能であるが，Linuxに対応する機能を見つけることができなかった．そのため，Linuxのメ

モリーマップの機能を利用して次のようにした．mmapで確保されたメモリーは，すぐには実メモリーに配置されないので，Windowsの仮想メモリーに近い効果がある．なお，seteexceptionとVirtualStackOverflowはbaseユニットで宣言されている．

```
unit vstack;

interface

function
   getmemory(size:integer):pointer;
   procedure freememory(size:integer);
   procedure InitMemory;

implementation

uses
   Libc, base ;

var
  StackBase:pointer;
  StackBottom:pointer;
  StackLimit:pointer;

function
getmemory(size:integer):pointer;
begin
  if LongInt(StackLimit)
   －LongInt(StackBottom)＜size then
    setexception(VirtualStackOverflow);
  GetMemory:＝StackBottom;
  Inc(Longint(StackBottom),size);
end;

procedure freememory(size:integer);
begin
  Dec(Longint(StackBottom),size);
end;

procedure InitMemory;
begin
  StackBottom:＝StackBase
end;

var StackSize:Cardinal＝$2000000; {32MB}

initialization

StackBase:＝mmap(nil, StackSize,
          PROT_READ or PROT_WRITE,
          MAP_PRIVATE or MAP_ANONYMOUS,
          0, 0);
 while StackBase＝MAP_FAILED do
    begin
       dec(StackSize, $4000000{64MB});
       StackBase:＝mmap(nil,
                     StackSize,
              PROT_READ or PROT_WRITE,
         MAP_PRIVATE or MAP_ANONYMOUS,
         0, 0);
    end;

StackBottom:＝StackBase;
 Cardinal(StackLimit):＝

Cardinal(StackBase)＋StackSize;

end.
```

⑹ 外部プログラムの起動

Windows版では，CHAIN文で外部プログラムを起動するためにWindows APIのShellExecExを利用している．Linuxに対応する機能を見つけることができなかったが，Linuxではvforkを利用して別プロセスを作るのが普通のやり方であるらしい．そこで，次のように書き換えて対応した．

```
uses LibC,base;
function
ShellExecNoWait(s1,s2:string):boolean;
var
   pid:integer;
begin
   result:＝true;
   pid:＝vfork();
   if pid＝0 then
      begin
        execlp(PChar(s1),
               PChar(s1),
               PChar(s2),
               nil);
        Libc._exit(－1);
      end
   else if pid＝－1 then
           setexception(OutOfmemory);
end;
```

⑺ 移植できなかった機能

JIS規格にある命令のうち，Linux版に移植できなかったものは，ASK CLIPとSET CLIPである．これらは，Windows版ではWindows APIを利用して実現されていたものであるが，Linux側に対応する機能を見つけることができなかったことと，存在させてもあまり使い道がないことで廃止とした．

また，GET POINTは，先述のように，機能に若干の不具合がある．

規格外の命令でLinux版で廃止となったものは，先に述べたMOUSE POLLの他，PAINT，FLOOD，ASOC PRINT，PLAYである．PAINTとFLOODはWindows APIを利用して実現されていることが理由で，ASOC PRINT，PLAYは，Linuxに関連付けの概念が存在しないのが理由である．

⑻ Linux上での作業

Windows上で仮想マシンソフトウェアのVMwareを導入し，VMwareにRedHat 9，Vein Linux，Fedora Core 1などの日本語に対応したLinuxをインストールし，その上にKylix3をインストールした．Celeron 800MhzのPCではVMware上のLinuxでKylix IDEを動作させるのは苦痛であったので，Linux上の操作の大半はKylixのコマンドラインコンパイラで行い，ソースの修正はWindows上のDelphi IDEを利用した．仮想マシンとの交信にはWindows上のFTPソフトFFFTP[iv]を用いた．FFFTPは文字コードの変換も行ってくれるので好都合であった．

⑼ 文字コードの問題

JIS規格ではアスキー7ビット文字の使用しか認めていないから，その範囲内であれば文字コードの問題は生じない．（仮称）十進BASICは広範囲に漢字の使用を認めているので，文字コードの違いは重大な問題である．

Linux版では，今のところ，EUCの2バイト文字までをサポートしている．EUCでは半角カナは２バイトなので，半角カナはとりあえず使える．ただし，PRINT USINGでの書式はバイト単位で適用するので２バイトの半角文字は意図に合わない結果になるかもしれない．EUCには３バイトの拡張漢字があるが，それは使えない．（仮称）十進BASICが2バイトのShift－JISを前提に書かれているため，3バイト文字を使えるように修正するのが面倒なためである．

Fedora Coreでは文字コードがUTF-8に移行している．KylixがUTF-8に対応していないの

で，Fedora Coreでは文字コードをEUCに設定しなおさないと正しく動作しない．とりあえず，そのことを利用者が意識しなくても使えるように，本体とは別にランチャーを用意し，そこで文字コードを指定してから本体を起動するようにした．単独で使う分には問題ないが，他のアプリケーションとデータをやりとりするとき，日本語を含む場合には文字コードの変換が必要になる．

⑽　ヘルプの移植

（仮称）十進BASICでは，Windows Helpを用いている．しかし，Linuxには，標準のヘルプ・システムがない．

Windows Helpは，ヘルプコンテキストと呼ばれる整数で表示されるヘルプを指定する仕掛けになっている．この仕掛け自体は意味付けが困難な整数を利用するという点で時代遅れのものであるが，BASIC本体にヘルプコンテキストが埋め込まれている関係上，そう簡単に書き直せるものではない．

結果的に，KylixのTTextBrowserを利用して，HTML形式のヘルプブラウザを作成し，ヘルプコンテキストからHTMLファイル名への変換は，インデックスファイルを利用して行う形をとった．なお，ヘルプシステムの作成には，Just-Great-Software.comのSimple HTML HelpViewer[v]が参考になった．

３．成果と課題

⑴　成　果

Windows版とほぼ同等な機能をLinux版で実現することができた．計算ルーチンなどは同一であるので，実行速度などはほとんど差がない．

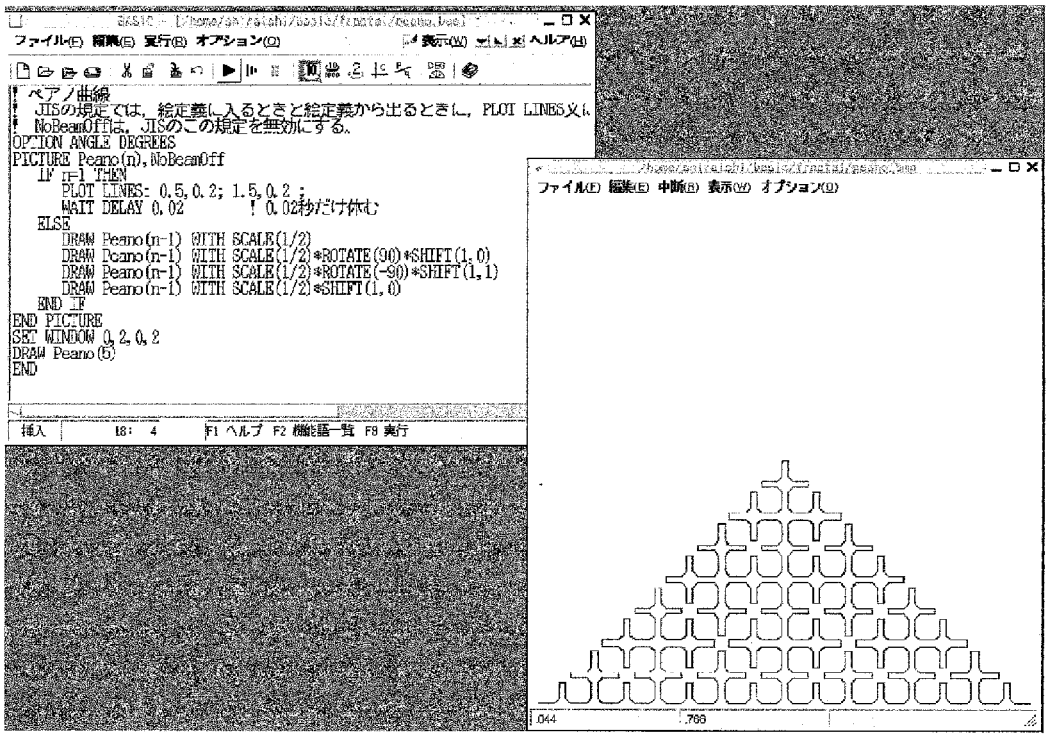

図　1

Fedora Core 1とVein Linux 2.6r3での動作を確認した．さらに，VMware(Windows版)上で，Red Hat Linux 9，Fedora Core 1，Fedora Core 2，Vine Linux 2.6r3，Plamo Linux 3.3，Turbo Linux 10D試用版，KNOPPIX3.2(VMware版)での動作を確認している．有料版のLinuxでの動作は試していないが，Turbo Linux 10D製品版での動作報告は得ている．また，Debian Linuxでの動作報告もあった．

作成したLinux版(仮称)十進BASICは，ソースコードとともに公開している．(仮称)十進BASICは，放送大学の科目「数学とコンピュータ」で使われている関係から，Linux版を利用する方もいたようで，移植の意義はあったと思う．

⑵ 問題点

DelphiのCLXとKylixを用いることでWindows版とLinux版を同時に開発できるが，CLXは機能が弱く，（仮称）十進BASICのWindows版をCLXに移行するわけにはいかない．そのため，ソースコードの共有も難しく，バグが見つかった場合の修正などはそれぞれごとに行わなければならない．

⑶ メニューの不具合

図１はLinux版の実行結果であるが，ファイルの読み込みを実行すると，この図のようにメニューバーの表示が乱れる．対策を試みているが，今のところ有効な対策が見出せていない．この現象はCLX版でも共通であることから，CLXのバグかも知れない．

参考文献

ⅰ）白石和夫，JIS準拠BASIC言語処理系の開発とその目的，日本数学教育学会会誌52-3，pp.8-15，日本数学教育学会(1998)

ⅱ）Kazuo Shiraishi，Development of a BASIC Language Processing System in accordance with ISO and its Aim，Proceedings of the Third Asian Technology Conference in Mathematics，pp.225-232，Springer(1998)

ⅲ）白石和夫，Windows環境におけるJIS Full BASICの実装，情報処理学会論文誌：プログラミングVol.41 No.SIG9(PRO 8)，pp.52-61（2000）

ⅳ） Sota，FFFTP，http:／／www2.biglobe.ne.jp／～sota／ffftp.html

ⅴ）Just-Great-Software.com，Simple HTML Help Viewer，http:／／www.helpscribble.com／linux.html