タイニーBASICをCで書く［立ち読み版］

本PDFドキュメントは『タイニーBASICをCで書く』（「原本のご紹介」参照）の冒頭37ページ分を原本どおりに抜粋したものです。「ご利用規約」をご了解いただいた上で、どうぞお楽しみください。

原本のご紹介

タイニーBASICをCで書く

パソコンでもマイコンでも走り、機能拡張し放題。

著者―鈴木哲哉
定価―3,200円＋税
判型/ページ数―B5変形/288ページ
ISBN978-4-8026-1020-9

ソシム株式会社　http://www.socym.co.jp/
〒101-0064 東京都千代田区猿楽町1-5-15 猿楽町SSビル3F 電話03（5217）2400（代表）

タイニーBASICをCで書く

パソコンでもマイコンでも走り、機能拡張し放題。

鈴木哲哉―著

ソシム

本書のサポートページ➲http://www.socym.co.jp/book/1020

本書に掲載したソフトウェアのプロジェクト、ソース、バイナリなどを提供しています。

［はじめに］

コンピュータの仕組みに関心がなく、ことさら優秀なプログラマを目指しているわけでもない人が、ひょんな事情でプログラムを書くことになったとしたら、おススメのプログラミング言語がBASICです。BASICは文法がコンピュータの仕組みに依存せず、プログラムは書いたそばから動き、もしどこかに大間違いがあってもそこまでは実行し、どうしようもなくなった段階で停止して継続できない理由を表示します。

本書は、その簡便なBASICで手っ取り早くコンピュータを動かすために、C言語でBASICを書きます。手短にいえば大半の時間をC言語で苦悶し、BASICの簡便さを味わうのはそのあとです。結局、手っ取り早いどころかとんだ回り道ですが、それはきっと楽しい道のりです。想像するところ、みなさんは少なからずコンピュータの仕組みに関心をもち、ことによっては優秀なプログラマを目指していると思うからです。

目標のBASICは、大型のBASICに比べたら小型の、いわゆるタイニーBASICです。比較的小型なので比較的簡素ですが、文法の目立った違いは「変数名が英1文字」というくらいです。これら多少の制約と引き換えに、構造がぐっとわかりやすくなって、本書がどうにか常識的なページ数におさまりました。ちなみに、変数名が英1文字だと困る長大なプログラムは、大型のBASICより、かえってC言語で書くほうが簡便です。

本書が思い描く状況はこんな感じです。ArduinoかRaspberry Piか、その種のマイコンボードを手に入れて、さしあたりC言語でLEDの点滅を成功させたとします。普通、これは単なる腕試しです。しかし、さらにタイニーBASICを乗せて誰でもLEDを点滅できるようにしたら話が違います。ドリカムのファンが5回点滅で「アイシテル」のサインを送り、第3級アマチュア無線技士はモールス符号で通信をするでしょう。

BASICの効能は、作る側が想定しなかったコンピュータの働きを、使う側が引き出してくれることにあります。その性質にあやかり、本書が書き手の実力を超えて、みなさんのお役に立てたら幸いです。たとえば、端々の説明がタイニーBASICに限らずプログラミングの参考になるかもしれません。あるいは、安眠を誘って充実した日常を創出するのだとしても、それは想定を超えており、文字どおり望外の喜びです。

著者しるす

［目次］

［第2章］移植編

個別の事情に対処する —— 61

62 内部の構造に少し立ち入る

ソースの成り立ち➲64

端末の制御➲76

乱数の生成➲90

移植の勘どころ➲96

Column【コラム】

［第3章］構造編

動作の不思議をなくす —— 109

110 キーテクノロジーをモノにする

中間コードの構造➲112

実行制御の仕組み➲134

式の構文解析➲154

一般の文の実行➲168

［第4章］拡張編

装置の個性を生かす——181

【関数】Functions

◉端末制御

◉標準機能

◉内部構造

［第1章］
基礎編

開発の足場を固める

実物で言語仕様を把握する

タイニーBASICは往年のパソコンが装備していたBASICに比べると小型で理解しやすい構造をもちます。しかし、OSのもとで「hello, world」を表示したり電子工作でLEDを点滅させたりするプログラムよりはずっと大型です。白紙の状態から完成まで、漏れなく丁寧に説明することは現実的でありません。そこで本書は、私が事前に完成させておいた豊四季タイニーBASICを実例とし、これを理解してもらう形をとります。

豊四季タイニーBASICは何ひとつ主張をしない無個性なタイニーBASICです。言語仕様はタイニーBASICの括りでいちばん有名なパロアルトタイニーBASICにならいました。もしかしたら名前が個性的だと感じるかもしれませんが、タイニーBASICの通例にしたがい先頭に地名を付けただけです。実例に打って付けですし、ゆくゆく独自のタイニーBASICを開発しようと決意したときには格好の雛形となります。

豊四季タイニーBASICはソースの一式とともに本書のサポートページで配布しています。この章では、その配布ファイルの取り扱いと豊四季タイニーBASICの使いかたを説明します。もしみなさんの関心が噂に聞いたタイニーBASICの実物を動かしてみたいだけなら話はここでおしまいです。そうなってしまっては困るので、機能を少し拡張したときどんな状況が生まれるのかについてもさりげなく言及しています。

この章の目的は実物の操作を通じて見た目の言語仕様を明らかにすることです。開発工程の学術的な分類に当てはめると要件定義から外部設計にあたり、次に内部設計へ進んで実現手段を検討します。今は操作に対する反応を見て、それがソースにどう記述されているのだろうかと想像してください。すぐに見当が付くところもあるでしょう。残りのぜんぜんわからない部分が、あとの章でのお楽しみというわけです。

タイニーBASICが文字の指示を解釈し、意図したとおりに動き、うっかりミスを的確に指摘する様子は、ときとして魔法使いの仕業に見えます。もちろん、そんなはずがありません。同じく、こういうものを書きあげるのに特別な才能がいるんじゃないかと考えたら思い違いです。タイニーBASICの動作は当たり前の処理を積み重ねた結果であり、それを書きあげる力量は、普通の努力を長く続けることで得られます。

RUN
R52
R51
R54
U1
J4
BROADCO
BCM2836RIF
TE1515 P10
500628-65
DISPLAY
R13
L1
L3
C100
C163
C7
U16
R55
R56
R15
Q5
C4
U3
2306KE
TDA4r
C3
C1
C162
C56
R14
Q2
U9
D5
C54
L2

SECTION

1 開発の方針

［第1章］
基礎編
開発の足場を固める

構造がわかりやすく移植や拡張が容易なBASICを目指す

BASICはダートマス大学のジョン・ケメニーとトーマス・カーツが1964年に完成させました。このいわゆるダートマスBASICはGE-235のタイムシェアリングシステムで動くコンパイラで、のちに普及したインタプリタとは構造が違います。ダートマスBASICが歴史に残した宝物は、実体ではなく、平易な文法と平易そうな名前でした。

BASICを実際に使った経験があるとしたら、たぶん往年のパソコンが装備していたマイクロソフトBASICだと思います。多数の文と関数をもつ大型のBASICですが、長大なプログラムを書く役割は、現在だとC言語が担っています。現在にいたってなおBASICに活躍の場が残されているならば、それはタイニーBASICの守備範囲です。

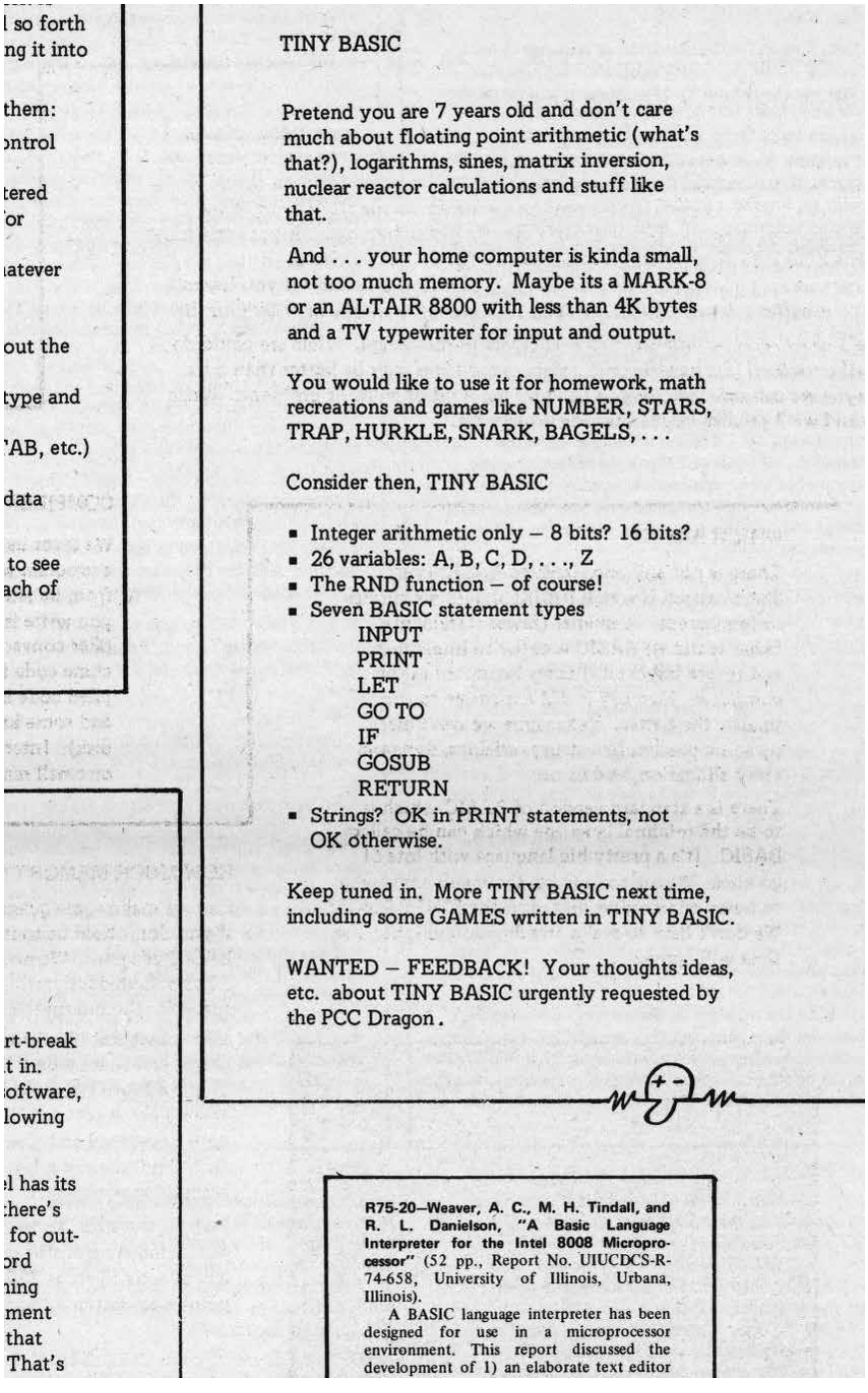
タイニーBASICはデニス・アリソンの指針に沿って小型化を目指したBASICの総称です。指針は1975年に発行されたピープルズコンピュータカンパニーの機関紙に掲載されました。原文を右に示します。ご覧のとおり機関紙は手作りです。ピープルズコンピュータカンパニーは、大きな組織のような名前を付けた、中規模の同好会です。

指針の内容を下にまとめます。凄く大雑把で、一見すると思い付きを並べたようですが、ひとつひとつ技術的な裏付けがあります。たとえば、変数を26個に抑えるとメモリの動的な割り当てが不要になって処理の手順が激減します。また、FOR ～ NEXTがないためGOSUB ～ RETURNとの間でスタックの調停をする必要がありません。

◉デニス・アリソンが策定した指針

要素	仕様
値	符号付き整数のみ（8ビットにするか16ビットにするかは思案中）
変数	A ～ Zの26個
関数	乱数を生成するRNDは当然いるよね
文	INPUT、PRINT、LET、GOTO、IF、GOSUB、RETURNの7個
文字列	PRINT文で使ってもいいがほかはダメ

◉ピープルズコンピュータカンパニーの機関紙に掲載されたデニス・アリソンの指針



l so forth
ng it into

them:
ontrol
tered
or
iatever
out the
type and
'AB, etc.)
data
to see
ach of

rt-break
t in.
oftware,
lowing
l has its
here's
for out-
ord
ning
ment
that
That's

TINY BASIC

Pretend you are 7 years old and don't care much about floating point arithmetic (what's that?), logarithms, sines, matrix inversion, nuclear reactor calculations and stuff like that.

And . . . your home computer is kinda small, not too much memory. Maybe its a MARK-8 or an ALTAIR 8800 with less than 4K bytes and a TV typewriter for input and output.

You would like to use it for homework, math recreations and games like NUMBER, STARS, TRAP, HURKLE, SNARK, BAGELS, . . .

Consider then, TINY BASIC

- Integer arithmetic only – 8 bits? 16 bits?
- 26 variables: A, B, C, D, . . ., Z
- The RND function – of course!
- Seven BASIC statement types
  - INPUT
  - PRINT
  - LET
  - GO TO
  - IF
  - GOSUB
  - RETURN
- Strings? OK in PRINT statements, not OK otherwise.

Keep tuned in. More TINY BASIC next time, including some GAMES written in TINY BASIC.

WANTED – FEEDBACK! Your thoughts ideas, etc. about TINY BASIC urgently requested by the PCC Dragon.

R75-20—Weaver, A. C., M. H. Tindall, and R. L. Danielson, "A Basic Language Interpreter for the Intel 8008 Microprocessor" (52 pp., Report No. UIUCDCS-R-74-658, University of Illinois, Urbana, Illinois).

A BASIC language interpreter has been designed for use in a microprocessor environment. This report discussed the development of 1) an elaborate text editor

小型化を目指した理由は当時のコンピュータに小さなメモリしかなかった（一説には大きなメモリが法外な値段で売られていることに反発した）からです。指針は、概略、こう呼び掛けています。「浮動小数点演算（何じゃそりゃ）、対数、三角関数、行列反転、核反応炉の計算はやらなくて、メモリが4Kバイト以下なら、こんな感じでどうよ」。

メモリにまつわる問題は時間が解決し、やがてタイニーBASICは役割を終えました。しかし、現在また必要性が生じています。電子工作で使うマイコンはメモリが当時のコンピュータと大差ない大きさです。そうでなくても、独自に追加した回路を動かすために、構造がわかりやすく、移植や拡張の容易なBASICが求められるのです。

本書のBASICは、そんな要求にこたえます。小型化はC言語で書くと限度があるため、数値目標を掲げず精一杯やることにして、どちらかというと構造のわかりやすさを重視します。そうすれば、あらかたのコンピュータでとりあえず動作し、もし高性能のコンピュータで物足りなく感じたとしても好きなように拡張することができます。

デニス・アリソンの指針に沿ったり既製のタイニーBASICと言語仕様を合わせたりしなければならない義理はありません。しかし、自由に作っていいというのは逆に掴みどころがなくて困ります。出来上がったものを説明するうえでも何かと不便です。現実的な判断として、既製のタイニーBASICからひとつを選び、お手本とします。

パロアルトタイニーBASICの外形的な働きをお手本とする

デニス・アリソンの指針が凄く大雑把なので、タイニーBASICはひとつひとつ言語仕様が異なります。通例ではそれをタイニーBASICの方言とみなし、先頭に開発者が住むところの地名を付けて呼び分けます。発表された順に紹介すると、テキサスタイニーBASIC、デンバータイニーBASIC、パロアルトタイニーBASICという具合です。

仮にマイクロソフトBASICを共通語とすれば、テキサスタイニーBASICとデンバータイニーBASICには強い訛りがあります。他方、パロアルトタイニーBASICは共通語で操作してもたまに「WHAT?」と表示する程度です。BASICだからこう使うのだろうという当てずっぽうがよく当たるので、本書のBASICはこれをお手本とします。

パロアルトタイニーBASICはリ・チン・ワンがIntel8080のアセンブリ言語で書き、ピープルズコンピュータカンパニーの機関紙から発展した『ドクタードブズジャーナル』の1976年第5号で発表しました。ほかの形で公表された事実は確認できないため、現在、インターネットで見付かるソースは本人以外の誰かが書き起こしたものです。

パロアルトタイニーBASICの冒頭の記述を右に示します。始めの10行で著作権を宣言しており、その内容は「@COPYLEFT ALL WRONGS RESERVED」となっています。これは一種のパロディで、定型的な著作権の宣言をさかさまに述べているため、議論の末、オープンソースのフリーウェアという解釈に落ち着いています。

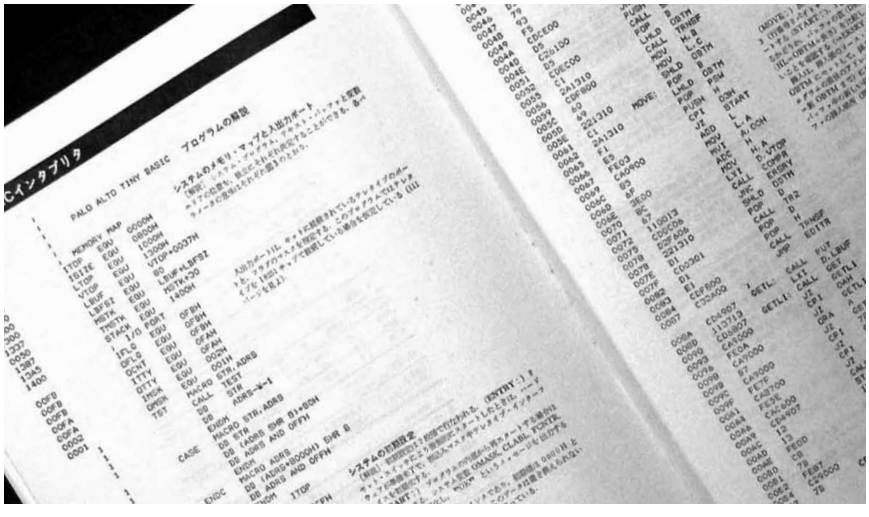
◉『ドクタードブズジャーナル』1976年第5号に掲載されたパロアルトタイニー BASIC（冒頭）

```
*****************************************************************
*
*                  TINY BASIC FOR INTEL 8080
*                       VERSION 1.0
*                    BY LI-CHEN WANG
*                     10 JUNE, 1976
*                       @COPYLEFT
*                   ALL WRONGS RESERVED
*
*****************************************************************
*
*   *** ZERO PAGE SUBROUTINES ***
*
*   THE 8080 INSTRUCTION SET LETS YOU HAVE 8 ROUTINES IN LOW
*   MEMORY THAT MAY BE CALLED BY RST N. N BEING 0 THROUGH 7.
*   THIS IS A ONE BYTE INSTRUCTION AND HAS THE SAME POWER AS
*   THE THREE BYTE INSTRUCTION CALL LLHH.  TINY BASIC WILL
*   USE RST 0 AS START OR RESTART AND RST 1 THROUGH RST 7 FOR
*   THE SEVEN MOST FREQUENTLY USED SUBROUTINES.
*   TWO OTHER SUBROUTINES (CRLF AND TSTNUM) ARE ALSO IN THIS
*   SECTION.  THEY CAN BE REACHED ONLY BY 3-BYTE CALLS.
*
                        ORG   X'0000'
0000 F3                 DI
```

パロアルトタイニー BASICは東京大学の小野芳彦がマクロ付きのアセンブリ言語でソースを書き直し、書籍や雑誌で紹介したことから、日本でもよく知られるところとなりました。紙面の一例を下に示します。先頭に「PALO ALTO TINY BASIC」と明記してあるのですが、世間では東大版タイニー BASICという名前が定着しています。

東大版タイニー BASICは、ソースが書き直されているものの、バイナリはパロアルトタイニー BASICと基本的に同じです。例外は、関数RNDがきちんとでたらめな値を返すように改良されていることだけです。したがって、東大版タイニー BASICをもとにパロアルトタイニー BASICの言語仕様を推し量ることができます。

◉『マイクロコンピュータのプログラミング』（共立出版）に掲載された東大版タイニー BASIC



パロアルトタイニーBASICの言語仕様は、のちほど本書のBASICと対照する形で説明します(➔38)。まずは要点だけ述べておくと、FOR ～ NEXTがあったり配列が使えたりしてデニス・アリソンの指針を大きく上回り、しかもサイズが2Kバイトを下回っています。一方、ソースは随所に奇妙な処理の手順があって解読に苦労します。

どうやらリ・チン・ワンは天才肌の変人で、普通の優れたプログラムに飽き足らず、極限まで小さくまとめる芸当に情熱を注いだようです。パロアルトタイニーBASICのほかにも数本、ホビイストから絶賛され、学者から酷評された、小型のプログラムを発表しています。そういう種類の頑張りかたは、C言語で書く場合、お手本となりません。

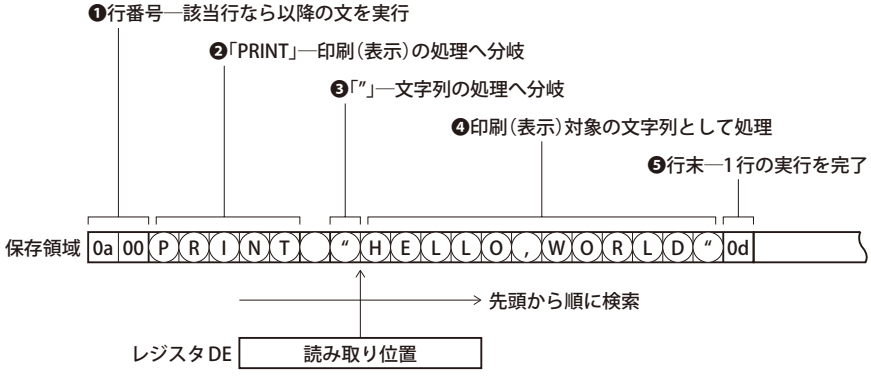
マイクロソフトBASICにならって中間コードを取り入れる

お手本のBASICとソースがある状況は、テスト用紙とカンペが一緒に配られたようなものです。あとはただ書き写せばいいわけですが、実際はそう簡単にいきません。カンペはアセンブリ言語で書かれ、解答欄はC言語で書く必要があります。双方の文法と作法に根本的な違いがあって、少なくとも機械的に翻訳することは不可能です。

C言語のプログラムは多数の小さな関数を積みあげる形で構成します。下の関数から上の関数へ渡せる情報(戻り値)はひとつなので、関数ごとにひとつ明確な機能をもちます。アセンブリ言語で伸び伸びと書かれ、縦横無尽に分岐するプログラムは、書き直しにあたり、いったん構造を整理し、必要に応じ、再構築しなければなりません。

最大の難関はプログラムを解釈、実行する仕組みです。パロアルトタイニーBASICは、下に示すとおり、プログラムを文字列で保存し、先頭から順に読み取り、キーワードが見付かったらそのつど対応する処理へ分岐します。途中に障害物競走のような厄介ごとが挟まったりはしませんから、一連の処理は一気に勢いよく実行されます。

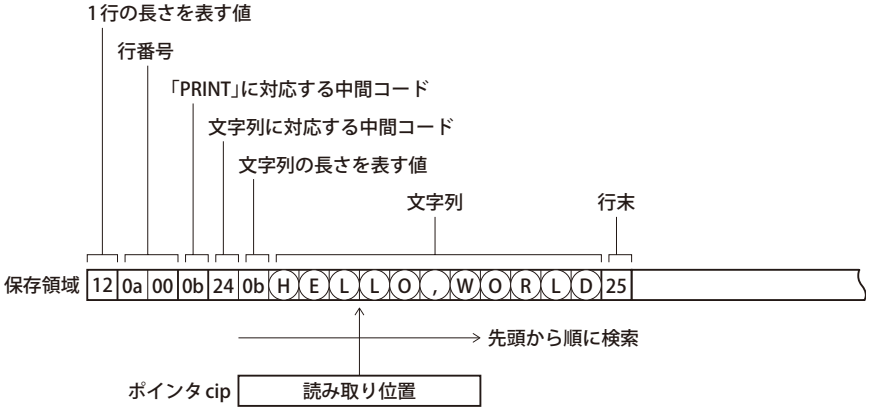
◉10 PRINT "HELLO,WORLD" が解釈、実行される仕組み



もしこの構造のままC言語で書き直すとすると、最初の関数はキーワードを見付けて持ち帰るあたりでひと区切りとなります。戻り値は、次に呼び出す関数を選ぶ都合で、キーワードそのものではなくキーワードに対応した値となるでしょう。だったら、プログラムを文字列で保存する構造を改め、中間コードを使うほうが合理的です。

プログラムを中間コードで保存した例を下に示します。キーワードがはじめから値となっているので（この例ではPRINTが0x0b、文字列が0x24）、キーワードを見付けて値を返す処理が丸ごと省略できます。さらに、以降の手順も大幅に簡略化されることが経験（事前に文字列のまま処理しようとしてみました）から判明しています。

◉10 PRINT "HELLO,WORLD"を中間コードで保存する場合の形式



ということで、プログラムを解釈、実行する仕組みに中間コードを取り入れます。同じ仕組みをマイクロソフトBASICが採用しています。本書のBASICは言語仕様をパロアルトタイニーBASICにならいますが、中間コードを取り入れるともうカンペの丸写しはできませんから、構造にマイクロソフトBASICの風味が加わります。

入出力装置は端末のみをサポートする

本書のBASICは、パソコン、マイコンボード、マイコンを応用した自作の装置で、そのまま動くか簡単に移植できるものとします。C言語で書く方針なので、CPUとメモリの違いはCコンパイラが埋めてくれます。困るのは、スイッチやLEDを含む広い意味での入出力装置が、あったりなかったり取り扱いの手順が違ったりすることです。

この問題は、言語仕様をパロアルトタイニーBASICにならうという名目で、強引に解決します。すなわち、入出力装置は端末だけをサポートし、あとは実態に合わせて各自で拡張してもらいます。端末の定義は下に示すテレタイプライタか同等の装置です。端末の制御は、通常、OSやライブラリが提供しますし、独自に書いても比較的簡単です。

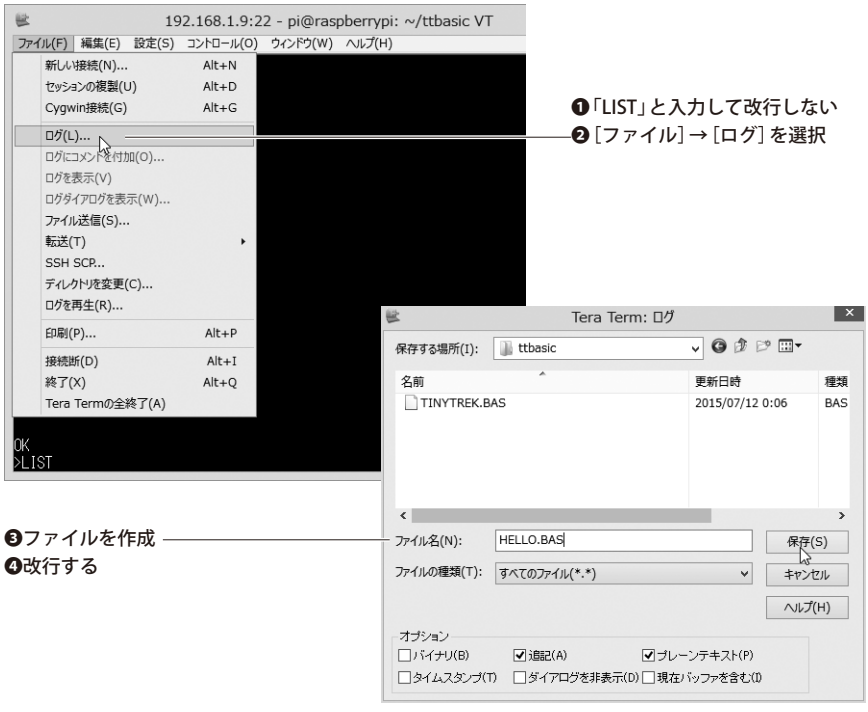
テレタイプライタは文字の入力と印字、紙テープのパンチと読み出しをします。これひとつで、BASICの最低限の操作ができます。たとえば、書きあげたプログラムをセーブやロードするためにもうひとつ記憶装置をサポートする必要はありません。リストを印字するかわりにパンチし、入力するかわりに読み出せばいいからです。

◉テレタイプライタ Model33ASR（参考）

Rama, CC-BY-SA-2.0-fr

現在の一般的な端末はパソコンの端末ソフトになるかと思います。これも、文字の入力や表示ができるのはもちろんのこと、紙テープのパンチや読み出しに相当する働きを備えます。TeraTermを例にあげると、右に示すとおり、ログをとることでリストの表示がファイルに保存され、それは、ファイル送信して入力することができます。

◉TeraTermでリストのログをとる手順



端末だけをサポートするBASICは、必要なことができて、余計なことをしない、ひとつの完成形です。BASICそのものに関心をもつ人は、ここを終着点とする選択肢があります。拡張してより多くの入出力装置をサポートする場合は、その雛形という位置づけになり、終着点は遥か彼方ですが、ここがいちばん重要な通過点です。

本書は目標のBASICが無事に完成するかどうかハラハラ、ドキドキしながら読んでいただく冒険物語ではありません。端末だけをサポートするBASICはすでに完成し、豊四季タイニーBASICという名前でサポートページに置いてあります。以降、その実際に動作するBASICで操作や文法を説明し、ソースを示して構造を明らかにします。

タイニーBASICの通例にしたがい、これまで「本書のBASIC」と呼んでいたものを、これからは「豊四季タイニーBASIC」と呼びます。「豊四季」（とよしき）は地名であり、それ以上の意味や主張はありません。ですから、みんながこれを自分のBASICと捉え、存分に利用し、大きく育てていただければありがたいことです。

SECTION

2 雛形の取り扱い

［第1章］
基礎編
開発の足場を固める

試用できて移植や拡張の原型となる4つの雛形

豊四季タイニーBASICは、概略、パロアルトタイニーBASICの働きをC言語で書き直した感じのBASICです。プラットフォームにCコンパイラがあればビルドできて、Windows版、Linux版、Arduino版、PIC24F版はビルドずみです。ビルドずみの4つは単なる既成品ではなく、移植や拡張の原型とする思惑があるので、「雛形」と呼びます。

雛形はサポートページからダウンロードしていただけます。概要を下にまとめます。配布ファイルは実行ファイルとソースを含みます。実行ファイルは基本的に当該のプラットフォームですぐ動きます。開発に使った機種とCPUが異なる場合は、もしかするとリビルドが必要かもれませんが、それは開発環境の機械的な操作で完了します。

拡張の作業を容易にするためソースはプラットフォームの開発環境が要求する形式にまとめてあります。たとえばWindows版のソースはVisualStudio Communityのプロジェクトです。一方、移植の作業では、どれかの雛形の純粋なソースだけを使います。それは開発環境の形式に紛れ込んでいて、どこにあるかは雛形ごとに異なります。

◉サポートページで配布している雛形

種別	配布ファイル	開発に使った機種	開発環境
Windows版	ttb_win.zip	64ビットWindows10	VisualStudio Community
Linux版	ttb_lin.tar.gz	Raspberry Pi 2 Model B	GCCツールチェイン
Arduino版	ttb_arduino.zip	Arduino Uno	Arduino IDE
PIC24F版	ttb_pic24f.zip	PIC24FJ64GA002自作機	MPLAB X IDEとXC16

雛形はCPUとメモリと端末だけで動く簡素な構造をもち、技術的な成り立ちがコンピュータの進歩に影響を受けません。今後、コンピュータの世界で何かしら革新的な出来事が起きたとしても、おそらく対応する必要がないので、バージョンアップする予定がありません。豊四季タイニーBASICの将来はみなさんの手の中にあります。

豊四季タイニーBASICの著作権の取り扱いは、パロアルトタイニーBASICにならって「@COPYLEFT ALL WRONGS RESERVED」としたいところですが、悪ふざけが過ぎると思うので、公式にはGNU General Public Licenseです。すなわち、いっさいの保証をしないという条件で、自由な実行、複写、配布、研究、変更、改良を認めます。

Windows版の実行ファイルとソース

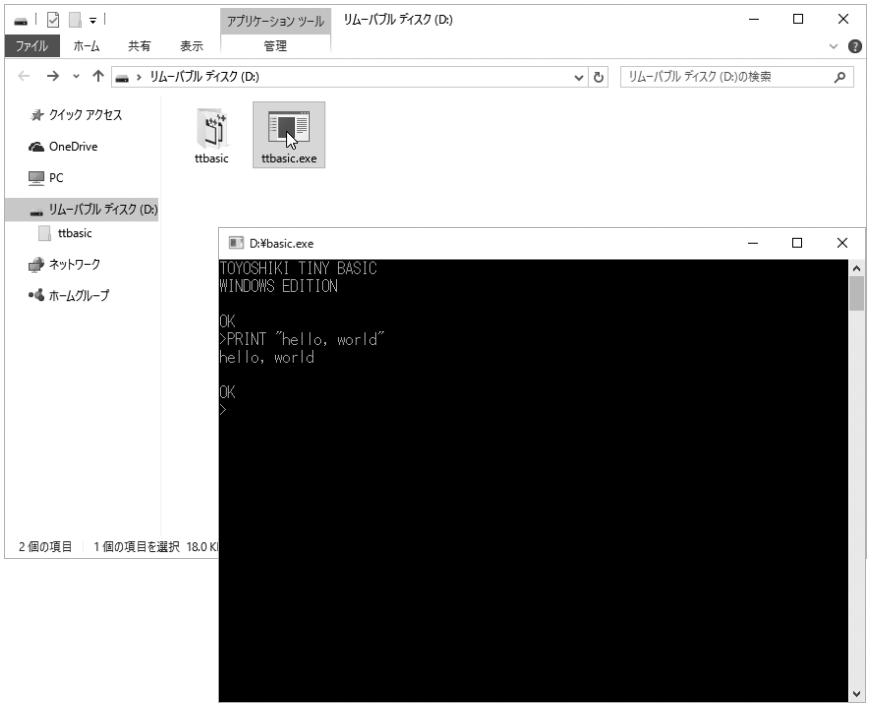
Windows版は最初に完成してほかの雛形の雛形になりました。簡便なBASICを開発するうえで、Windowsのパワーは不可欠でした。しかし、Windowsが簡便なBASICを必要としているようには思えません。完成したWindows版の役割は、いちばん普及したプラットフォームで雛形を試用し、移植の構想を練ってもらうことにあります。

配布ファイルに含まれる主要なファイルを下に示します。実行ファイルはttbasic.exeです。事前のインストールなしに実行できて、不要になったときのアンインストールもいりません。x86の指定でビルドしてありますが、現役のすべてのWindowsにおいて、32ビットと64ビットの両方で動作することを確認しています。

◉Windows版の配布ファイルに含まれる主要なファイル

Windows版のソースを取り扱う場合、VisualStudio Communityでttbasic.slnを開いてください。開発終了時点のプロジェクトが再現され、そこから継続できます。たとえばx64を指定してリビルドすることができます。移植する場合はbasic.cppを使います。通常、それを新しいプラットフォームへもっていって新しい開発環境で書き直します。

◉Windows版の実行例



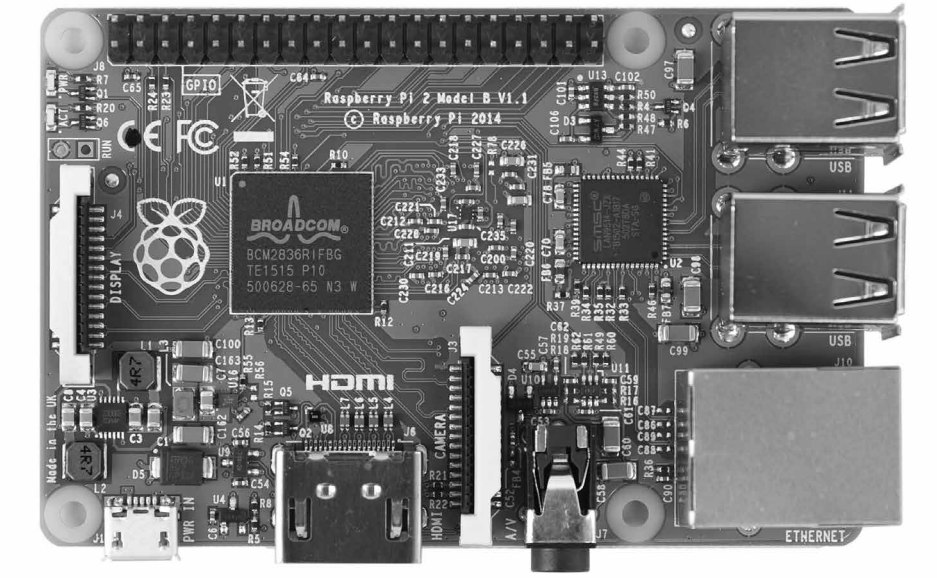
Windows版の実行例を上に示します。端末ソフトで接続する必要はありません。ということはプログラムを保存する手段がないわけで、これは欠点です。美しく終了する方法も用意しておらず、操作を終えるときはウィンドウの［×］(閉じる) ボタンをクリックします。これら不便なところを改善すると、移植の作業が不便になります。

Linux版の実行ファイルとソース

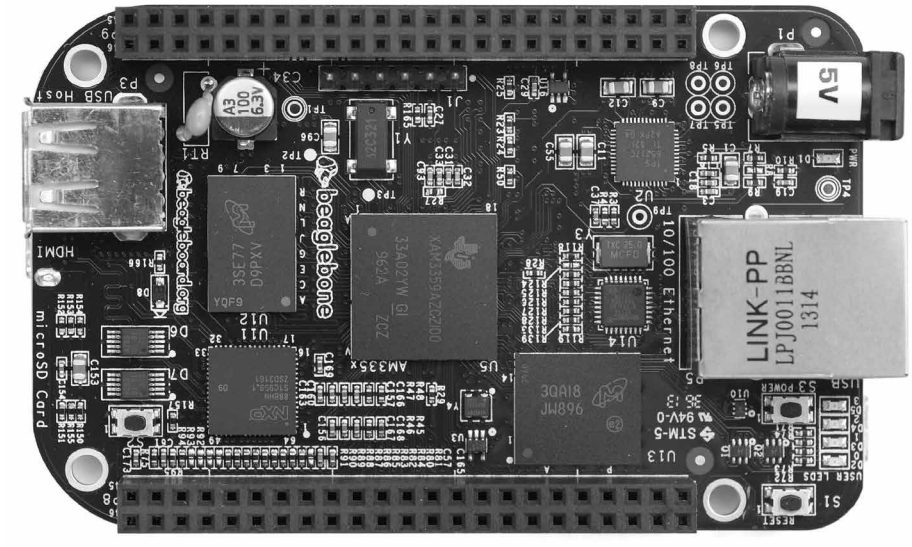
Linux版はRaspberry Pi 2 Model BとRaspbianの組み合わせで開発しました。開発環境はRaspbianをはじめ多くのLinuxで動くGCCツールチェインです。Raspbianでビルドすると実行ファイルは32ビットのARM対応になります。予想どおり、それはBeagleBone BlackとDebianの組み合わせでもリビルドなしに動作しました。

Raspberry Pi 2 Model BとBeagleBone Blackの外観を右に示します。Linux版の主要なターゲットはこれらハイエンドの一群に属するマイコンボードです。ネットワークとUSBと、電子工作に適した拡張ピンを備え、高度な入出力装置が作れます。雛形を拡張してそれらをサポートすれば、ハイエンドのタイニーBASICが出来上がります。

◉Linux版がそのまま動くマイコンボードの代表例



Raspberry Pi 2 Model B（CPU—BCM2836/ARM Cortex-A7 quad core/900MHz、RAM—1Gバイト）



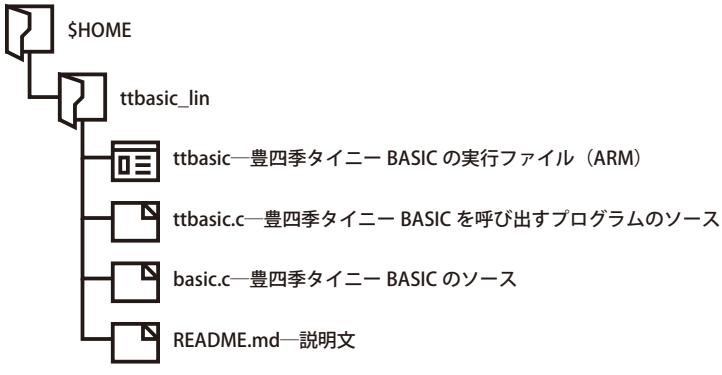
BeagleBone Black（CPU—AM335x/ARM Cortex-A8/ 1GHz、RAM—512Mバイト）

◉Linux版をクローニングする操作（数字の表示は実際と異なる場合があります）

```
pi@raspberrypi ~$ git clone https://github.com/vintagechips/ttbasic_lin.git
Cloning into 'ttbasic_lin'...
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
pi@raspberrypi ~ $
```

Linux版はサポートページともうひとつGitHubにも置いてあります。サポートページからダウンロードして展開するより、GitHubをクローニングするほうが簡単です。Linuxへパソコンの端末ソフトで接続してクローニングした例を上に示します。この操作でttbasic_linディレクトリが作成され、下に示すファイルが展開されます。

◉Linux版を構成するファイル



実行ファイルはttbasicです。CPUが32ビットのARMなら、これをそのまま実行することができます。Raspberry Pi 2 Model BのRaspbianで実行した例を右に示します。ttbasic_linディレクトリへ移動し（操作❶）、「./ttbasic」と入力します（操作❷）。Linux版に限り、終了してLinuxに戻るSYSTEM命令があります（操作❸）。

◉Raspberry Pi 2 Model BのRaspbianで実行した例

```
pi@raspberrypi ~ $ cd ttbasic_lin ――❶
pi@raspberrypi ~/ttbasic_lin $ ./ttbasic ――❷
TOYOSHIKI TINY BASIC
LINUX EDITION

OK
>PRINT "hello, world"
hello, world

OK
>SYSTEM ――❸
pi@raspberrypi ~/ttbasic_lin $
```

CPUが32ビットのARMでない場合はリビルドが必要です。コマンドはttbasic.cの先頭のコメントに記述してあり、必要に応じ、コピーとペーストができます。amd64版Debianでリビルド（操作❶）してから実行した例を下に示します。この例はデスクトップ環境の端末で操作しています。まったく違うCPUで、まったく同じに動作しました。

◉amd64版Debianでリビルドしてから実行した例

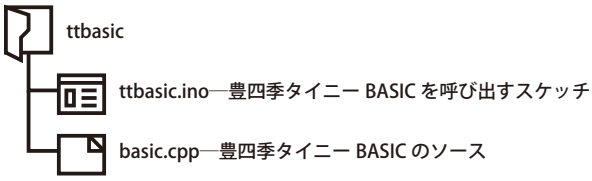
Linux版はLinuxの伝統的なAPIのみを使っており、理論上、すべてのディストリビューションでリビルドに成功します。一方、その伝統的なAPIが端末の取り扱いに面倒な手順を求めるため、ソースは先頭のほうがやや込み入っています。ほかのプラットフォームに移植する場合、Linux版のソースを流用することは賢明といえません。

Arduino版のスケッチとソース

Arduino版はRAMが2Kバイト以上あるArduinoが対象です。CPUは、AVR系でもARM系でもインテル系でも大丈夫です。全部のArduinoを動作確認することは経済的に無理なので、標準の開発環境Arduino IDEで対象のArduinoを検証したところ、警告ひとつなく成功しました。逆に、対象外は「Arduino NG or older」くらいです。

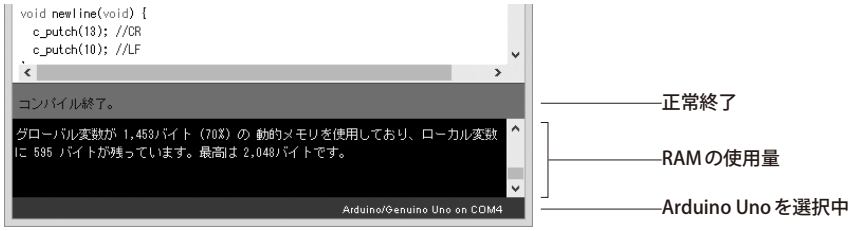
配布ファイルは下に示すファイルを含みます。これをttbasicフォルダごと所定の位置（Windowsだとドキュメントの Arduinoフォルダ）へ移動すると Arduino IDEの一覧にttbasicが追加され、［開く］ボタンで開けます。構造上、実行ファイルは存在しません。ソースは簡潔で、移植の原型として、Windows版とともに有力な選択肢です。

◉Arduino版の配布ファイルに含まれるファイル



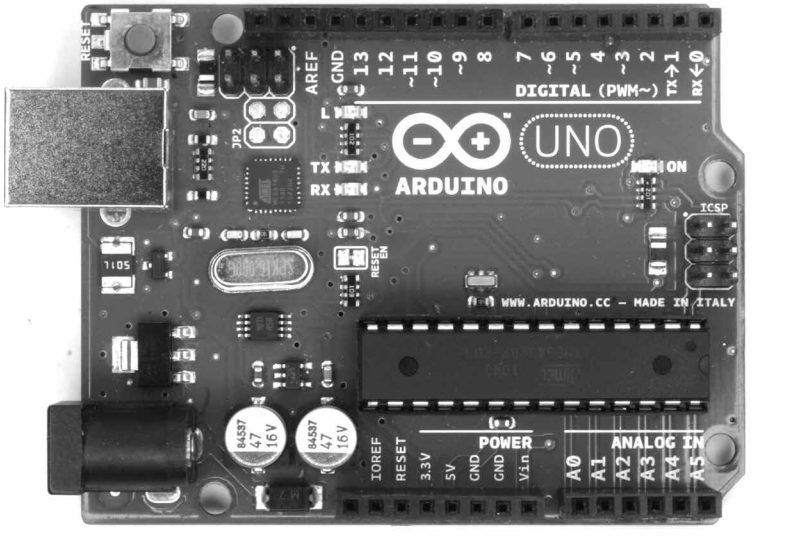
豊四季タイニーBASICの特徴が生きるのはArduino Unoとの組み合わせです。下に示すとおり、雛形だけでRAMの70%を占め、拡張したら底を突きかねない状況です。いいかえると、Arduino Unoでまともに動くBASICは豊四季タイニーBASICくらいしかありません。双方にとって過不足がなく、理想の相方ということができます。

◉Arduino IDEでArduino Unoを選択して検証した結果

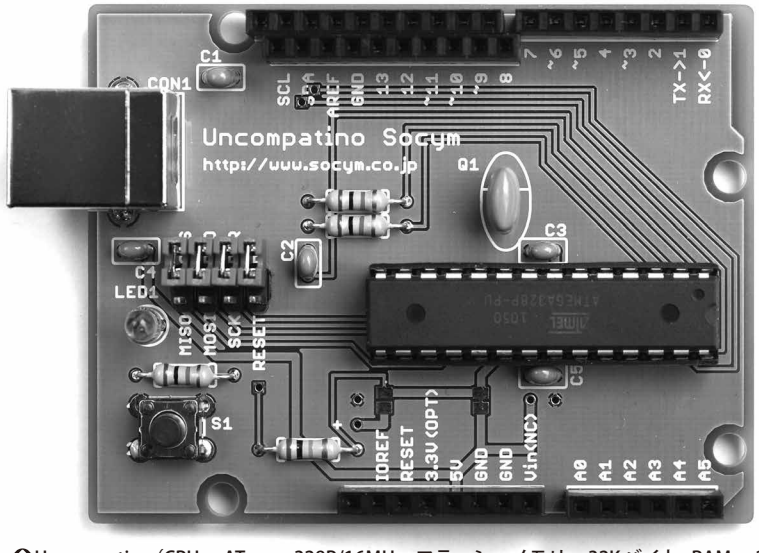


Arduino版の動作確認では下に示すArduino Unoとその互換機を使いました。電子工作で事実上の標準を確立したマイコンボードですから、作りためたシールドがあると思いますし、そうでなくてもインターネットに応用例がたくさん見付かります。それらが、雛形に残された30%の拡張の余地をきっちりと埋めてくれることでしょう。

◉Arduino版の動作確認をしたマイコンボードの代表例

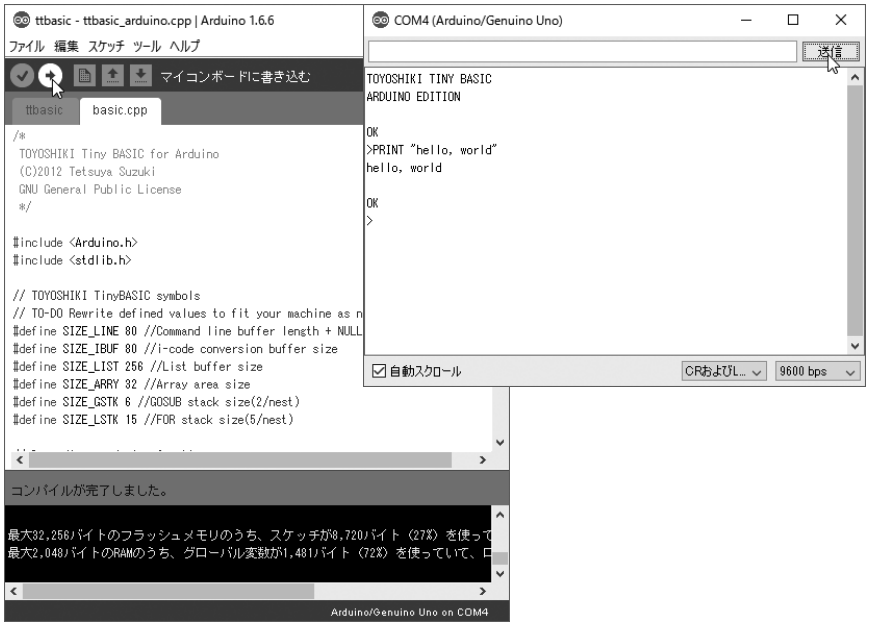


Arduino Uno（CPU―ATmega328P/16MHz、フラッシュメモリ―32Kバイト、RAM―2Kバイト）



Uncompatino（CPU―ATmega328P/16MHz、フラッシュメモリ―32Kバイト、RAM―2Kバイト）

◉Arduino版の実行例



Arduino版の実行例を上に示します。本来は端末で接続するのですが、動作確認なのでArduino IDEのシリアルモニタを使っています。通信速度は9600bps、改行はCR+LFです。これらの仕様はのちほど述べる手順でカスタマイズできます（➲54）。美しく終了する方法はありません。Arduinoにはそもそも終了という概念がありません。

PIC24F版のHEXファイルとソース

PIC24F版は、ほかの雛形と比べてちょっと不便です。対象はマイクロチップテクノロジーのマイコンPIC24FJ64GA002で、配線が苦手な人は動かすことさえままなりません。ソースは整理されていなくて、少なくとも移植の原型には不向きです。こういう雛形がひとつ混じっているのは、たとえていえば記念碑のようなものです。

Windowsで簡便なBASICの開発を始め、まだうまくいく見とおしが立っていなかったころ、移植や拡張の可能性はいちばん安上がりなPIC24FJ64GA002で検証しました。この雛形は、Windows版から移植し、初めて文と関数の拡張に成功したとき、以降の開発でもし間違っても後退しないように、あるがままの姿を残したものです。

とはいえ、記念碑には記念碑なりの意義があります。この雛形は試験的に拡張した機能を残していて、下に示すとおり、セーブとロード、LEDの点滅、スイッチの読み取りができます。製作物におけるこれらの働きは、ほかの雛形を選んだ人にも参考になると思います。面倒くさいことが嫌いな人にかわり、すぐあとで製作例を紹介します。

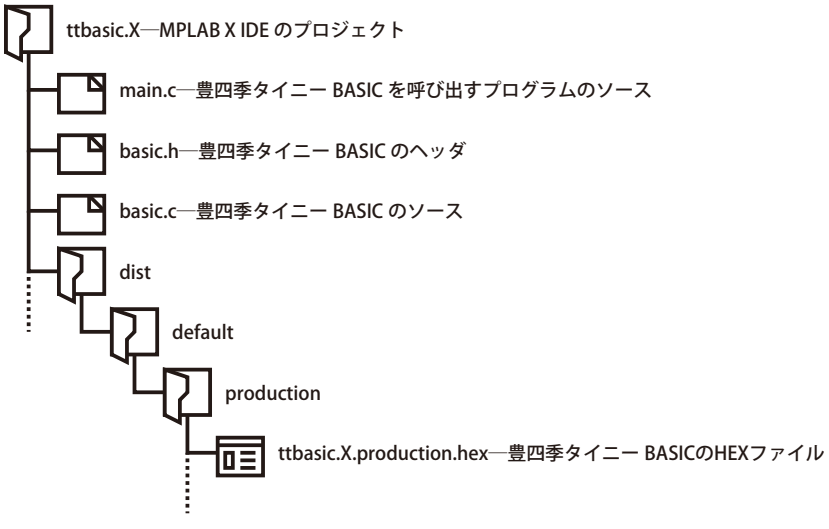
◉PIC24F版で拡張ずみの機能

書式	機能
SAVE [BOOT]	プログラムを[パワーオンランの指定で]セーブする
LOAD	プログラムをロードする
LED《式》ON|OFF	《式》で指定したLEDをON（点灯）|OFF（消灯）する
SW()	戻り値のビットとスイッチが対応。押していると1、離していると0

配布ファイルに含まれる主要なファイルを下に示します。ttbasic.Xフォルダと下層のファイルの一式がMPLAB X IDEのプロジェクトを構成します。PIC24FJ64GA002に書き込むのはttbasic.X.production.hexです。書き込み装置はPICKit2またはPICKit3を想定しています。両方を持っていたら、ぜひ古いほうのPICKit2を使ってください。

PIC24F版をいじり回して使うことはおススメしません。しかし、PIC24FJ64GA002は今もって人気の高い端正なマイコンであり、たまたま取り扱いに習熟しているとしたら、その限りではありません。MPLAB X IDEでttbasic.Xフォルダを開くとプロジェクトが再現され、開発を継続して、ttbasic.X.production.hexを更新できます。

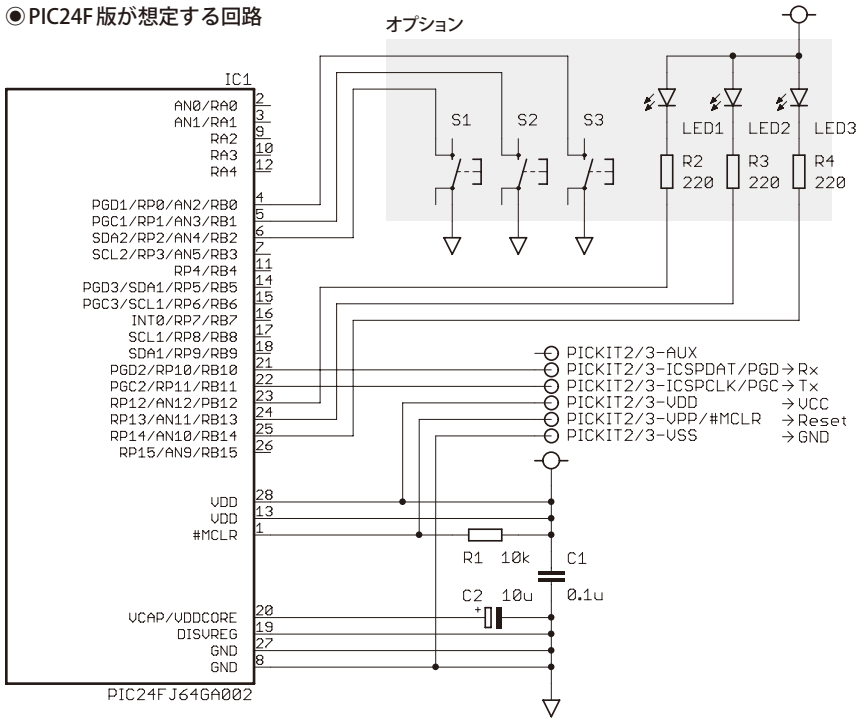
◉PIC24F版の配布ファイルに含まれる主要なファイル



PIC24F版が想定する回路を下に示します。スイッチとLEDは、お好みで取り付けてもいいオプションです。必須の部品は、PIC24FJ64GA002、抵抗1本、コンデンサ2本です。普通はさらに電源と端末のインタフェースを取り付けることになりますが、もし書き込み装置がPICKit2ならそれで兼用できるため、必須とまではいえません。

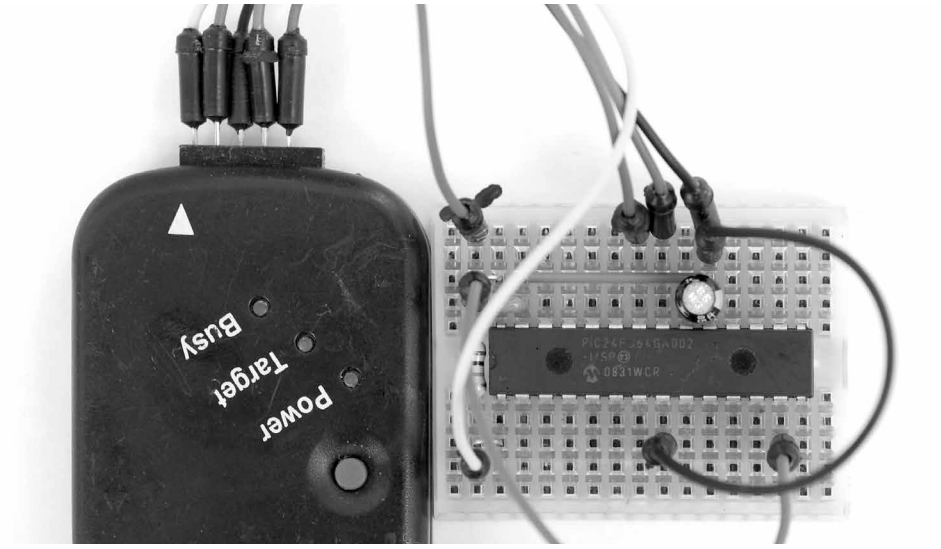
PICKit2やPICKit3の接続先は、書き込みを完了すると、電源（VCCとGND）、リセット（Reset）、端末の送信（Tx）と受信（Rx）に割り当てられます。PICKit2は、接続したまま電源を供給し、端末のインタフェースとして働きます。PICKit3を使う場合は、シリアル-USB変換ケーブルなどでパソコンと接続し、端末ソフトで操作してください。

◉PIC24F版が想定する回路



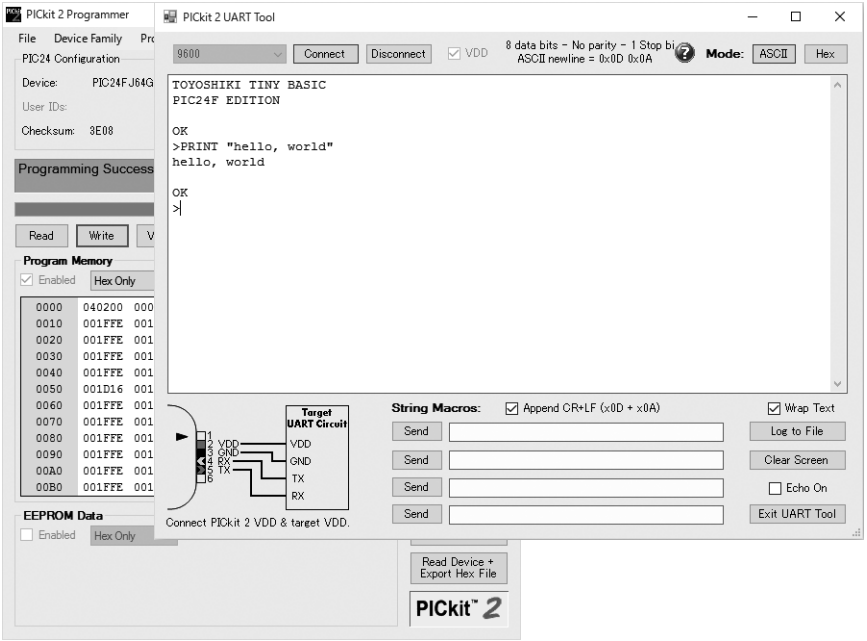
ブレッドボードに組み立てた最小の構成を右に示します。書き込み装置がPICKit2なのでttbasic.X.production.hexの書き込みから端末の操作まで、接続を変更する必要がありません。ソースを書き換えてもリビルドしてすぐ動作確認ができます。かつて移植や拡張の可能性を検証したとき、能率的に作業できてとても助かりました。

◉PIC24F版が動作する最小の構成



PIC24F版の実行例を下に示します。通信速度は9600bps、改行はCR+LF、Echo Onは不要（チェックを外す）です。電源を入れてから端末を接続するという手順になるので起動メッセージは見られないかもしれません。その場合も、1回、改行すればプロンプトが表示されます。なお、この端末は［BackSpace］キーに正しく反応しません。

◉PIC24F版の実行例

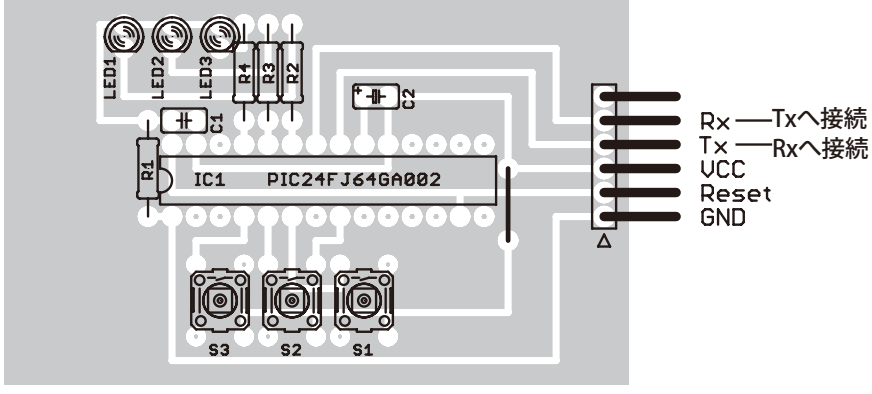


もぐらたたきとバーサライタの製作例

引き続き、スイッチとLEDを取り付けてPIC24F版の全部の機能が使える状態で動かしてみます。記憶装置を取り付ける必要はありません。プログラムはフラッシュメモリにセーブされます。それでも、部品点数がこのくらいになるとブレッドボードよりユニバーサル基板に組み立てるほうが便利です。配線図と部品表を下に示します。

PICKit2またはPICKit3をピンヘッダに接続し、ttbasic.X.production.hexを書き込みます。このあとピンヘッダは、電源、リセット、端末の接続先になります。書き込み装置がPICKit2ならそれだけですみますが、PICKit3を使っている場合は、書き込んだあと取り外し、電源と端末のインタフェースを接続する必要があります。

◉配線図と部品表



部品番号	仕様	数量	備考
IC1	PIC24FJ64GA002-I/SP	1	マイコン
LED1 ～ LED3	OS5RKA3131A	3	LED
R1	10kΩ	1	1/4Wカーボン抵抗
R2 ～ R4	220Ω	3	1/4Wカーボン抵抗
C1	0.1μF	1	積層セラミックコンデンサ
C2	10μF	1	電解（またはタンタル）コンデンサ
S1 ～ S3	小型タクトスイッチ	3	製作例はDTS-6（Cosland）
―	DIP28ピンICソケット	1	汎用部品
―	6ピン1列L型ピンヘッダ	1	40ピンをカットして使用
―	ユニバーサル基板	1	製作例はCタイプ（秋月電子通商）

この製作物で走るもぐらたたきのプログラムを下に示します。どれかのLEDが短く光りますから、その間に対応するスイッチを押します。成功すると全部のLEDが3回点滅（万歳三唱）します。ゲームが苦手でも3回点滅を確認できるよう必勝法を明かしておきます。どれかのスイッチを押し続け、対応するLEDが光るのを待ってください。

◉もぐらたたきのプログラムMOGURA.BAS

```
100 REM Mogura Tataki
110 REM --------
120 @(0)=1
130 @(1)=2
140 @(2)=4
150 REM --------
160 FOR I=1 TO 3
170 LED 0 ON; LED 1 ON; LED 2 ON; GOSUB 1000
180 LED 0 OFF; LED 1 OFF; LED 2 OFF; GOSUB 1000
190 NEXT I
200 REM --------
210 FOR I=1 TO 8
220 GOSUB 1000
230 NEXT I
240 REM --------
250 W=0
260 L=RND(3)-1
270 LED L ON
280 FOR I=1 TO 10000
290 IF SW()=@(L) W=1
300 NEXT I
310 REM --------
320 IF W=1 GOTO 160
330 LED L OFF
340 GOTO 210
350 REM --------
1000 REM 150mS
1010 FOR D=1 TO 10000
1020 NEXT D
1030 RETURN
```

120 @(0)=1 / 130 @(1)=2 / 140 @(2)=4 ―― 配列にLEDとスイッチの対応を記録

160 FOR I=1 TO 3 ―― ❶全部のLEDを3回点滅

210 FOR I=1 TO 8 / 220 GOSUB 1000 / 230 NEXT I ―― ❷しばらく待機

260 L=RND(3)-1 / 270 LED L ON ―― どれかのLEDを点灯

290 IF SW()=@(L) W=1 ―― もしスイッチと一致したら成功

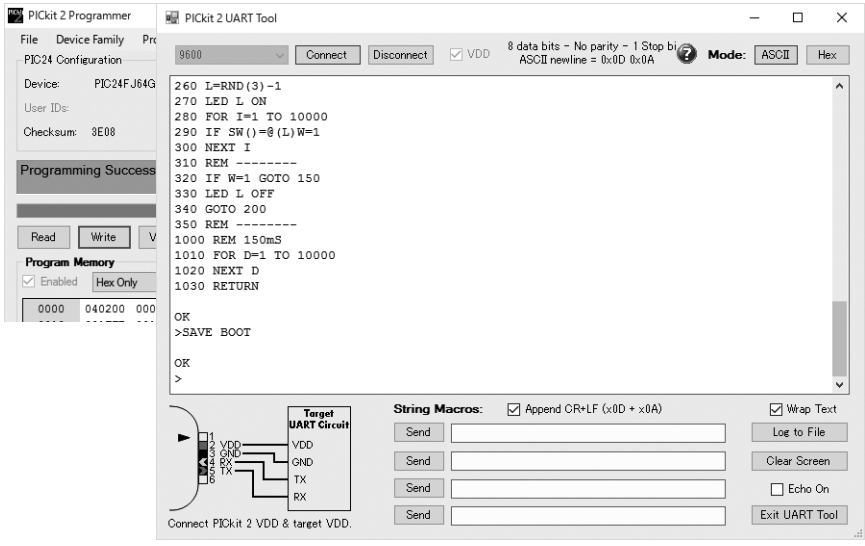
320 IF W=1 GOTO 160 ―― もし成功なら❶から繰り返す

330 LED L OFF / 340 GOTO 210 ―― LEDを消灯して❷から繰り返す

1000 REM 150mS / 1010 FOR D=1 TO 10000 / 1020 NEXT D / 1030 RETURN ―― 約150m秒待機するサブルーチン

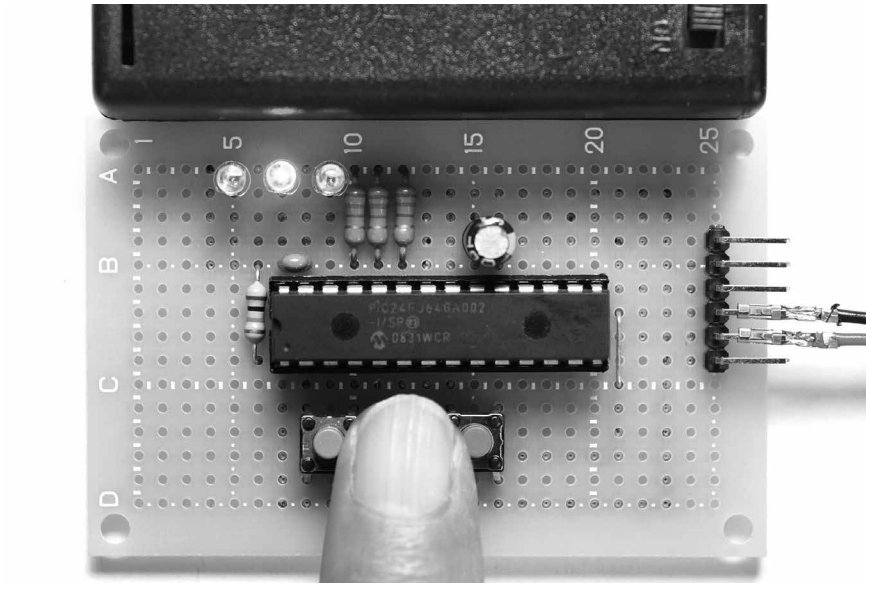
もぐらたたきに必要な入出力装置はスイッチとLEDです。端末はプログラムの入力やデバッグの段階で使いますが、ひとたび動き出したらもういりません。ですから、完成したプログラムをパワーオンランの指定でセーブしておくと、ただ電源をつなぐだけで端末なしにもぐらたたきが始まり、パソコンから離れたところで遊べます。

◉パワーオンランの指定でセーブした例



パワーオンランの指定でセーブした例を上に示します。このプログラムは電源を入れるかリセットすると自動的にロード、実行されます。単三乾電池2本入りの電池ボックスを接続し、端末は接続しないで、もぐらたたきに興じている様子を下に示します。必勝法で見事にもぐらをたたきましたから、このあと全部のLEDが3回点滅します。

◉もぐらたたきの実行例



目先をかえてもうひとつ、バーサライタのプログラムを下に示します。どれかのスイッチを押すと3本のLEDが一見でたらめに点滅します。この状態で製作物を左から右へ振ると、空中に0～9の数字が描かれます。製作物に端末がつながっていたら振れません。パワーオンランの指定でセーブし、端末を取り外して、電池で動かします。

◉バーサライタのプログラムVWRITER.BAS

```
100 REM Versa Writer
110 REM --------
120 @(0)=7;@(1)=5;@(2)=5;@(3)=5;@(4)=7
130 @(5)=6;@(6)=2;@(7)=2;@(8)=2;@(9)=7
140 @(10)=7;@(11)=1;@(12)=7;@(13)=4;@(14)=7
150 @(15)=7;@(16)=1;@(17)=7;@(18)=1;@(19)=7
160 @(20)=4;@(21)=5;@(22)=7;@(23)=1;@(24)=1
170 @(25)=7;@(26)=4;@(27)=7;@(28)=1;@(29)=7
180 @(30)=7;@(31)=4;@(32)=7;@(33)=5;@(34)=7
190 @(35)=7;@(36)=1;@(37)=1;@(38)=1;@(39)=1
200 @(40)=7;@(41)=5;@(42)=7;@(43)=5;@(44)=7
210 @(45)=7;@(46)=5;@(47)=7;@(48)=1;@(49)=7
220 REM --------
230 IF SW()=0 GOTO 230
240 FOR N=0 TO 9
250 GOSUB 800
260 NEXT N
270 GOTO 240
280 REM --------
800 FOR I=0 TO 4
810 GOSUB 900+@(N*5+I)
820 NEXT I
830 LED 3 OFF
840 GOSUB 1000;GOSUB 1000
850 RETURN
860 REM --------
900 LED 2 OFF;LED 1 OFF;LED 0 OFF;GOTO 1000
901 LED 2 OFF;LED 1 OFF;LED 0 ON;GOTO 1000
902 LED 2 OFF;LED 1 ON;LED 0 OFF;GOTO 1000
903 LED 2 OFF;LED 1 ON;LED 0 ON;GOTO 1000
904 LED 2 ON;LED 1 OFF;LED 0 OFF;GOTO 1000
905 LED 2 ON;LED 1 OFF;LED 0 ON;GOTO 1000
906 LED 2 ON;LED 1 ON;LED 0 OFF;GOTO 1000
907 LED 2 ON;LED 1 ON;LED 0 ON;GOTO 1000
910 REM --------
1000 FOR D=1 TO 500
1010 NEXT D
1020 RETURN
```

字体を定義

もしスイッチが押されていなければ進まない

❶Nに0～9を入れて❷を呼び出す

❷Nの字体の0行～4行を❸で描く

❸行を描く

約8m秒待機して呼び出し元へ戻る

バーサライタの実行例を下に示します。少し離れて目を細めると「0123456」に見えます。この写真はカメラのシャッターを開放にして撮影しました。実物を肉眼で数字と認識するには、いっそうの努力が求められます。だとしても、出来あいの製作物を応用して曲がりなりにも数字らしきものを描いたことは評価されていいでしょう。

◉バーサライタの実行例

以上ふたつの実例は同じ製作物にまったく違う仕事をさせます。コンピュータがソフトウェア次第でいろいろに動き、無限の可能性をもつことは、もはや周知の事実です。重要なのは、それをC言語のファームウェアではなくBASICのプログラムで実現したことです。見た目に区別がつかないこの小さな違いは、案外、大きな利点です。

C言語のファームウェアは、この製作物でいうと、PIC24FJ64GA002のデータシートを読破し、回路の構成を理解したうえで書くことになります。そういうことが得意な人に、面白い応用を思い付く才能まで求めるのはあんまりです。BASICが使えるなら問題がありません。面白い応用を思い付いた人が、プログラムを書けばいいからです。

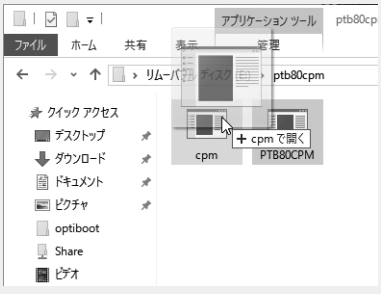
BASICは文法がコンピュータの仕組みに依存しないので、PIC24FJ64GA002も回路の構成も知らなくて大丈夫です。唯一、BASICの言語仕様を知っておく必要がありますが、コンピュータの仕組みから理解するよりは簡単です。そして、ひとたびおぼえたら、その知識は同じ名前のBASICが走るすべてのコンピュータに通用します。

東大版タイニーBASICの実物を動かしてみる

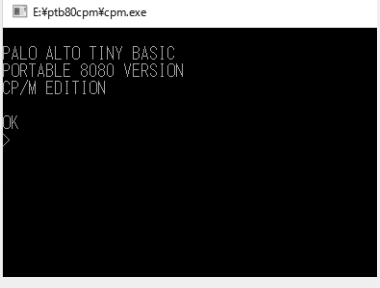
豊四季タイニーBASICの言語仕様はパロアルトタイニーBASICをお手本としています。パロアルトタイニーBASICの言語仕様は東大版タイニーBASICから推し量りました。そんなわけで豊四季タイニーBASICを開発している間ずっと、傍らにIntel8085（Intel8080のバイナリを実行できるCPU）のコンピュータを置いて東大版タイニーBASICを動かしていました。私はビンテージ級のCPUを多数所蔵しているのでこういうことができます。誰もができることではありません。

普通の人がIntel8080のバイナリを動かしてみたいと思ったとき現実的な環境はCP/Mエミュレータでしょう。そう考えて東大版タイニーBASICをCP/Mに移植し、サポートページに置いてあります。配布ファイルを展開すると実行ファイルPTB80CPM.COMがあります。これを適当なCP/Mエミュレータで実行します。

動作確認に使ったCP/MエミュレータはCP/M program EXEcutor for Win32です（入手元➲http://hp.vector.co.jp/authors/VA000084）。インストールは必要ありません。配布ファイルを展開するとcpm.exeがありますから、そのアイコンにPTB80CPM.COMのアイコンをドロップするのがいちばん簡単な起動の方法です。東大版タイニーBASICはコマンドプロンプトのウィンドウで実行されます。



CP/M program EXEcutorの操作例



東大版タイニーBASICが起動した様子

歴史に名を残したコンピュータは写真で見たり資料で想像したりするしかありませんが、ソフトウェアは実物に触れることができます。東大版タイニーBASICを動かし、バイナリひとつひとつの挙動から、デジタルの世界がまだ混沌としていた時代の息吹を感じ取ってください。