Vol.2012-UBI-34 No.8
2012/5/17

iArduino: マイコン上で対話的な開発を実現するインタプリタ型言語

光永 法明[1,a)]

概要：本論文では，マイコンを利用した製作の初心者へ向けて，マイコン上で動作し対話的にプログラミングができるインタプリタ言語 iArduino と，プログラム動作と入出力を可視化するツール iArduinoTerminal を作成したので，その設計方針と実装結果を紹介し議論する．

キーワード：Arduino，インタプリタ，可視化，プログラミング言語，

iArduino: An interpreter language which runs on a micro-controller

Noriaki Mitsunaga[1,a)]

Abstract: This paper reports an implementation of an interpreter “iArduino” which runs on a micro-controller and visualization tool “iArduinoTerminal” for it. We develop iArduino and iArduinoTerminal to help beginners understand programming and electronic circuits.

Keywords: Arduino, interpreter, visualization, programming language

1. はじめに

近年，マイコン（マイクロコンピュータ/マイクロコントローラ）が安価になる一方で処理速度や記憶容量といった性能が向上し，応用範囲が広がっている．また実世界指向インタフェースへの関心の高まりとともに，Arduino [1] や Gainer [2] といったアーティストやデザイナー向けにプロトタイプ作成用に開発されたボードと開発環境が広く知られるようになり，マイコンを利用する層も広がりを見せつつある．また 2012 年度から中学校の技術・家庭科では実世界とつながったコンピュータのプログラム作成の指導が必須となる [3]．したがって，マイコンを使った製品や作品製作に関心が高まるとともに，初心者にも分かりやすい開発環境がより重要となる．

マイコンを利用した製品や作品を制作するには，目的に合った回路の設計と実装，開発に利用するプログラミング言語とライブラリ (API) についての知識と活用が必要とされる．比較的初心者にも取り扱いやすいといわれるマイコン向けの BASIC 言語処理系である Parallax の BASIC Stamp [7] モジュールや各社のマイコン向け BASIC コンパイラ [10], [11], [12] などでは，典型的な利用方法の場合にはマイコンや周辺回路の初期化や操作について記述量が少なくなるように工夫されており，ライブラリのリンクを意識させない．C(C++) 言語の処理系では記述量の少なさよりも汎用性や拡張性などが重視される傾向があり，サンプルプログラムのみでライブラリが用意されない場合や，用意されてもライブラリのリンクに明示的な指定が必要な場合が多い．そのため難しいという印象を持たれることが多いが，Arduino 言語（C/C++言語をベースとする）では初期化，操作やライブラリのリンクについて，多くの知識を必要としないよう工夫がされている．

また多くのマイコンには，あらかじめキャラクタ液晶や

[1] 大阪教育大学
Osaka Kyoiku University
[a)] mitunaga@cc.osaka-kyoiku.ac.jp

可変抵抗，LED 等を実装した基板と，サンプルプログラムをセットにした学習キット（トレーニングセット）が用意されている．学習キットで学んだあとで，キットの回路をベースに目的の回路を制作することが想定されており，サンプルプログラムの流用について制限が少ない場合も多い．さらに，マイコン（あるいはマイコンボード）と親和性の高い電気的・機械的なインタフェースをもつセンサ・アクチュエータなどのモジュールも多く市販されており，サンプルプログラムが用意されているものも多い．

このように理解・利用しやすい単位のソフトウェアやハードウェアのブロックを多く用意し，設計・製作を容易にする様々な手法が提案・実現・市販されている．一方で各ブロックの利用方法を理解し，適切に利用することが不可欠であることは変わらない．ソフトウェア，ハードウェア共に理解には，実際に実装をし動作を確かめることが早道である．動作を確認するために，ソフトウェアではデバッガ，ハードウェアでは回路計やオシロスコープといった可視化ツールが用いられる．しかし，それらのツールを利用するには，基本的なプログラミング言語，マイコン，回路動作の理解が必要である面があり，またツール操作の習得や実際の操作にも時間が必要なことから，使用が有効な場面でも初心者には使われない傾向がある．

したがってソフトウェアの基本的な操作ができれば使用できる，可視化ツールを用意すれば初心者の助けになると考えられる．またプログラミング言語で書いた計算式（命令）の動作を対話的に確かめられるインタプリタ型言語であれば理解の助けになるとともに，可視化ツール用のインタフェースをソフトウェアで実装することも容易と考えられる．そこで，Arduino 言語と互換性のあるインタプリタ型言語 iArduino と，iArduino と通信して変数や入出力ピンの変化を容易に観察，操作できるターミナルソフト iArduino Terminal を作成したので報告する．

以下，現在のマイコン向け開発環境について概観する．そして iArduino と iArduinoTerminal の設計指針と実装について紹介し，最後に議論とまとめを述べる．

2. マイコン用開発環境の比較と設計方針

2.1 マイコン用開発環境の比較

表 4 に初心者に向くと考えられるマイコン向け開発環境として，グラフィカルプログラミング環境，統合開発環境をもつコンパイラ型かインタプリタ型の BASIC 処理系，Arduino（合計 11）を取り上げて比較を示す．表からは BASIC 処理系が 7/11 と多数を占めている．これは BASIC が初心者にも分かりやすいと解釈されたためと考えられる．しかし，Arduino が急速に普及したことからも，BASIC の文法が初心者にも易しく，C/C++言語の文法が難しいとはいえない．また，コンパイラ型言語が 8/11 となっている．この理由はメモリの限られたマイコンにおいてはメモリの利用効率が重要となり，また動作速度の要求にも応えようとしているためである．

入出力・変数値・実行文（命令）の可視化に注目すると 12Blocks [6] と Scratch をのぞくと初心者向けの可視化支援がされていない．6/11 の開発環境では，開発環境による可視化の支援はなく，いわゆる printf デバッグ（シリアルポート利用）もしくは，出力ピンの操作と回路計やオシロスコープの組み合わせが必要となっている．また 3/11 の開発環境ではシミュレータやデバッガの利用が必要であり，可視化について初心者に向けて十分に配慮されている開発環境は 2/11 と多くない．

PIC Pico Basic [4], Arduino BASIC [5] はインタプリタ型の BASIC 言語を実装しており，対話的な計算（実行）や実行の中断が可能となっている．一方で統合開発環境は持たず，シリアルターミナルによりマイコンとユーザが対話して開発をするため，可視化のツールを持たない．

ところで 8 歳から 16 歳を対象に設計されている Scratch [13] ではグラフィカルプログラミング言語を採用している．開発環境では，アイコンをクリックするだけでアイコンの命令が動作し，対話的に命令の動作を把握できる．また実行中の命令が分かるように表示に工夫がされている．このような対話的操作や可視化は年齢を問わず有効であると考えられる．一方で Scratch ではマイコンは I/O のために用いられており，開発したプログラムはマイコン単体では動作しない．

2.2 インタプリタ型言語 iArduino の設計方針

ROM が数 10kbytes，RAM が数 kbytes 程度とメモリの制約が大きいマイコン上で動作するインタプリタ型言語では，実行可能なプログラムの大きさも限られる．そのため規模の大きいプログラムを作成しようとしたときに，スムーズにコンパイラ型言語へ移行できることが望ましい．そのためには，インタプリタ型言語が動作するマイコンをターゲットとするコンパイラ型言語開発環境の入手が容易で，二つの言語間で文法やライブラリ API の互換性があるとよい．また市販の開発環境の中には付属するライブラリを用いた他の言語処理系の作成が禁じられているものもあるが，そのような制約の少ない開発環境であればインタプリタ型言語のライブラリの充実が容易である．

そこで iArduino の開発に当たっては Arduino マイコンボードを対象とし，Arduino 言語と互換性を持たせることにする．Arduino マイコンボードは既に 20 万個以上が出荷されており，初心者向けとしても人気があり，開発環境 Arduino IDE の入手も容易である．また Arduino 言語のコンパイラは gcc をベースにしており，付属のライブラリは GPL や Lesser GPL などのライセンスで提供されている．そのため上記のような制約が少ない．そこで iArduino の開発には Arduino 言語 (C/C++) を用いる．

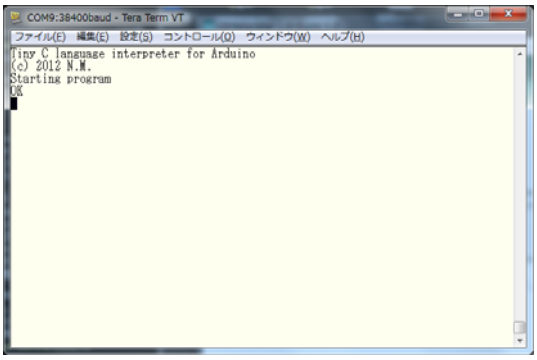

図 1 iArduino が起動時に出すメッセージ

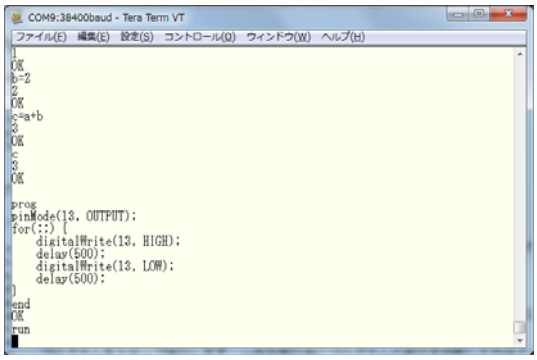

図 2 iArduino をシリアルターミナルから使っている様子

iArduino 言語は Arduino 言語と文法の互換性を持つが次のような点で異なる．プログラムの規模が小さいことから，iArduino ではプログラムの先頭から順次実行し，関数の定義をサポートしない．変数については宣言を不要とする代わりに，あらかじめ用意された変数名を使うものとする．演算はすべて符号付き 16 ビット整数とする．文字列は取り扱わない．

3. インタプリタ型言語 iArduino

3.1 iArduino の概要

Arduino マイコンボード Arduino Uno にインタプリタ型言語 iArduino を実装した．Arduino Uno には Atmel 製の AVR アーキテクチャ ATmega328P マイコン (フラッシュ ROM 32kbytes, RAM 2kbytes, EEPROM 1kbytes) が搭載され, 16MHz のクロックで動作する．iArduino がサポートする変数名や制御文，定数などの一覧を表 5 に示す．関数名と動作は Arduino 言語に準じ，デジタル入出力，アナログ入力，PWM 出力，ラジコンサーボ出力，トーン出力などが可能であるが，Arduino 言語と違い多態性をもたない．後述するデバッグインタフェースを含め，iArduino のバイナリの大きさが約 20kbytes，実行可能なプログラムのサイズが 600bytes である．

iArduino は非同期シリアル (115.2kbps) を通したユーザインタフェースを持つ．iArduino の初期化が終わると図 1 のようにメッセージと OK のプロンプトを出す．OK のプロンプトが表示されているときにユーザインタフェース（シェル）が受け付けるコマンドを表 1 に示す．式を入力するとすぐに結果を返すので対話的に，関数の実行結果などを確かめられる (図 2)．図 2 では，いくつかの式を評価させ，Arduino 上にある 13 ピンにつながった LED を 1 秒周期で点滅させるプログラムを入力後，実行している．プログラム実行は通常の実行の他，ステップ実行，0.5 秒に 1 文の実行があり，実行中に実行文の表示もできる．プログラムは RAM 上に置き実行されるが，マイコン内蔵の不揮発性の EEPROM へ保存し起動時に読み込み，実行も可能である．つまりプログラミングが終了すれば，マイコン単体で動作する．

3.2 可視化ツール iArduinoTerminal

iArduino と通信し変数の値等を可視化するツール iArduinoTerminal を作成した．開発には Microsoft Visual C# 2010 Express Edition を用いている．図 3 に可視化ツール iArduinoTerminal の実行中の様子を示す．図左上が iArduinoTerminal のメインウィンドウで，シリアルターミナルとしての機能と変数，入出力ピンの表形式での表示機能を持つ．また頻繁に入力するコマンドをマウス操作で入力出来るボタンを用意している．図の右上のウィンドウは入出力ピンの変化をグラフで表示する．マウスオーバで値と時刻を読むことも出来る．図左下のウィンドウはプログラムエディタである．プログラムの実行中には実行中の行を赤で表示する．図右下のウィンドウでピンを操作出来る．マウスでピンの機能（デジタル入出力，アナログ入力，PWM 出力，ラジコンサーボ出力）を選択し，ラジオボタン，スライダーで値を操作し，回路の動作などを確かめられる．またマウス操作と等価になるプログラムの文を表示する．

このような可視化ツールを用意するために定めたデバッグ用のプロトコルを表 2 に示す．プログラムの実行文の範囲をのぞき，iArduinoTerminal の要求により iArduino が応答をする．各要求/応答を iArduinoTerminal または iArduino が送る際には 0x7f，プロトコル番号，データの順に送り可変長である．0x7f によりターミナルから入力されるシェルの操作と区別している．チェックサムなどの誤り検出・訂正は行っていない．iArduinoTerminal（Atom プロセッサ 1.6GHz1 コアの PC 使用）から 100ms 周期で要求を出し，26 の変数と 3 つのデジタルポート，6 つのアナログ入力を可視化できている．

3.3 iArduino によるプログラムの実行速度

iArduino によるプログラムの実行速度を確かめるため，図 4 のプログラムでデジタル出力ピンを無限ループで H/L と変化させてオシロスコープで観察した．図 4 の場合と 3 行目と 4 行目の間へ加乗除の計算と時間待ち関数を入れた場合について観察結果を表 3 に示す．表から Arduino 言語をコンパイルしたバイナリの実行速度に対し，iArduino

表 1 iArduino のシェルが受け付けるコマンド

コマンド	説明
animate	プログラムを実行する．実行中に命令を表示し，一命令実行毎に 0.5 秒の間をあける．
autorun	Arduino 起動時に EEPROM に記憶されたプログラムを実行する．
debug	プログラムを実行する．実行する命令を表示する．
edit ＜行番号＞	プログラムリストの＜行番号＞の行を置き換える．置き換える行は edit コマンドの続いて入力する．
list	プログラムリストを表示する．先頭に行番号を表示する．
noauto	Arduino 起動時に EEPROM に記憶されたプログラムを実行しない．
prog	prog を入力すると，プログラムの入力待ちになる．プログラムの最後に end を入れるとコマンド待ちに戻る．
run	プログラムを実行する．
save	現在のプログラムを EEPROM に記憶する．
step	プログラムをステップ実行する．一命令実行毎にプログラムの実行を中断し入力待ちとなる．
式	式を入力すると計算結果を表示する

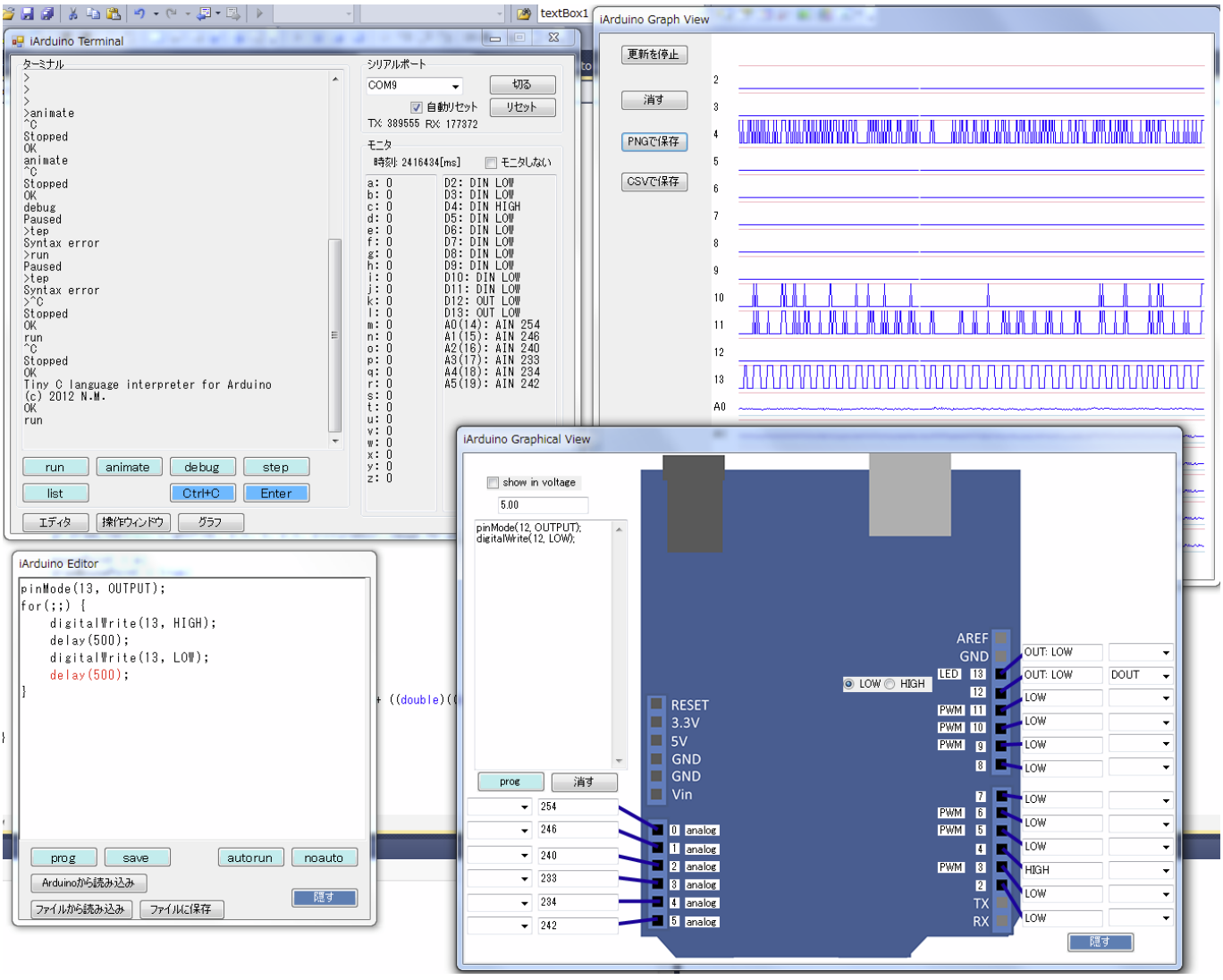

図 3 iArduinoTerminal のウィンドウ．左上のウィンドウが起動時に表示されるシリアルターミナルと変数，入出力ピンモニタである．右上のウィンドウは入出力ピンの変化をグラフで表示するウィンドウである．左下のウィンドウはプログラム用のテキストエディタで，プログラム実行中は，実行している行が赤で示される．右下のウィンドウはピンの操作ウィンドウである．ピンを GUI で操作し，回路の動作などを確かめられる．また操作をプログラムの文として表示する．

表 2 iArduinoTerminal から iArduino への要求と応答

番号	要求/応答
0x00	バージョン
0x01	変数値
0x04	入出力ピンの値
0x0d	プログラムリスト
0x0e	プログラム実行時に実行位置を送る (応答なし，以降プログラム実行中に実行文の範囲を送る)
0x0f	Arduino 起動時からの経過時間 (ms 単位)
0x10	ピンの入出力設定 (応答なし)
0x11	デジタル出力 (応答なし)
0x12	PWM 出力 (応答なし)
0x13	ラジコンサーボ出力 (応答なし)
0x14	ラジコンサーボ出力の設定 (応答なし)

```
1  volatile int a = 2;
2  for (;;) {
3  digitalWrite(2, HIGH);
4  digitalWrite(2, LOW);
5  }
```

図 4 デジタル出力ピンによる実行速度の測定に使ったプログラム (測定部分のみ)．digitalWrite 関数は 1 つ目の引数のピンに 2 つ目の引数を出力する．iArduino では volatile int を省く．

表 3 図 4 のプログラムで観察されたデジタル出力ピンの H/L となる時間の観察結果

	Arduino-1.0	iArduino	iArduino + iArduinoTerminal
なし	H:3.9us	H:6.0ms	H:6.0ms〜8.6ms
	L:3.9us	L:6.1ms	L:6.1ms〜9.2ms
$a = a + 1$	H:4.6us	H:6.1ms	H:6.1ms〜9.0ms
	L:3.9us	H:6.1ms	H:6.1ms〜9.2ms
$a = a * 3$	H:4.8us	H:6.1ms	H:6.1ms〜9.0ms
	L:3.9us	L:6.1ms	L:6.1ms〜9.2ms
$a = 10/a$	H:44us	H:6.1ms	H:6.6ms〜9.0ms
	L:3.9us	L:6.1ms	L:6.1ms〜9.2ms
delay(1)	H:1.0ms	H:9.3ms	H:9.5ms〜13ms
	L:3.9us	L:6.1ms	L:6.1ms〜9.2ms
delay(10)	-	H:18ms	H:18ms〜23ms
	-	L:6.1ms	L:6.1ms〜9.2ms
delay(100)	-	H:11 × 10ms	H:11 × 10ms[*1]
	-	L:6.1ms	L:6.1ms〜9.2ms

[*1] 108ms〜111ms で揺らぐ

での実行速度は 1/1500 ほどであるとわかる．また主としてプログラムの解釈に時間がかかっており，加算と除算で出力が H である時間が変わらない．時間待ち関数の引数の単位は ms であるが，呼び出し等のオーバヘッドにより iArduino では待ち時間が短い場合に誤差が大きい．一方で，変数等の可視化に iArduinoTerminal を使用した場合のオーバヘッドは最大で 5ms 程度であると分かる．

4. 議論

iArduino のプログラムの実行速度からは，ソフトウェアによる LED やモータの調光，速度調整を目的とした PWM 生成や，可聴音の生成，LED のダイナミック点灯は難しい．しかし，PWM 出力やトーン出力の関数を用意しているので LED のダイナミック点灯以外は容易に実現出来る．また，人の操作であったり，環境光の明るさや環境音の大きさの緩やかな変化への反応を実現するような，数 10Hz 以下の制御周期で十分な用途には問題がないと考えられる．

ところで iArduinoTerminal による動作の可視化デモを関西の技術系のイベント等で展示をしたところ，初心者の方にもベテランの方にも可視化について好意的な意見をいただいた．とくに初心者の方からは最近苦労した経験からか「わかりやすい」との声が多かった．インタプリタ言語であることについては，ベテランの方から 8 ビットパソコンの BASIC を思いだし，懐かしいという声があった．また Arduino 言語からも iArduinoTerminal の可視化を使いたいという要望があり対応している．

5. まとめ

本論文ではマイコンを使った製作の初心者へ向けて，インタプリタ言語 iArduino と動作の可視化ツール iArduinoTerminal の作成について設計方針を述べ，実装と動作速度を紹介した．今後は，これらを使った作例や手引きなどを充実させていきたい．

参考文献

[1] Arduino project：Arduino, 詳細 (http://arduino.cc/)
[2] GAINER.jp：GAINER, 詳細 (http://gainer.cc/)
[3] 文部科学省：中学校学習指導要領解説技術・家庭編 (平成 20 年 9 月), 2008.
[4] picobe：PIC Pico Basic, 詳細 (http://www14.ocn.ne.jp/p̃icobe/)
[5] Mike Field：Arduino BASIC, 詳細 (http://ec2-122-248-210-243.ap-southeast-1.compute.amazonaws.com/mediawiki/index.php/Arduino_Basic)
[6] HannoWare：12Blocks, 詳細 (http://12blocks.com/)
[7] Parallax Inc.：BASIC Stamp Windows Editor, 詳細 (http://www.parallax.com/tabid/295/Default.aspx)
[8] Parallax Inc.：Propeller/Spin Tool Software, 詳細 (http://www.parallax.com/tabid/407/Default.aspx)
[9] Revolution Education Ltd.：PICAXE Programming Editor, 詳細 (http://www.picaxe.com/)
[10] MCS Electronics：BACOM-AVR, 詳細 (http://www.mcselec.com/index.php?option=com_content&task=view&id=14&Itemid=41)
[11] mikroElektronika：mikroBASIC Pro for PIC, 詳細 (http://www.mikroe.com/eng/categories/view/98/mikrobasic/)
[12] MicroEngineering Labs.：PICBASIC PRO, 詳細 (http://melabs.com/)
[13] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond：The Scratch Programming Language and Environment, in the *ACM Transaction on Computer Education*, vol.10, no.4, pp.16:1-16:15, 2010.
[14] The Playful Invention Company：PicoBoard, 詳細 (http://www.picocricket.com/picoboard.html)

表 4 初心者に向くといわれるマイコン（マイコンモジュール）用の開発環境の比較

[tbp] 名称 (バージョン)(備考)	ハードウェア	言語	言語処理	対話的実行	入出力・変数値・ 実行文の可視化	開発環境の費用
PIC Pico Basic (20111218) [4]	Microchip PIC24FJ64GA002	BASIC	インタプリタ (マイコン上)	あり	なし (実行中断後 print 可)	無償
Arduino BASIC (ver. 0.03) [5]	Arduino	BASIC	インタプリタ (マイコン上)	あり	なし (実行中断後 print 可)	無償
12Blocks (2.0.2) [6]	Parallax Propeller, PICAXE ほか	グラフィカル言語	コンパイラ	なし	あり (表・グラフ) (PICAXE 等ではなし)	$49〜
BASIC Stamp Windows Editor (v2.5.2) [7]	Parallax BASIC Stamp モジュール	BASIC	コンパイラ (中間言語)	なし	なし	無償
Propeller/Spin Tool Software (v1.3) [8]	Parallax Propeller	Spin 言語	コンパイラ	なし	なし	無償
Arduino IDE(Arduino-1.0) [1]	Arduino	Arduino 言語 (C/C++ベース)	コンパイラ	なし	なし	無償
PICAXE Programming Editor (ver. 5.5.0) [9]	PICAXE (専用ファームウェアの 入った Microchip PIC マイコン)	BASIC	コンパイラ (中間言語)	なし	なし	無償
BACOM-AVR (2.0.5.0) *1	Atmel AVR マイコン	BASIC	コンパイラ	なし	シミュレータ上で可	容量制限版は無償
mikroBASIC Pro for PIC (5.40) [11]	Microchip PIC マイコン	BASIC	コンパイラ	なし	デバッガによる	容量制限版は無償
PICBASIC PRO (3.0.5) [12]	Microchip PIC マイコン	BASIC	コンパイラ	なし	デバッガによる	$49.95〜
Scratch (1.4) [13] (マイコンボード単体で動作せず PC が必要)	PicoBoard [14] (互換ボードあり)	グラフィカル言語	インタプリタ (PC 上)	あり	あり (表形式)	無償

表 5 iArduino 言語のサポートする変数名，制御文，定数等の一覧

変数	a-z の 26 個（16bit int 型）
制御文	if, else, for, while, break, continue
定数	LOW, HIGH, INPUT, OUTPUT, true, false
演算子	+, -, *, /, %, &, |, ; &&, ||, <, <=, >, >=, ==, ! =, >>, <<
数値	10 進数，16 進数，2 進数（16bit int のみ）
関数	abs, analogRead, analogWrite, delay, digitalRead, digitalWrite, max, millis, min, noTone, rand, pinMode, servo?.attach*1, servo?.write, tone, print

*1 servo?.attach, servo?.write の ? には 0 から 11 の数値が入る