並列処理プロセッサを活用した
メディアアートのための汎用インターフェース

長嶋洋一

静岡文化芸術大学

理工系でないデザイン系の学生でも、サウンド・インスタレーション作品やインタラクティブ作品を実現できる汎用インターフェースの新しい提案である。リアルタイム処理のために必要な「割り込み」や「モニタ/カーネル」を不要とする並列処理プロセッサ(Parallax社Propellerチップ)の活用によって、ソフトウェア部品を単純に連結するプログラミングで、複雑な高速多重処理を容易に実現できるようになった。

Universal Interfaces for Media Arts using the Parallel Processor

Yoichi Nagashima (*nagasm@computer.org*)

Shizuoka University of Art and Culture

This is a report of universal interfaces for media arts - sound installations and interactive installations. The Propeller chip makes it easy to develop universal interfaces. Its eight processors (cogs) can operate simultaneously, either independently or cooperatively, sharing common resources through a central hub. We have full control over how and when each cog is employed; there is no compiler-driven or operating system-driven splitting of tasks among multiple cogs. The Propeller can execute at up to 160 MIPS.

1. はじめに

筆者はSUAC(静岡文化芸術大学)において、メディア・デザイン系の学生に対して、インスタレーションなどのインタラクティブ作品の創作に関連する教育を行っている[1-6]。デザイン系のアーティストにとって、インタラクティブな仕組み(広義のセンサと広義のディスプレイ)や、インタラクションのアルゴリズムの実現(プログラミング)を工学的に全てゼロから行うことは不可能であり、アート/デザインの制作の裏側に、ある種のブラックボックスとして活躍する制作支援環境が必要となる。プラットフォームとしては、Max/MSP/jitter[7-9]やGAINER[10](Fig.1) およびFLASH、そしてインターフェースとして秋月電子のAKI-H8[11-12](Fig.2)を活用してきた[13-14]。

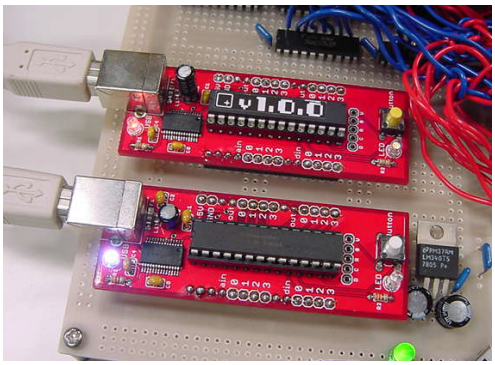

Fig.1 GAINER

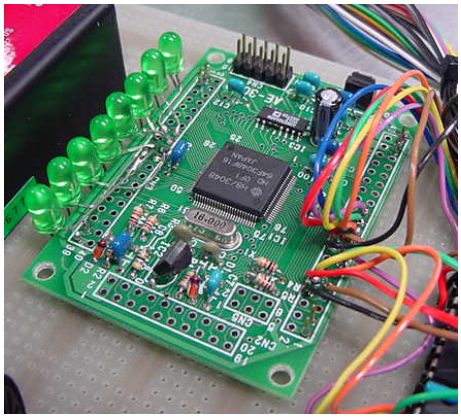

Fig.2 AKI-H8

2.代表的なプラットフォームの検討

AKI-H8とGAINERとPropeller[15-17]を検討したプラットフォームに関する発表[13-14]の場でフロアから得たコメントにより、イタリアのArduino[18-19](Fig.3)を知ったので、この4種類のシステムを比較検討した。この他にも、PICやBasic Stampなどのプラットフォームも有名であるが、ここでは実際に具体的な製作事例とともに筆者が実験したり、多くの事例による実績のある4種に限定した。

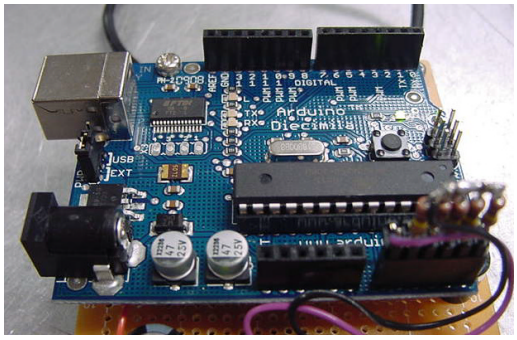

Fig.3 Arduino

2-1. AKI-H8

秋葉原・秋月電子のAKI-H8は、これまでメディアアートのプラットフォームとして多くの作品の基盤として活躍してきた。筆者のサイトを参考にシステムを製作する人も多く、ICCでの2年間の常設展示(連続運転)など信頼性の上でも実績は多い。MS-DOS時代からの開発環境は最近では古めかしいが、スピードを求めなければC言語で、また性能を求める場合にはアセンブラによってファームウェアを開発し、実機上でEEPROMに書き込める手軽さはまだまだ現役である。従来のMIDI/RS232Cに加え、USBやイーサネットI/Fも登場した。MIDI等で通信するスタンドアロンシステムのプラットフォームである。

2-2. Arduino

イタリアのグループが開発したArduinoは、オープンソースの開発環境と低価格のボード、スタンドアロンおよびパソコンI/F(Max/MSP, Flash, Processingに対応)の動作モード、さらにZigbeeに対応したXbee Shieldオプション等を売りにしている。ホストPCとはUSBで接続するが、シリアルポートが1ポート(Tx/Rx)しかなく、標準開発環境のC言語ベースでは、センサ情報のMIDI送信には問題ないものの、MIDI受信には対応できない問題点を確認した。あまり時間的な制約のない範囲で、多種のセンサ情報に対応した関係性を実現するスタンドアロンシステムの中核として有力な候補である。

2-3. GAINER

IAMASの小林氏の開発したGAINERは、日本においてPhysical Computingの概念をセンセーショナルに普及させた記念碑的なシステムである。スタンドアロン動作を排してパソコンI/Fモードに限定し、ホスト環境としてはMax/MSP/jitterとFlashとProcessingをカバーしており、デザイン系の学生などがインタラクティブなインスタレーションを実現する最短経路を提供する。CPUのファームウェアを書き換える等の荒技を別にすれば、MIDI入出力や単独動作など出来ない事も多いが、16ビットのポートを入出力のいろいろな組み合わせとする支援環境などの充実により、最近もっとも注目されている。

2-4. Propeller

Basic Stampの開発元で知られるParallax社が新しく発信しているCPUが、Propeller(Fig.4)である。従来とは基本的概念から異なるユニークなCPUであ、8個の並列動作する32ビットCPUコアによって、非常に強力な性能のシステムをスタンドアロンで実現できる。

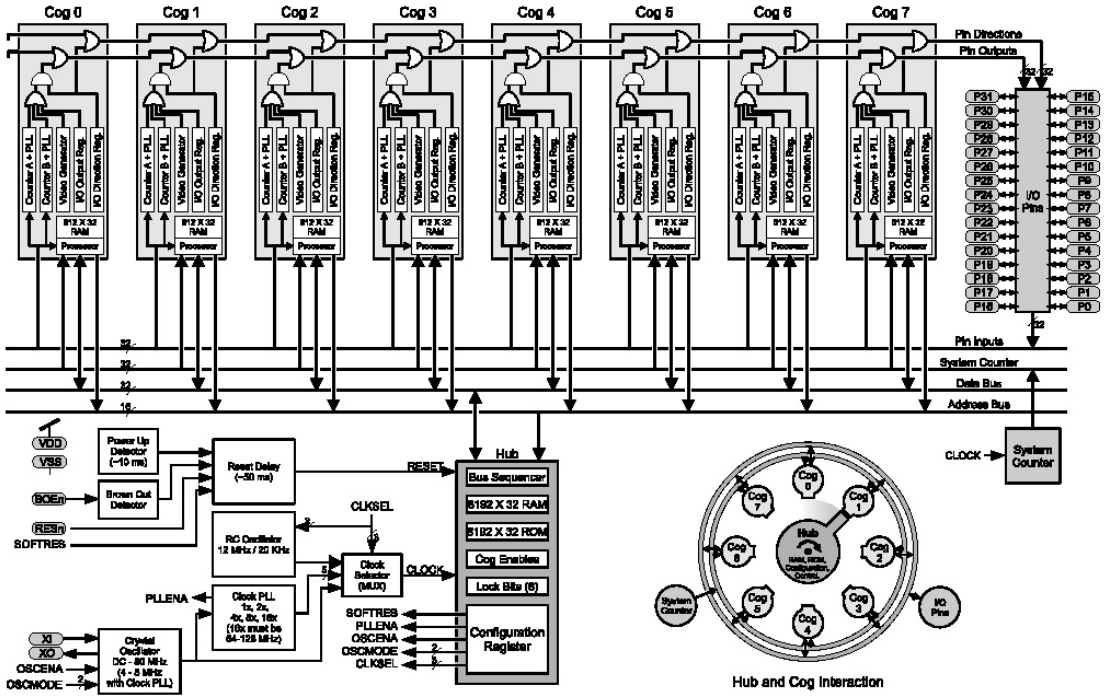

Fig.4 Propellerチップのシステムブロックダイアグラム

UART等に頼らずシリアル通信でもCDレベルのディジタルオーディオ出力でもソフトウェアだけで実現し、さらにNTSC/PAL対応のビデオ信号まで8個のCPU(Cogと呼ぶ)のうち2個により生成できてしまう能力には注目したい。オブジェクト指向の並列処理記述言語Spinによる開発環境(アセンブラと協調)も充実し、Propellerチップを取り囲むライブラリ等の支援環境によって、デザイン系の初学者でもハイレベルのシステムを実現できる可能性を感じさせる。

本稿末尾の表(Table.1)に、AKI-H8、GAINER、Arduino、Propellerの4種のチップ/システムを、メディアアートのプラットフォームという視点で比較してみた。

3. Propellerによるシステムの製作例

3-1. Propellerの概要

PropellerチップはFig.4のような内部構成である。すなわち、並列動作する8個の32ビットCPU(最大クロック80MHz)を持ち、32ビットの周辺ポートをそれらが共有している。"Cog"と呼ばれる8個の32ビットCPUにはそれぞれに、512long(32ビット幅なのでバイトでもワードでもなくlong)のRAMとカウンタ回路があり、これは同時にソフトウェアによってA/Dコンバータ、D/Aコンバータ、あるいはNTSC/PAL/VGA生成回路とすることも可能なので、使い方によっては1チップで複数系列のビデオ信号を同時に生成することもできる。チップ上にはさらに8KlongsのROMとRAMが共有メモリとして搭載されており、各Cogに対してはハードウェアが強制的に時分割して割り当てる。つまりPropellerには割り込みという概念が存在せず、ハードウェア的・本質的に並列処理を実現する。

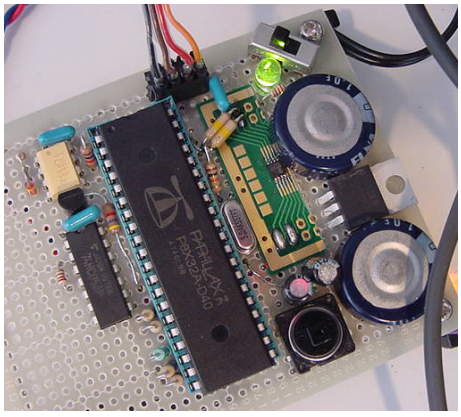

Fig.5 Propeller実験ボード

Fig.5は、筆者が制作したPropeller実験ボードの写真、Fig.6は実際に開発したソフトウェアでPropellerシステムが動いている様子の一例である。ここでは、Max/MSPから送ったMIDI信号を受信し、内部でバッファリングして再びMIDI信号出力し、そのピッチに対応した正弦波をステレオでオーディオ出力し、さらにNTSCビデオ信号としてグラフィック描画出力し、そのパラメータをMIDI入力によってリアルタイム変化させているところである。重要なのは、MIDI入出力の通信にUARTなどの専用チップは存在せず、全てCogのソフトウェアでシリアル通信している(1ビットごとに上げ下げ処理)こと、ステレオサウンドもソフトウェアによるPWM処理で44.1KHzサンプリングでリアルタイム生成していること、NTSCビデオ信号もソフトウェアによってグラフィックドライバ(1Cog)+リアルタイム生成(1Cog)していること、である。

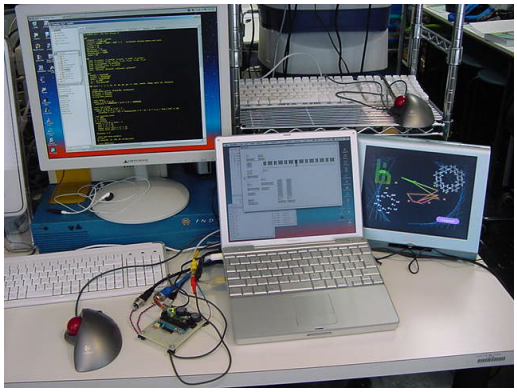

Fig.6 Propeller実験ボードの動作風景

3-2. Propellerの開発環境

Propellerの開発環境についても、IT教育に力を入れるParallax社のこだわりを感じられる。Fig.7はWindows環境でフリーに提供されているPropellerの開発環境(IDE)の画面例である。アセンブラとともに、並列処理に最適化されたオブジェクト指向言語"spin"によって、それと意識せずに、割り込みの存在しない本当の並列処理を簡単にプログラミング出来る、というのは特筆に値すると筆者は考える。CやJavaよりも簡単にマルチタスクを実現できるメリットは大きく、スタンドアロンのインスタレーションの中枢としてだけでなく、デザイン系学生のプログラミング教育の可能性もありそうである。

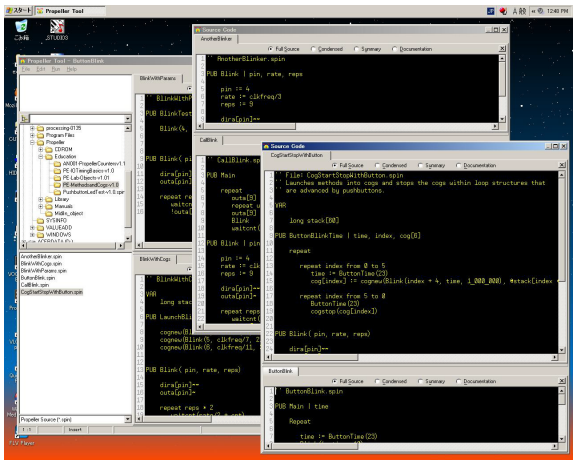

Fig.7 Propellerの開発環境

Propellerの起動モードについて解説した図がFig.8である。Propellerはリセットされると、まずはUSB経由でホストPCに接続されているかど

うかを調べ、ホストPCからプログラムを内部RAMに取り込む。ホストPCが無い場合には次に外部EEPROMが接続されているか調べて、そこからプログラムを内部RAMにダウンロードして実行する。

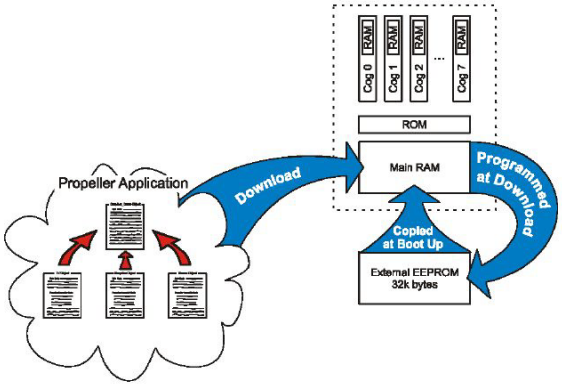

Fig.8 Propellerの起動モード

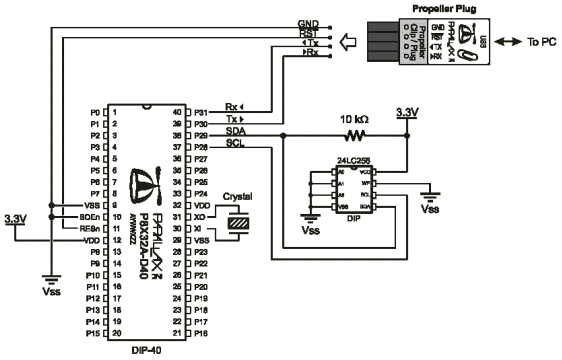

Fig.9 Propellerの基本回路

Fig.9が、Propellerチップのもっとも基本的なシステム回路である。外部には8ピンのシリアルEEPROMを接続し、ホストPCにはParallax社が提供している"Propeller Plug"というUSBインターフェースを接続するだけで、実験・試作システムはすぐに動き出す。

AKI-H8の場合、そしてGAINERでもArduinoでもファームウェアを開発して改訂する場合には、開発してみたプログラムをCPUの内部EEPROMに転送してリセットして実行してみる、というループを繰り返すことになるが、EEPROMの書き込み回数には制限があるために、何百回も何千回もオーバーライトすることには不安があった。

しかしPropellerシステムの場合には、CPUの内部にはEEPROMが無くて、PCモードではUSBからPropeller内部のRAMにプログラムを転送してテストランさせる。これはRAMなので、書き込み回数の制限を気にすることなく無限にテスト(プログラム開発)を繰り返すことが出来ることを意味し。システムの試作において歓迎できる。

3-3. Propellerの並列処理

Propellerチップの最大の特長である並列処理については、プログラマがPropellerシステムのハードウェアについて、並列処理のためのメカニズムをシステム内に隠蔽していることが重要である。Fig.10はPropellerチップ内で、spinインタプリタがそれぞれのCogsにアプリケーションを割り当てている動作の概念図である。Javaと同様に、プログラマはあるメソッドを1つだけ記述しておき、spin言語の"cognew"コマンドにより、そのメソッドのインスタンスがCog内部のRAMにコピー転送されて実行されるが、具体的に0-7のうち何番のCogに割り当てられるかは、プログラマは気にしなくて良い。システム(ハードウェア)が、たまたま空いているCogを捜して割り当てる。内部RAMの容量制限が厳しいのが最大の難点であるが、ここは「お任せ」である。

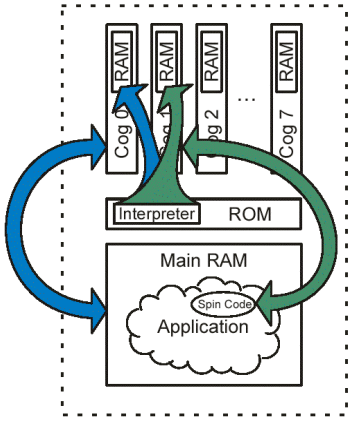

Fig.10 Propellerの並列処理の概念図

組み込みシステムではプロセスの終了という概念はあまり使わない(たいてい無限ループに割り込み)が、PropellerではCPUが8個もあるので、spinで記述したプログラムが終了すると、ハードウェアは自動的にそのCogを解放して、次の"cognew"コマンドに備える。

Fig.11は、Propellerの開発環境のメニューにある"Object Info."である。ここに、spinソースコード(インタプリタ中間言語)、定数ブロック、ワークメモリ定義などが表示されることで、限られたメモリと相談しながらプログラミングを進める。

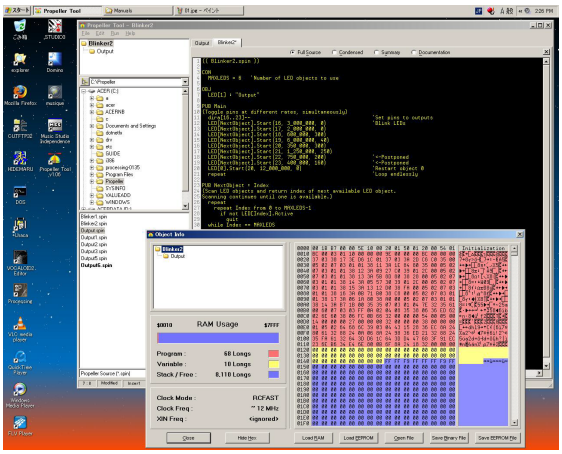

Fig.11 PropellerのIDEのメモリ表示

3-4. Propellerのアセンブラ設計思想

紙面の関係で本稿では詳解できないが、「これまでのあらゆるCPUについて検討した上で設計した」とParallax社が自賛するだけあって、spin言語だけでなく、Propellerチップのアセンブラは非常に洗練された素晴らしい体系を持っている。Propellerは32ビットCPUなので、全てのコードが32ビット(1long)で完結している。さらに、Cog内メモリとシステムメモリの容量制限を逆手に取って、1longのコード中にソースとデスティネーションのアドレシングを各9ビット格納している。面白いのは、そのコードのある部分だけをリライトできる機能である。一例として、初期化等であるコードを実行してもう使わない場合、そのコードの一部を動的に書き換えて別の処理で再利用する、というような事が容易である。プログラムは固定している必要はなく、動的に自分自身のプログラムを書き換えながらCPUがそれを実行していくのである。IT教育の意味からも非常に興味深い。詳しく参考文献[17]を参照されたい。

3-5. Propellerの音声信号処理

紙面の関係で本稿では詳解できないが、このPropellerチップのアーキテクチャには、音声信号処理をソフトウェアで実現するための設計が意識的に盛り込まれているように筆者は感じた。Parallax社のサイトにあるサンプルに「4声のポリフォニーでリアルタイム音声合成でコラールをハモる」というものがある。そのアセンブラコードを解析してみると、コーデックとかのディジタル音声信号処理を非常に効率良く実現できるように設計されていた。"MAKE"誌のインタビューによれば社長のこだわりであるらしい。今後、Computer Musicの新しいプラットフォームとして、パソコン不要でリアルタイム信号処理を実現するプラットフォームとしてのPropeller、の可能性もありそうである。詳しく参考文献[17]を参照されたい。

3-6. Propellerによるインスタレーション例

新しいメディアアートのプラットフォームとしてPropellerを活用する、という目的のために、インスタレーション作品(的なもの)を試作してみた。Fig.12は、2つの画面に別々のグラフィクスを描画出力し、マイクで環境音をセンシングして画像を変化させ、CdSの照度センサで明るさをセンシングして画像を変化させ、これらのセンサ情報をMIDI出力する、というシステムの開発中の模様である。詳しく参考文献[17]を参照されたい。

4. おわりに

メディア・デザイン系の学生のインスタレーションなどインタラクティブ作品、サウンドインスタレーション作品などの創作を支援する環境の新しいプラットフォームの提案について、Propellerチップを中心に報告した。

今後も、ますます発展するであろうこの領域での、サウンドとグラフィクス、あるいは身体性/インタラクションを統合した、メディアデザインのための新しいプラットフォームを提案していきたいと考えている。

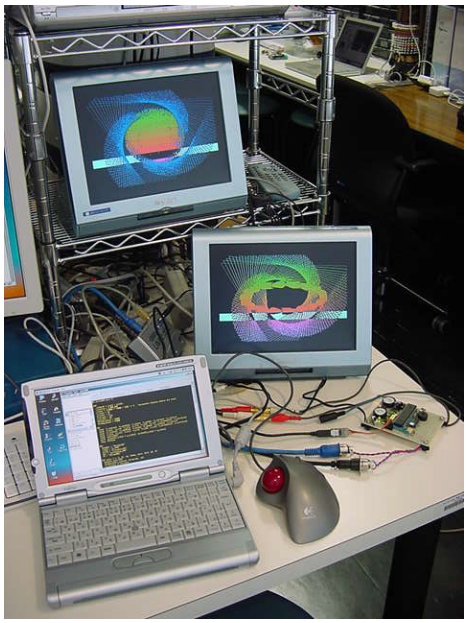

Fig.12 Propellerのシステム開発の例

参考文献/リンク

[1]長嶋洋一，メディアコンテンツ・デザイン教育におけるコンピュータサウンドの活用事例，情報処理学会研究報告 Vol.2007,No.102 (2007-MUS-72)，情報処理学会，2007
[2]http://nagasm.org/
[3]http://1106.suac.net/
[4]http://www.suac.ac.jp/~media/
[5]http://1106.suac.net/news2/installation/
[6]http://1106.suac.net/news2/installation2/
[7]長嶋洋一，インタラクティブアートの統合的システム・プラットフォームとしてのMax/MSP，DSPサマースクール2002論文集，静岡文化芸術大学，2002
[8]http://www.cycling74.com/
[9]http://opensoundcontrol.org/
[10]http://www.gainer.cc/
[11]http://nagasm.suac.net/ASL/mse/
[12]http://nagasm.suac.net/SSS/
[13]長嶋洋一，マルチメディア心理学実験のためのプラットフォームについて，日本音楽知覚認知学会2008年春季研究発表会資料，日本音楽知覚認知学会，2008
[14]長嶋洋一，サウンド・インスタレーションのプラットフォームについて」，情報処理学会研究報告 Vol.2007,No.50 (2008-MUS-75)(2008-HCI-128)，情報処理学会，2008
[15]http://www.parallax.com/Default.aspx?tabid=295
[16]http://www.parallax.com/tabid/407/Default.aspx
[17]http://nagasm.suac.net/ASL/Propeller/
[18]http://www.arduino.cc/
[19]http://nagasm.suac.net/ASL/Arduino/

System 項目	AKI-H8	Arduino	GAINER	Propeller
CPU	Hitachi H8/3048 32bits	Atmel ATmega 8bits	Cyperss 8bits CY8C29466	32bits * 8 CPU
clock	16/25 MHz	16 MHz	12 MHz	80 MHz
RAM	4K bytes	1K bytes	2K bytes	32K bytes
EEPROM	128K bytes	16K bytes	32K bytes	32K bytes (external only)
Power Supply	+5V	+5V	+5V	+3.3V
IDE	MS-DOS batch	Processing like	Max/MSP Flash Processing	original IDE
Language	Assmbler C	C	Max/MSP Flash Processing	Spin Assembler
PC interface	RS232	USB	USB	USB
Standalone	○	○	×	○
Serial Ports	2	1	4 (max)	8 (max)
A/D	12bits / 8ch	10bits / 6ch	14bits / 12ch	16bits / 28ch(max)
Audio D/A Out	100KHz 8bits 2ch	6ch PWM	×	44.1KHz 16bits 14ch(max)
Video Out	×	×	×	NTSC/PAL 2ch (max)
Character/Font Table	△	×	×	○
inter process communication	interrupt / polling hand-shake	polling	polling	shared memory polling
fast response	◎ interrupt	×	×	◎ parallel CPU
MIDI Out	○	○	○	○
MIDI In	○	×	×	○

Table.1 Cmparison table of 4 systems