並列処理プロセッサ“ Propeller ”による
スケッチング・プラットフォーム

長　嶋　洋　一

静岡文化芸術大学研究紀要抜刷
第10巻　2010年 3 月

並列処理プロセッサ "Propeller" によるスケッチング・プラットフォーム

Sketching platform with the "Propeller" parallel processor

長嶋　洋一
デザイン学部メディア造形学科

Yoichi NAGASHIMA
Department of Art and Science, Faculty of Design

マイコンシステムの高性能化とオープンソース・ソフトウェアの潮流が生み出した「フィジカル・コンピューティング」に関して、工学的な専門知識をブラックボックス化した、デザイン教育やメディアアートのためのプラットフォームを目指している。本稿では、この分野の世界的な傾向の解説とともに、SUAC研究科長 / デザイン学部長特別研究として進めてきた実験について報告する。合わせて、2009年に発表参加した国際会議の参加報告を行う。

This is a report of physical computing to which people recently pay attention in design process field and media-arts field. I introduce and discuss (1) trends in this field, (2) platform project in SUAC, and (3) special approach with the "Propeller" parallel processor.

1．はじめに

筆者は本学において、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。本稿では、マイコンシステムの高性能化とオープンソース・ソフトウェアの潮流から最近注目されている「フィジカル・コンピューティング」に関して、工学的な専門知識をブラックボックス化した、デザイン教育やメディアアートのためのプラットフォームの実現を目指した研究について報告する。まず、この分野の世界的な傾向の解説とともに、SUAC研究科長 / デザイン学部長特別研究として進めてきた実験について報告する。合わせて、2009年6月から9月にかけて発表参加した3つの国際会議(NIME09、Sketching09、ICEC2009)の参加報告を行う。なお、筆者のこれまでの関連研究/活動等の報告については、文献[1]に70件の参考文献リンクを置いて本稿では省略したので、そちらを参照されたい。

2．「作ってみよう」文化の発展と問題点

工業化社会の産物として、あらゆるものがメーカに製造され販売されている。ネットショッピングと宅配業の普及により、誰でも居ながらにしてクリック一つで欲しいものが手に入れられる時代である。その一方、健康な文化的反動としてか、「作ってみよう」がビジネスとなってきた。高度経済成長時代以前であれば「自分で作る」というのは「高くて買えない」からであったのが、現代では「自分でわざわざ作りたい」というモチベーションに変化したのである。メーカが製造販売する完成製品を購入するだけで満足しない人々に対して、あらかじめ設計済みの「完成されたパーツ」に分割された部品を組み立てる「(工作)キット」の文化は、「ミニ四駆」「電子工作」「ディアゴスティーニ」「大人の科学」などのビジネスとして成立してきた。

この状況は一般的には決して悪いことではないが、SUACデザイン学部の学生など、新しいものをデザインする若者には、必ずしも良い影響を与えているとは言い難い。新しいアイデアのシステムの試作やインスタレーション作品の制作において、100均ショップの粗悪な完成品の改造や工作キットの安易な流用によって、ほとんど画一的なモノを組み立てただけで「作った(デザインした)」と勘違いする学生には、デザイナとして明るい未来はとうてい期待できない。しかし理系でないデザイナに現代の膨大な工学的知識と経験まで求めるのは無理であり、産業界では若い設計技術者の慢性的な不足と熟練デザイナの高齢化に悩まされている、

3．フィジカル・コンピューティングとスケッチング

このような状況に対する新しいアプローチとして世界的に注目されているのが、マイコンシステムの高性能化とオープンソース・ソフトウェアの潮流によって登場した「スケッチング」の発想である[1]。パソコンを中枢に据える大掛かりなシステムでなく、いわゆるワンチップ・マイコンによる組み込みシステ

ムを実現するには、従来はマイコンごとに固有の言語や開発環境の熟達、そして電子回路の設計技術、さらには試作開発における信頼性・開発効率・コスト設計・量産化技術・検査技術など、多くのプロフェッショナル的な技術が備わっていることが必須であった。これがマイコン技術の高度化・複雑化の進展とともに１人のエンジニアで扱える規模を超えると、新しい打開策が求められるようになった。

ソフトウェア開発においては、オープンソース化という標準化がこれを解決する切り札となった。専業メーカが提供する高価でクローズドな統合開発環境に頼るのでなく、フリーウェァの精神で開発環境もソフトウェア部品も共有して皆んなでハッピーになろう、という発想である。ハードウェアにおいても、日本のマイコンメーカの囲い込みは破綻し、全面的な情報開示とともに「ハードウェアのオープンソース」という新しい発想に対応したプロセッサを提供するメーカだけが勝ち残りそうな状況である。

紙面の関係で具体例については文献[1]の資料リンクに譲るが、１９９１年から世界のメディアアート領域のプラットフォームとなり続けて進化しているMax/MSP/jitterを開発/実行環境とした上で、現実の物理世界とのインターフェースとしてI-Cubeのような MIDI周辺装置、さらにIAMASの小林茂氏の開発したGainerがスタンダードとなり、多くのインスタレーション作品やパフォーマンスの出現に貢献した。これらの作品では、その内部や背後に、Max/MSP/jitterの走るコンピュータが必要であり、高度で複雑な関係性(アルゴリズム)を容易に実現できる反面、小型軽量の組み込み機器のような形態は実現できなかった。

これに対して、過去には日本のお家芸であった、産業界向けの組み込みマイコン(4/8/16/32ビットCPU)を用いて、国内では秋葉原の秋月電子のAKI-80やAKI-H8などのカードマイコンが、海外ではMITが支援しつつBasicStampなどが登場して、パソコンを不要とした小型のスタンドアロンシステムとして、インスタレーションやパフォーマンス支援の機器を実現できるようになってきた。しかし日本では、メーカ単位で閉じた開発支援環境と開発言語体系に縛られて、世界に対して大きく遅れる状況となってきた。世界はUnix/Java/gcc/CC(クリエイティブコモンズ)などのオープンソースの文化が発展して、組み込みマイコンの開発支援環境として、フリーの統一的な環境が成長を続け、processingのような画期的な環境がフリーで活用でき、多くの作家が素晴らしい作品を生み出している。

日本の組み込みマイコンが産業界だけを向いて閉鎖的であったのに対して、世界ではこのオープンソースの文化が、ソフトウェアだけでなくハードウェアのオープンソース化を強力に支援した。イタリアの研究グループが開発したArduinoはそのもっとも成功した事例であり、限定された能力であるにも関わらず、多くの教育機関・研究機関の提供するライブラリ群や雑誌「make」による取材紹介は、新たにこの領域に取り組むデザイナの援軍となった。多種多様なArduinoクローンや変形Arduinoが登場し、ますます多くのインスタレーションがArduinoによってスタンドアロン化されていく傾向が続くだろう。重要なのは、この開発環境がprocessingとほとんど同一と思えるほど似ていることで、ここではプラットフォームの違いをソフトウェアが吸収するオープンソースの思想そのものを体感できる[2]。

後述する国際会議「Sketching」は、この流れをアーティストだけでなく、デザイナにもより一般化して普及させよう、という動機からスタートした。Sketchingの主催者Mike Kuniavskyは、世界の多くの企業のコンサンティング・デザイナとして活躍する中で、工学を専門としないデザイナが容易にアイデアを具体化/試作/実現(スケッチング)するための「オープン・ハードエウェア」を提唱した。ここに、教育者・研究者・アーティスト・企業経営者・技術者・ジャーナリストなどが賛同して集っているのが国際会議「Sketching」である。これは、コンピュータの内部でバーチャルに閉じていた世界を、現実の物理世界とインターフェースさせて拡大させる、という「フィジカル・コンピューティング」の思想とも共通する部分が非常に多く、

世界から注目されている新しい領域なのである。

4. シングルタスクマイコンの限界とパラレル CPU "Propeller"

筆者は1985年頃からこれまで、基本的にはシングルタスク組み込みマイコンをプラットフォームとして活用してきた。多くの製作事例は回路図やソースコードとともに全てWebで公開しているので、見ず知らずの作家や学生から質問のメイルを受けたり、レクチャー / ワークショップ講師の依頼が突然に届いたり、プロジェクトの予算が付いたので特別なシステムの開発を依頼する、というようなオファーも多数あった。これらについては[3][4]などを参照されたい。

このようなシステム開発において、筆者が文献[5-7]などで指摘しているのは、スタンドアロンシステムの中核となるボードマイコン（AKI-H8 や PIC や Arduino）のファームウェアを書き込む内蔵 Flash メモリの「繰り返し書き込み」の問題点である。昔であれば組み込みマイコンのプログラムはシステム内に紫外線消去型 EPROM のソケットがあり、ここに最終的なプログラムを搭載したが、ここ10年以上の主流は、CPU内部の ROM 領域が FlashEEPROM になっていて、CPUのシリアルポートを経由してここにプログラムを書き込む、という方式である。例えば、開発中の実機に搭載された AKI-H8 の RS232C ポートから ROM プログラマボードとUSB-RS232Cアダプタを経て、開発ホストである Windows パソコンと接続する。このようにオンボードマイコンに直接、開発中のプログラムをダウンロードできることで、システム開発中のEPROMソケットの抜き差しや「ROMエミュレータ」のソケット接続の手間から解放された。

ここで問題となるのは、カタログデータではCPUの内蔵FlashEEPROMの繰り返し書き込み回数は10万回以上とされているものの、CPU個別のばらつきや周辺ノイズ環境などにより、実際には100回程度のプログラム書き込みでもエラーが起きることがあった、という経験的事実である。ICE(インサーキットエミュレータ)などを使わずに、開発プログラムをシンプルなものから漸次拡張していき、EEPROM に書き込みしてはリセットしてデバッグする、という開発プロセスにおいて、完成度を上げるためには何度でも繰り返せることが必須である。ところが「書き込み回数が増えるとエラーが起きることがある」とか「もう書き込めない(使えなくなる)段階がいずれ迫ってくる」という状況は、プログラマにとって水面下でのプレッシャー / ストレスとなる。実際に筆者の経験でも、ある程度まで完成したプログラムについては、そこから繰り返し細かく改訂するというよりも、システムの使い方で対応するとか、MIDIを経由してやりとりするホストシステム(Max/MSP など)の側のプログラミングによって細かい改良の部分を吸収する、という戦略は、ほとんど無意識的に採用している「常識」である。

また、一般にこのようなスタンドアロンシステムでは、「パネルスイッチや外部センサからの入力」「パネルLED/LCDやモータ制御などの出力(タイミング制御)」「開発ホスト PC との通信(開発中のみ)」「システムホストとの通信(MIDI/ イーサネット / 無線)」「デバッグ用の動作表示、数値表示」など、多数のタスク(ジョブ)が必要になってくる。最終的には実機で必要なくなる機能であっても、デバッグの過程ではわざわざハードウェアを試作増設して、ファームウェアの動作確認、数値データなどの外部出力によって実現している動作モニタリングは、スケッチングにおいて開発期間の短縮と完成度の向上に大きく寄与するので、ちょっとした回り道であっても定番のテクニックである。筆者の場合には、Max/MSP/jitterを作品全体のホストとすることが大部分であるために、どんなシステムであってもまずは MIDI 入出力ポートを設置して、開発やデバッグの際には、システムの内部状態を MIDI 出力したり、Max からの MIDIを受けて動作切り替えなどを行う。最終的に MIDI とは無縁のスタンドアロンシステムの開発においても、である。

ここで問題となるのは、MIDI入力には割り込みが必要であり、またタイミング設計のために CPU 内部のタイマによるタイマ割り込みを使うことも多いことである。ポーリング

とかハンドシェイクのようなシンプルなジョブの積み重ねだけで必要な処理を実現できれば問題ないが、小型軽量低価格を目指す組み込みマイコンではクロック周波数やメモリ(RAM)容量に限界があり、Cだけでなく時にアセンブラも組み合わせた、本格的なプログラミングスキルを要求することも少なくない。インスタレーション作品とは本質的にリアルタイム・マルチタスクシステムだからである。

ここに、前述のCPUプログラム書き込み回数の制限という要因が組み合わされることで、事態は非常に困難な状況となる。つまり、仕様としての機能と開発 / デバッグ用の機能をシステムに盛り込むことは、タイミングも精度も異なる多種のタスクを、割り込みを必要とするレベルで調停しつつリアルタイムに実現するという要請であり、転送一発でそう簡単には実現できない。デバッグ資源 / 機能を活用しつつ、何度もトライすることで完成度を上げていく。ところが、このトライ回数に暗黙の上限が制約として加わることで、開発そのものの足を引っ張るという二律背反の状況となるのである。この世界になかなか若手エンジニアが入ってこれない理由の一つは、ベテランなら経験的に分かっているこのあたりの勘所に不慣れなためであろう。

文献[8]で紹介したように、筆者は2008年に、BasicStampで有名なParallax社の提供するPropellerプロセッサと出会い、その概念や細部を解析するとともに、いくつかの実験システムを試作し、さらにインタラクティブなインスタレーション作品を実際に制作して公開展示した[9-10]。このシステム"DodecaPropeller"では、Propellerチップを13個用いて、縦3台・横4台、計12台のビデオモニタにそれぞれ個別のリアルタイムCGを生成表示し、その12個の画面が来場者からの働きかけや会場の環境音に反応してダイナミックに変化するが、従来のようにホストPCを必要とせず、たった1枚の基板でシステムが完結している(本稿末尾の図3のFigure 6)。このPropellerチップは内部の8個のCPUがハードウェア的に並列処理を実行するために、ソフトウェア開発において「割り込み」の概念が不要である。Cogと呼ばれる各CPUのタスクを起動すれば、相互のタイミングを設計しなくても、各Cogはそれぞれの処理を中断なく実行する。これは新楽器のように、本質的にマルチタスクなシステム開発(スケッチング)においては、非常に有効なプラットフォームである。

さらにPropellerでは、開発用ホストPC(高級言語spin/アセンブラを共存可能)からUSB接続されて開発中のプログラムを書き込むが、そのターゲットとしてPropellerの内部RAMを指定できる。つまり開発中はPropellerの内部RAMに転送したプログラムで実験 / デバッグし、最終的な完成プログラムについては、ターゲットをPropellerチップの隣に置いたFlashEEPROMに変更するだけでよい。リセット時にPropellerはまずホストPCとの接続を捜し、接続があれば開発中として内部RAMのプログラムで走る。接続されていなければ外部EEPROMから内部RAMに自動転送したプログラムで走る。この機能の意味するところは非常に重要で、RAMであるから開発中は「何度でも無限にリトライ出来る」マイコン、という事である。細かい実験や試行錯誤や修正はRAMで行い、ある段階まで完成したところでソースを保管するなり外部EEPROMに書き込む、というプロセスにより、従来のマイコンとは桁違い(無尽蔵)のテスト繰り返しを可能とする。

筆者はPropellerで外部とMIDI通信するモジュールをオリジナル開発し、Max/MSPと合わせた開発環境を実現した[8]が、さらにPropellerでは内部のCogのうち2個を使い、40ピンの入出力ピンのうち3本の外部に3本の抵抗を接続するだけで、NTSC/PALビデオ信号を簡単に生成出力することができる。Propellerコミュニティにより Webフリー公開されているグラフィックドライバなどをブラックボックスとして呼び出すだけで、デバッグのために「ビデオモニタに10進/16進形式での数値表示」やOpen-GLにも似た「カラーグラフィック表示」も容易に実現できる。これは、システムとして必要なマルチタスクを実機として動作させながら開発する中で、デバッグのために簡単に内部動作をモニタできる、いわば「内蔵デバッグ装置」のように活用できる。まさに試行錯誤の開発に好適なプラットフォームであると実感している。

5．ICEC2009 Tutorialからの紹介

ここで、関連したトピックとして、筆者が2009年6月から9月にかけて発表参加した3つの国際会議(NIME09、Sketching09、ICEC2009)の参加報告を行う。紙面の関係で、過去の紀要で紹介した2つの国際会議(NIMEおよびSketching)については簡単な報告にとどめ、ICEC2009でのチュートリアル発表について重点的に報告する。

２００４年に欧米以外としては初めてSUACで開催した[11]国際会議NIMEは、2009年は6月に米国ピッツバーグのカーネギーメロン大(CMU)で開催された[12]。筆者はPropellerを新楽器や新インターフェースのプラットフォームとして活用する、という提案を発表した[13]。本稿の末尾3ページ(図1-図3)に紹介したのは、そのプレゼンテーション・ポスターである。物理コンピューティングやスケッチングの専門家が集まる国際会議Sketchingは、2009年はロンドン市内のUniversity College of London (UCL)で開催された[14]。筆者は2008年に発表したSUACでのインスタレーション作品の事例紹介に続いて、2008年から2009年に新たに取り組んだ事例紹介[15]とともに、新しい学生作品「はやくスシになりたい」(野口佳恵)をロンドンに持参して、世界初演となる展示発表を行い絶賛された。

国際会議ICECとは、エンタテインメントコンピューティングの国際会議であり、その発端は2002年に幕張で開催された国際エンタテインメントコンピューティングワークショップIWEC2002である。筆者はこのIWEC2002での発表以来の参加となったが、研究発表としてでなく、Propellerを実験システムのプラットフォームとして活用するためのチュートリアル(レクチャー)として応募し採択され、パリ・Conservatoire National des Arts et Metiers (CNAM)で開催されたICWC2009の初日に、「Parallel Processing Platform for Interactive Systems Design」というタイトルの、全6時間の1日講習を行った[17]。受講者は、東欧やアジア(韓国)など世界各地から集まった10数名であり、SUACでのメディアアートの事例紹介から始まって本格的なシステム開発に関するノウハウまで、熱心に受講していたのが印象的であった。以下、筆者の講習メニューの項目ごとに、ごく簡単に内容を紹介しておく。

5-1 Abstract

このチュートリアルの募集としても使われた「概要」は以下である。(原文のまま)
This lecture is intended for the designer of the entertainment system. As a platform of an interactive entertainment system, it is popular to combine MIDI sensor and GAINER with Max/MSP. It is also general for a simple system such as toys to use the 1-chip microcontroller such as Arduino. However, it is very difficult in the multimedia system to combine video signal, CD-audio output and many sensors without depending on PC. You will learn a new platform with parallel 8 processors in one chip. As a result, complex processing can be achieved with the standalone system. Propeller's 8 processors can operate simultaneously, either independently or cooperatively, sharing common resources. I introduce this unique processor and discuss about the possibility to develop interactive systems and smart interfaces in media arts, because we need many kinds of tasks at a same time in entertainment computing. We will also show that the design of the Propeller system is effective for the programming education.

5-2 Installation/Performance - Examples

エンタテインメントコンピューティングでのシステム開発の事例紹介として、インスタレーション作品の本質について、インタラクションのデザインについて、パフォーマンスやメディアアートとの関係について整理した。

5-3 Platforms for Installations
インスタレーションのプラットフォームについて整理した。検討した項目は「アーキテクチャ」「ソフトウェアと実行環境」「ハードウェアとファームウェア」などである。具体的な例としてI-Cube・PIC・BasicStamp・AKI-H8・Gainer・Arduino・Propellerを紹介した。

5-4 The "Propeller" Processor
このチュートリアルの主役であるParallax社の Propellerプロセッサについて、アーキテクチャや技術的な特長について解説した。

5-5 Propeller System Design Lecture
この部分が本チュートリアルの根幹である。Propellerを実際に使ってシステムをスケッチングするための方法、支援環境、オープンソースのライブラリの活用、各種のインターフェースの実例紹介(筆者が開発し公開している世界初のアプローチもここで紹介)、周辺回路の拡張テクニックなどを紹介した。以下がその各項目の見出しであるが、これ以上の詳細は紙面の都合により省略する。

5-5-1. Development environment
5-5-2. Experiments with examples
5-5-3. MIDI I/O interface
5-5-4. Video signal generaton
5-5-5. Audio signal processing
5-5-6. Sensor Interfaces
5-5-7. Serial Communication Interfaces
5-5-8. Expanding Parallel Output Ports
5-5-9. Expanding Parallel Input Ports
5-5-10. D/A (voltage) Output
5-5-11. A/D (voltage) Input

5-6 Case studies with Propeller
具体的な Propellerを活用したケーススタディとして、以下の最新の3つの話題を紹介した。

5-6-1. uOLED-96-PROP
Propellerを搭載した超小型カラーディスプレイモジュール「uOLED-96-PROP」について紹介し、実際にこれを使用して製作したシステム2種類をMIDIインターフェースで接続し、加速度センサによって傾きを検出して画面上のグラフィクスを描画しつつ、通信ネットワークで情報交換するデモを実演した。

5-6-2. Dodeca Propeller
MAF2008で発表したインスタレーション作品「Dodeca Propeller」について詳細に解説・紹介した。

5-6-3. 4 Mouse Interface (Propeller+Gainer)
メディア造形学科学生(見崎央佳)の新しいインスタレーション作品「OTOkakekko」のために制作した、「4つのPS/2マウスの挙動をGAINER経由でホストに伝える」インターフェースは、Propellerの4つの内部CPU(Cog)がそれぞれのマウスと通信する。このデモビデオを持参して詳細について報告した。

5-7 Future work
まだ未公開というより制作中である「The New Interface for Musical Expression (32 channels Thelemin)」について、渡欧の3日前に録画したデモビデオを持参して紹介した。これはMAF2009でデモ発表するとともに、筆者が作曲し作品として公演するのは2009年12月5日(国立音大)の予定である。

6. おわりに

最近注目されている「フィジカル・コンピューティング」について、デザインプロセスおよびメディアアートとの関係に注目して、特にパラレルプロセッシング CPU "Propeller" について、個々のタスクを増設・記述するだけで、タイミング設計なしに、時分割処理による高いパフォーマンスを実現できた実例を紹介した。全てをマスターするには本学大学院の2年間でも不足するものの、効果的な汎用プラットフォームの助けによって、学生の作品制作レベルは年々向上している。今後、ますます発展するであろうこの領域において、メディアデザイン教育のための新しい

汎用プラットフォームの実現に向けて、さらに検討していきたいと考えている。

参考文献 / リンク

[1] 長嶋洋一, デザインプロセスにおける「スケッチ」と物理コンピューティング, 静岡文化芸術大学紀要・第9号 2008年, 静岡文化芸術大学, 2009
[2] http://nagasm.suac.net/ASL/Arduino/
[3] http://nagasm.org/
[4] http://1106.suac.net/
[5] http://nagasm.suac.net/ASL/sensor01/
[6] http://nagasm.suac.net/ASL/original/
[7] http://nagasm.suac.net/ASL/mse/
[8] http://nagasm.suac.net/ASL/Propeller/
[9] 長嶋洋一, Propellerを使った体験型アート作品の製作(前編/後編), トランジスタ技術 2008年9月号/10月号, CQ出版社 (2008).
[10] http://nagasm.suac.net/ASL/12Propeller/
[11] http://suac.net/NIME/
[12] http://nime2009.org/
[13] http://nagasm.suac.net/ASL/paper/NIME09_2.pdf
[14] http://www.sketching09.com/
[15] http://nagasm.suac.net/ASL/paper/Sketching09.pdf
[16] http://www.entertainmentcomputing.org/icec2009/
[17] http://nagasm.suac.net/ASL/ICEC2009/

Parallel Processing System Design with "Propeller" Processor

Yoichi Nagashima (nagasm@suac.ac.jp)
Shizuoka University of Art and Culture

Abstract
This is a technical and experimental report of parallel processing, using the "Propeller" chip. Its eight processors (cogs) can operate simultaneously, either independently or cooperatively, sharing common resources through a central hub. I introduce this unique processor and discuss about the possibility to develop interactive systems and smart interfaces in media arts, because we need many kinds of tasks at a same time with NIME-related systems and installations. I will report about (1) the Propeller chip and its powerful IDE, (2) external interfaces for analog/digital inputs/outputs, (3) VGA/NTSC/PAL video generation, (4) audio signal processing, and (5) originally-developed MIDI input/output method. I also introduce three experimental prototype systems: (a) 1 chip standalone system with MIDI input, audio synthesis, sensor inputs and double NTSC outputs, (b) a compact display module for CG generator and MIDI monitor, and (c) a compact sound installation system for 12 video display monitors with 12 Propeller chips generating real-time CG and 1 master Propeller chip.

Keywords: Propeller, parallel processing, MIDI, sensor, interfaces.

1. Introduction

Propeller [1] is supported by Parallax Inc. With its internal eight processors, 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. A shared system clock keeps each cog on the same time reference, allowing for true deterministic timing and synchronization. (see Figure.1) We can use two programming languages: the easy-to-learn high-level Spin, and Propeller Assembly which can execute at up to 160 MIPS (20 MIPS per cog). There is a popular technique "Interrupt" to realize multi-task with all CPU. However, Propeller doesn't have the "Interrupt" because parallel processing is controlled by its special hardware. Its resources - common memories (32KB RAM /ROM) and 32 external I/O pins - are automatically assigned to round-switched cogs. We can easily make parallel processing software for Propeller by the smart and powerful IDE, without special consideration for synchronization. Because I have no enough space to introduce more both Propeller's languages and Propeller's IDE here, please refer my analyzing/experiments report [2]. This website was only in Japanese, but the English version is now available.

2. Propeller interfaces / MIDI input/output

Propeller has 32 I/O pins that can be accessed by each cog with double or more monitoring and overwriting. Each cog has special timing circuits for counter/timer modes. Propeller can deal serial communications like MIDI only by software, without special hardware like UART. There is a sample MIDI-in object in the Parallax web page [3], but I arranged and developed the universal MIDI-in/MIDI-out module [2]. This module deals MIDI information with deep Rx/Tx FIFO buffers in common memory in the chip, so it is easy to make intercommunication of each cog. Figure 2 shows the original circuits with MIDI I/O, audio D/A and NTSC video output (described later).

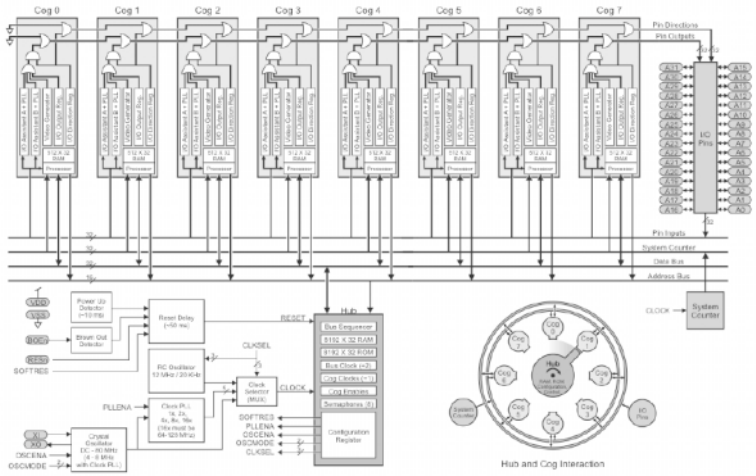

Figure 1. Block Diagram of "Propeller" processor.

3. D/A output and A/D input

Propeller can generate an analog output signal by PWM converter with external 2 capacitors and 1 resistor. Thus Propeller can generate easily 44.1 KHz sampling, 16 bits stereo audio signal with 2 external pins and 1 internal cog. We can also get A/D input with reference D/A output by 1 internal cog and 2 pins, using Parallax's sample program. The Propeller's cog is fast enough for CD quality sampling conversion. I discovered and confirmed that the assembly language of Propeller was specially designed to achieve audio signal processing compactly and efficiently.

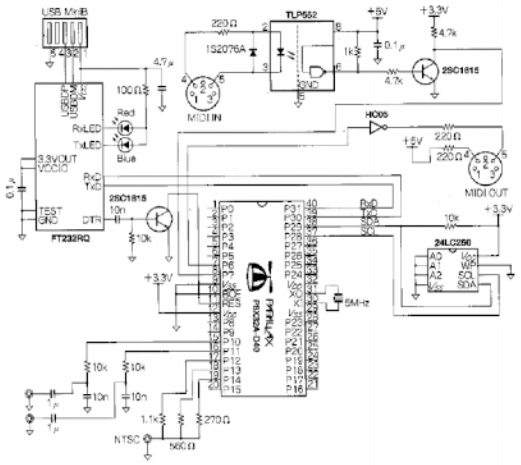

Figure 2. The original circuits with MIDI I/O, audio D/A and NTSC video output.

4. Analog (RC) input / NTSC-PAL-VGA video output

Because each cog of Propeller has an individual timing circuit in it, RC-type (time constant circuit) A/D conversion is simpler to construct. We can easily achieve high accuracy and high speed A/D by the specification of Propeller (80MHz, 32 bits). We can use many types of sensors: CdS, strain-gauge, piezo, static-electricity, carbon-rubber, electric-capacitor, etc. Surprisingly, Propeller can generate the video signal of NTSC/PAL/VGA in the background, with 1 cog for graphic driver and 1 cog for video signal D/A with only 3 external resistors. So, we can produce many multimedia systems with using only 2 cogs of Propeller, and we can use remaining cogs for 6 individual parallel tasks.

5. Double NTSC output / 4-Mouse Interface

After this research, I produced four experimental systems, and the 4th one is a media-installation work.
Using 4 cogs for double NTSC video outputs, I developed a multi monitor MIDI-CG system. However, by the limitation of Propeller's internal memory, the double CG drawing-mode were only "storage-display like" without double-buffer computing (Figure 3).
Figure 4 show the new interface system for a new installation work with 4 * PS2 mouse, NTSC Video output and the interface with GAINER to connect Max/MSP/jitter platform and FLASH platform.

Figure 3. Double NTSC video, MIDI IN/OUT system.

Figure 4. 4 * PS2 Mouse, Video out, GAINER I/F system.

6. Propeller Compact Display Module

Next, I produced a compact display module for CG generator and MIDI monitor. The small module is supported by Little PCB Solutions [4], and I developed the prototype Compact Display Module. The system generates 3 patterns of realtime CG and MIDI display by Hexadecimal format (see Figure 5).

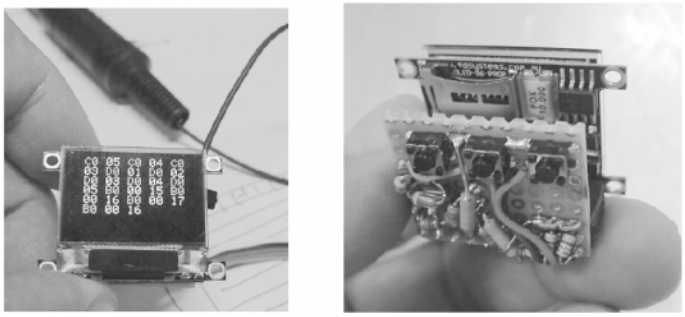

Figure 5. Propeller Compact Display Module.

7. "Dodeca Propeller"

The newest work with Propeller is the installation work called "Dodeca Propeller". This was presented at Media Art Festival 2008 at SUAC in Japan in December 2008. Figure 6 shows the system board and there are 13 Propeller chips on it. We have two huge display systems that arrange 12 large-scale video monitors in SUAC. "Dodeca Propeller" was designed for this display system, so the output is 12 NTSC video lines. 12 display Propeller run the real-time CG generating program. The "controller" Propeller works as: MIDI control (foot switch sensors) receiver, individual real-time On/Off switcher for each 12 screen, and the sound sensor in the gallery hall to control display patterns. As an important point, PC doesn't exist in the system that achieves this complex real-time generation of 12 screens CG and the interaction with the sensors. I developed this system with 2 undergraduate students. It was very easy to make CG programs of Propeller for the students, and I think that Propeller is a very good platform for education of students' programming. Figure 7 shows the presentation of "Dodeca Propeller".

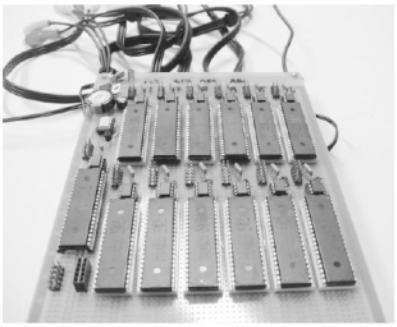

Figure 6. System Board of "Dodeca Propeller".

Figure 7. Presentation of "Dodeca Propeller".

9. Conclusions

This is my first report of the "Propeller" chip. I think (1) it has deep possibility to design interactive systems, and (2) it is good for education in computer programming. I will report next step in the future with some projects in media arts.

References

[1] Propeller General Information, Parallax Inc., http://www.parallax.com/tabid/407/Default.aspx

[2] Yoichi Nagashima, Propeller Diary, http://nagasm.suac.net/ASL/Propeller/index.html

[3] Tom Dimock, Propeller MIDI in objects, http://obex.parallax.com/objects/229/

[4] OLED-96-PROP, Little PCB Solutions, https://www.littlepcbsolutions.com/uOLED-96-PROP.html