環境反応型イルミネーション装置の開発
～GCを備えた高級言語による組み込みマイクロプロセッサ用開発環境の作成～

材料技術室　石川 隆朗

Development of Illumination Device that Reacts to Environmental Incidents.
～ Development of Framework for Embedded Micro Processors with GC ～

ISHIKAWA Takaaki

光，音，接触等，周囲の環境の変化にインテリジェントに反応するイルミネーション装置の開発を目的とした。そのために，GC(Garbage Collector)を持つ高級言語による開発フレームワークを作成した。本処理系はScheme言語をベースとしており，柔軟で効率的なプログラム開発を行うことができる。また，マルチタスキング等のOS機能も保持している。本開発環境を用いることで，様々な近代的なプログラミング技法が利用可能になり，n-queen問題の解法器，リバーシゲーム，お茶運びロボットを非常に高効率に開発することができた。

１．はじめに

イルミネーションは様々なスポットを彩り，我々の目を楽しませてくれる。イルミネーションそれ自身が観光の目的となり，観光客を呼ぶこともある。

従来のイルミネーションは，目的を静的に照らすのみか，時間とともに変化するのみであった。昨今では，ディジタル技術の進歩により，マイク，光度計等のセンサを用い，周囲の環境の音，明るさ等を読み取り，それらの外的要因に対してインテリジェントな反応を示すイルミネーション装置も出現し始めている。

本研究では，これらインテリジェントなイルミネーション装置の用途を商店の店先，個人の家庭まで広げることを目的とした。そのため，装置の制御器には以下の事項を要求した

・小型軽量であること
・電源の自由度を高めるため，低消費電力であること
・24時間ノーメンテナンスで風雪に耐え稼働すること

従来のインテリジェントなイルミネーション装置は制御にPC，それに類するものが使われることが多い。それらはそれなりに筐体が大きく，消費電力も数十ワット程度要求される。部品点数も多く，ファンや固定ディスク等，故障の原因となる機械動作部分もある。

組み込み用マイクロプロセッサは，ワンチップにROM，RAM，周辺インターフェースが内蔵されており，非常に小型，軽量である。消費電力はミリワットオーダーであり，多数のLEDを発光させることを考えれば無視できるレベルである。部品点数が少ないことは信頼性の向上につながる。

イルミネーション装置はその特性上，顧客に合わせた一品もの，もしくは，少量生産品になる場合が多く，加えて流行り廃りが激しい。そのため，ユニークなアイデアを短期間で実装する必要がある。また，試作品を短期間で作成し，試作品を改良するPDCAサイクルを高速に回すラピッドプロトタイピング開発が有効である。

しかし，組み込み開発は一般的にGC(Garbage Collector)のないC/C++言語を用いて行われ，開発効率が高いとは言えない。

そこで，本研究ではGC等のプログラミング言語としてモダンな要素を持つ高級言語による組み込みマイクロプロセッサ開発フレームワークの作成を行うこととした。本フレームワークにはマルチタスキング等，OS機能も含むものとした。本フレームワークを用い，インテリジェントな反応を示すイルミネーション装置の試作を行い，その有用性を示すことを目的とした。

２.フレームワーク概要

本開発フレームワークの概要を示す。

本フレームワークはルネサスエレクトロニクス株式会社のRXシリーズマイクロプロセッサを搭載したGR-SAKURA(株式会社若松通商製)，RX621マイコンボード(株式会社秋月電子通商製)，RX210マイコンボード(株式会社秋月電子通商製)で動作を行った。

２.１　言語処理系

本処理系はAtsushi　Moriwakiの開発したmini-schemeを参考に開発した。Scheme言語のサブセット言語である。サブセット言語ではあるが近代的なプログラミング言語が持つ以下の要素を持つ。

・GC(Garbage Collector)
・レキシカル・スコープ
・関数が第一級のオブジェクト
・クロージャ
・末尾呼び出しの最適化

本処理系はHaskell言語で記述されたコンパイラとC言語で記述されたVM(Virtual　Machine)からなる。Scheme言語で記述されたプログラムが与えられると，コンパイラによってバイトコード列に変換される。このバイトコード列を組み込みマイクロプロセッサ上のVMで動作させる。バイトコード列はmini-schemeのSECDRマシンと同様のアルゴリズムの動作を行うよう生成される。

扱えるデータは32bit整数，シンボル，コンスセル，クロージャであり，それらは，コンスセルで構成されたリストで管理される。リンクトリストとしてメモリ空間に配置されるため，一般的なスタックポインタの増減を利用するスタックと異なり，連続したメモリ空間を要求せず，オブジェクトは不連続に，自由に配置される。

２．２　OS機能

本処理系はマルチタスク等のOS機能も担っている。

マルチタスクの手法は協調型マルチタスクである。一般的な協調型マルチタスクとはことなり，言語処理系にマルチタスク処理が組み込まれているため，粒度の細かいタスク切り替えが行われ，また，タスク切り替えを意識せずにプログラムを作成することができる。

３　試作イルミネーション装置

３．１　昨年度までの試作品

本研究は平成 25 年度に予備研究が行われ，平成 26 年度，平成 27 年度の 2 年間にわたって行われた。

平成 25 年度には 5×5 の LED マトリクス上に 5-queen 問題の解を表示するイルミネーション装置を作成した。(写真 1)

・n-queen 問題の解法器
・5×5 の LED マトリクスのダイナミック点灯
・チャタリング除去を行うキー入力処理

を，並行で行う。以上の処理を 130 行で記述できた。

平成 26 年度にはリバーシゲームの作成を行った。(写真２)　AI を搭載し，マイクロプロセッサと人間で対戦するものである。コンピュータの戦略にはミニマックス法を用いた。

・ミニマックス法の解法器
・8×8 のフルカラーLED マトリクスのダイナミック点灯
・チャタリング除去を行うキー入力処理

を並行で行う。以上の処理を440行で記述できた。開発期間は3日，8時間程度で開発ができた。

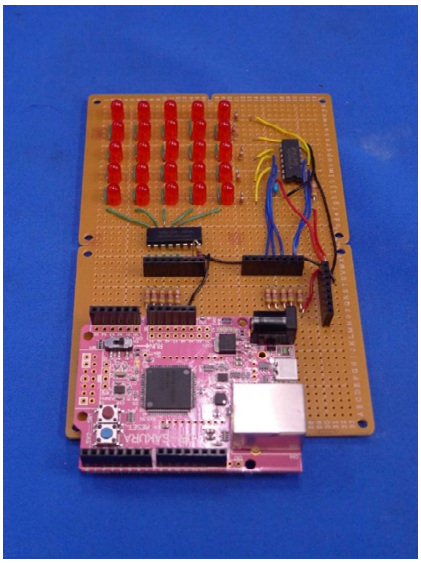

写真１ 5-queen解法器

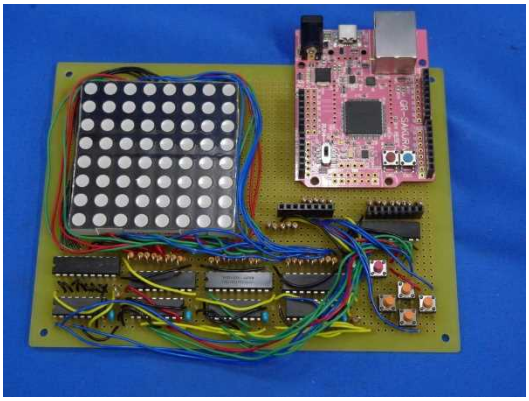

写真２　リバーシゲーム

３．２　ジェネレータを用いたプログラミング

ジェネレータは一回呼び出すことによって一つの値を返す関数である。ただ一つの値を返すことが保証されているため，プログラムを簡潔に書くために役立つ。Schemeは言語の機能としてジェネレータを持っていないが，情報を保持した手続きであるクロージャを使うことによりジェネレータを作成することができる。

作例としてエラトステネスの篩を用いた素数生成器を作成した。

簡単にエラトステネスの篩を説明する。2から始まる数列を用意する。数列から先頭の数を取り出す。最初は2である。数列から2の倍数を除去する。(篩にかける)。次に数列の先頭をとると3となり，同様に数列から3の倍数を除去する。このように，数列の先頭から数字を取り，その数字の倍数を除去するという操作を繰り返し行うアルゴリズムがエラトステネスの篩である。

ジェネレータを用いエラトステネスの篩を行うには以下の3つのジェネレータが必要となる。

1. 2から始まり，一つずつ大きくなる数列を作るジェネレータ
2. 数と数列を受け取り，与えられた数列から与えられた数の倍数を除去した数列を返すジェネレータ
3. 保持しているジェネレータから数を一つ取り出し2番のジェネレータをかぶせ，それを保持するジェネレータ

1番のジェネレータが最初の数列，2番のジェネレータが篩に相当し，3番のジェネレータで篩を重ねる操作を行う。

これらのジェネレータをクロージャを用い，実装した。

ジェネレータを使わない場合，現在注目している数，今まで見つかった素数等を陽に保持する必要があるが，ジェネレータを利用した場合，それらは隠ぺいされ，3番のジェネレータのみ保持すればよい。また，1番，2番，3番のジェネレータはそれぞれ独立して記述でき，プログラムのフローは単純になり，簡潔に記述できる。

３．３　お茶運び人形の作成

エレキギターの音でコントロールされるお茶運び人形を作成した。(写真３)音の周波数の高低を判断し，動作を行う。

外部の音をエレクトリックコンデンサマイクで読み取り，オペアンプで増幅し，周波数をカウントし，モーターのコントロールを行う。

ソフトウェア部分は非常に簡潔に記述できた。プログラムの修正も容易であり，ラピッドプロトタイピング開発のサイクルを高速に行い，作成することができた。

写真３　お茶運び人形

４　考察

４．１　メモリ管理

マルチタスクシステムは各タスクごとに，文脈情報(プログラムコンテキスト，プログラムが記憶しておくべき情報)を保持しなければならない。連続するメモリ領域を要求する手法でメモリオブジェクトを管理する手法の場合，それぞれのタスクごとに，そのタスクが要求する最大のメモリ容量を割り当てなければならない。また，タスクの要求するメモリ量を見積もることは一般的に難しく，

余裕をもって割り当てる必要がある。本処理系はオブジェクトを不連続に配置することができるため，全メモリ領域を使い尽くすまで，それぞれのタスクが自由にメモリ領域を獲得することができる。メモリの利用効率が高いことは，組み込みマイクロプロセッサ開発で大きなアドバンテージとなる。また，タスクごとのメモリ管理を行わなくてよいため，迅速に開発を行うことができる。

４．２ REPLの利用

Scheme言語はLISP言語の一方言である。LISP言語はFortran言語，COBOL言語とともに最古の言語のひとつである。長年の歴史で積み重ねられたその開発文化資産は大きい。

LISPに連なるプログラミング言語の開発手法としてREPL(Read-Eval-Print-Loop)がある。REPLを使うことによってプログラムの小片から試行錯誤して組み立てることができ，非常に高効率に開発できる。

本処理系は現在ではインタプリタモードが無く，本処理系自身でREPL開発を行うことはできないが，Gauche等のREPL開発に対応したScheme言語処理系を使うことにより，ハードウェアに直結しない部分の開発はREPL開発で進めることができる。

試作イルミネーション装置の作成は以下のように行われた。

1. Gaucheを用い，AI等の部分をREPL開発で行う。
2. PC上で本処理系を動作させ，ハードウェア制御部分の開発を行う。
3. 実機にインストールし動作の確認を行う。

４．３ 高級言語機能の活用

試作イルミネーション装置の開発において，本処理系の持つプログラミング要素が役に立った。

再帰はプログラミングに非常に有用な手法である。しかし，再帰処理は文脈情報を記憶する必要があるため，記憶領域が貴重な組み込み開発では避けられる。本処理系は末尾呼び出しの最適化が行われるため，不必要な文脈情報を記憶する必要がなく，効率的に再帰処理を行うことができ，抽象的な処理を効率的に開発することができた。

ジェネレータを利用したエラトステネスの篩では，関数(クロージャ)が第一級のオブジェクトであるため，作成した全てのジェネレータ(最初の数列と篩から出てきた数列)を区別することなく統一的に扱うことができた。ジェネレータにジェネレータを渡すということができたのも，関数が第一級のオブジェクトだからである。

ジェネレータを作成できたのは，手続きと情報のペアであるクロージャが利用できたからである。5の篩のジェネレータを作成することを例にとると，「5という数字オブジェクト」，「3の篩ジェネレータオブジェクト」という情報，「篩う」という手続きがクロージャによって一つにまとめられ，「5の篩」という扱いやすいジェネレータオブジェクトが生成された。

クロージャ等によってプログラムの複雑性を隠ぺいすることができたが，オブジェクトの存在，オブジェクトの寿命もプログラマから見えにくくしてしまう。GCがオブジェクトの存在の有無，オブジェクトの寿命を適切に管理することによって，プログラマはオブジェクトの生成と利用のみ注力すればよくなった。

C言語が持たない近代的な機能を利用することによって，組み込み開発の効率が高くなることがわかった。

５ 結言

平成25年度の予備研究，平成26年度，27年度の研究により，Scheme言語のサブセット言語，マルチタスク等のOS機能からなる組み込みマルチプロセッサ用開発フレームワークが作成された。その開発フレームワークを用い，5-queenの解法器，リバーシゲーム，お茶運び人形を題材とした試作イルミネーション装置の開発が行われた。本研究で作成された開発フレームワークを用いることで，ソフトウェア部分は非常に高効率かつ簡潔に作成でき，ハードウェアの作成に注力できた。プログラムの改変も容易で，ラピッドプロトタイピング開発のサイクルを効率的に回すことができた。

LISP言語のREPL開発は組み込み開発でも非常に有効であった。本処理系は現在，単独ではREPL開発を行うことができない。単体でもREPL開発が行えるよう環境を整備したい。また，簡易的なハードウェアシミュレータの開発も行いたい。