環境反応型イルミネーション装置の開発
～GCを持った高級言語による組み込み開発用マルチタスクフレームワークの開発～

材料技術室　石川 隆朗

Development of Illumination System that Reacts to Environmental Conditions
～ Development of Embedded Multitasking Framework Using Higher Level Language with GC ～

Takaaki ISHIKAWA

周囲の環境情報をセンサ等で読み取り，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作製を行う。

インテリジェントなイルミネーションを開発するにあたって，GC(Garbage Collector)を備えた組み込み開発用高級言語の開発を行った。本組み込み開発フレームワークはマルチタスク機構等OS機構も併せて備えている。

本研究で開発した処理系をルネサスエレクトロニクス株式会社製マイクロプロセッサRX63N搭載のGR-SAKURAマイコンボード上に実装した。本処理系を用い，オセロゲームイルミネーションを作成した。

本フレームワークを用いることによってオセロのソルバを437行で記述でき，3日で開発可能となった。

１．はじめに

社会のグローバル化により，国内の工場は海外に移転され，雇用の喪失が問題となっている。周辺他国と比較して高賃金である日本でものづくりを行うならば，日本で生産することによって，製品に価値が付与させることを顧客に示さなければならない。

様々な産業が海外に流出している中，海外から生産が戻ってきている産業がある。10 年前，PC は10 年前は小品種大量生産を海外工場で行うことにより，廉価な製品の販売が行われてきた。しかし,近年では顧客のカスタマイズ指向が高まり，多品種少量生産の製品を受注生産で作る需要が増えてきた。そのことにより,納品時間の観点から，生産が国内の工場に戻ってきている。

このように，カスタマイズ可能な製品の受注生産，顧客の要望，好みに合わせた少量生産製品の短期開発は，高利益なものづくりとして非常に有望である。特に大消費地東京に近い千葉県では，地理的にアドバンテージが大きいと考える。

季節の移り変りには，駅前に様々なイルミネーションが行われている。アミューズメント施設等の観光地では，イルミネーションを行うことによりその価値を高めている。第63回NHK紅白歌合戦ではPerfumeが組み込みマイクロプロセッサを搭載したイルミネーション衣装を用い，試聴者に強い印象を与えていた。このように大がかりなものだけではなく，イルミネーション製品を持っている個人の家庭も少なからず存在する。

イルミネーション装置は顧客の目的に合わせた一品物，もしくは，顧客の好みに合わせて，カスタマイズ可能であるか，細かくバージョンが異なった多品種少量生産である場合が多い。

また，流行の流行り廃りに大きく影響されるため，開発の計画を立ててから顧客が製品を手に入れるまでの期日が大きな問題となる。

このように，イルミネーション装置は，高付加価値なものづくりを考えるにあたって，非常に有効な製品である。

上記に挙げたイルミネーション装置の多くは利用中スタッフが操作するか，単純な動作の繰り返しを行うのみのものとなっている。近年では，鑑賞者の状態を様々なセンサで読み取り，それらに対してインテリジェントな動作を行うものも出はじめてきている。本研究ではそのようなインテリジェントなイルミネーション装置を小型な組み込

み用マイクロプロセッサで制御することを対象とした。

一般的に組み込みプロセッサ開発はC言語で行われている。C言語は，標準ではGC(GarbageCollector)，すなわち，オブジェクトの寿命を自動的に管理し，メモリ等のコンピュータ資源を再利用できるようにする仕組みを有していない。

コンピュータプログラム上で扱われる要素，オブジェクトが必要になる時点は簡単にわかるが，不必要になる時点すなわちオブジェクトの寿命を判断することは困難である。しかし、GCがあればオブジェクトは無限のエクステントを持つものとしてプログラムを組むことが可能となり、プログラムの設計に際しオブジェクトの寿命を考慮する必要がなくなる。本研究で対象とするインテリジェントな動作を行うプログラムは一般的に複雑であり，オブジェクトの寿命を判断することが困難なことから,GC の採用はプログラムの開発速度において非常に大きな意味を持つ。

そこで本研究では，周囲の環境情報を様々なセンサで読み取り，それらに対してインテリジェントな反応を示す装置を開発するにあたり，組み込みマイクロプロセッサ開発をGC を持った高級言語で行えるようにすることを目的とした。併せて，CPU時間等の計算機資源の抽象化を行うOS機能を含んだ開発フレームワークの実装を行った。

２．過去の研究

２．１　開発フレームワーク

２．１．１　Scheme処理系

LISP言語の一方言であるScheme言語のサブセット言語を開発した。あくまでサブセットの言語ではあるが，

・GC(Garbage Collector)
・クロージャ
・末尾呼び出しの最適化

等の近代的なプログラミング言語が持つ要素を持っている。

本処理系はAtsushi Moriwakiのmini-schemeを参考にし，そのSECDRマシンと同等の動作を行うVM(Virtual Machine)をRX63N上に実装した。そのVM上で動作するバイトコード列を生成するSchemeコンパイラをHaskell言語で記述した。このコンパイラはLinux，Mac OS X，Windows等のOSを搭載したPC上で動作する。VMもPC上でも動作し，ハードウェアに依存しないデバッグはPC上で行うことができる。

２．１．２ OS機能

本開発フレームワークにはI/O等の計算機資源の抽象化，マルチタスク機能等のOS機能も含まれている。

本フレームワークのマルチタスク機能は,Unix,Mac OS X,Windows等で採用されているプリエンプティブマルチタスクではなく，Mac OS Classic等で利用されていたノンプリエンプティブマルチタスク(協調型マルチタスク) を採用した。ノンプリエンプティブマルチタスクは各タスクが協調して動き，適宜実行権をOSに返すことが要求される。実行権をOSに返還しないタスクが存在した場合，他のタスクが実行権を得ることができなくなることが，ノンプリエンプティブマルチタスクの欠点である。本フレームワークでは，VMが強制的に実行権をOSに返還するため，このような欠点が存在しなくなる。コンテキストスイッチングに必要な資源が少なくてすむというノンプリエンプティブマルチタスクの利点のみ享受できる。

２．１．３ 試作イルミネーション

上記処理系を用いて,n-queen問題のソルバの実装を行った。

５×５のLEDマトリクス上に5-queen問題の解の表示を行う。このLEDマトリクスはダイナミック点灯で制御されている。スイッチを押すことによって，5-queenの10個の解を順に表示する。

本質的には必要ではないが，マルチタスク動作の確認のため,5-queenの解が求まった後も,再度,解の計算を行う。永遠に計算を行うこととなる。

本イルミネーションの動作には以下の機能が必要となる。

・5-queenのソルバ
・LEDマトリクスのダイナミック点灯
・チャタリングの除去を行ったキー入力

以上の処理を時分割のマルチタスクで行う。

このプログラムの記述を130行で行うことができた。

３．成果物

３．１　オセロ

本年度はインテリジェントなイルミネーション装置の試作品としてオセロゲームを作製し，開発フレームワークのリファインを行った。

３．１．１　ミニマックス法

本試作品のAIの戦略として，ミニマックス法を採用した。ミニマックス法は相手にとって有利な局面とは自分にとって不利な局面であることを利用したアルゴリズムである。相手は最良の選択を行うとして，その最良手が最も自分の有利になるものとなるような(相手から見れば不利になるように）手を選択する手法である。

オセロの盤面の有利，不利を考える指標として静的評価値を考える。静的評価値は，盤面の状態をXとすると，ある関数f(X) で導出されるものとする。静的評価値が正しく盤面の優劣を表わすことは難しいが，静的評価値が正しいと仮定し，指し手はその静的評価値が最大となる選択をするものとする。

一つ前の指し手を考える。何らかの手を指すと，指した後の盤面から，次の指し手が得る最も高い静的評価値が得られる。相手にとって評価が最小であるものが自分にとって評価が最大であるため，自分にとって最も有利な選択は，その選択をしたことによって，相手の最大の評価値が最小になるものである。この値を評価値とする。その前の指し手はその評価値が最大になるように手を選択する。これをn回繰り返す。このようにn手先の盤面評価値がお互いに，最大になるように，最小になるように交互に手を選択していくことをシミュレートし，現在の最良手を考える手法がミニマックス法である。

本試作オセロイルミネーションでは静的評価関数として，自分の駒が

・4 角にある場合16 ポイント
・4 角の隣接したマスにあれば-8 ポイント
・上記以外の盤面の端にあれば4 ポイント
・端のマスに隣接しているマスにあれば-2ポイント
・上記以外であれば1 ポイント

加算され，相手の駒があれば上記の条件に-1 をかけたものが加算される単純なものを採用した（図１）。

３．１．２　試作オセロイルミネーション

試作したオセロイルミネーションの写真を図２に、また，概略図を図３に示す。8×8のフルカラーLEDはＹ軸を指定したダイナミック点灯で表示される。各列はRed，Green，Blueそれぞれシリアル通信で送られたものが，シフトレジスタによりパラレル信号化され，表示される。

キーはそれぞれタクトスイッチと抵抗のみによって構成された最小の構成のもので，チャタリングが発生する。

このオセロイルミネーション装置を動作させるためには

・オセロを指す人工知能
・チャタリングの除去を行うキー入力
・盤面イメージからシリアルデータへの変換
・LED マトリクスのダイナミック点灯制御

が必要となる。また，試作基板にはバグがあり，そのハードウェアのバグをソフトウェアで吸収した。これらの記述が437行で記述できた。作製にかかった時間は3日，実質8時間程度であった。

	4	4	4	-8	16
	-2	-2	-2	-8	-8
	1	1	1	-2	4
	1	1	1	-2	4

図１　静的評価関数概略図

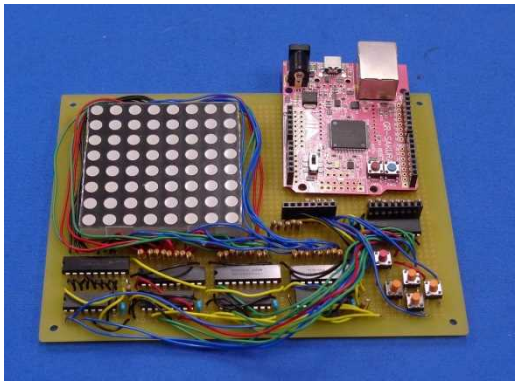

図２　オセロイルミネーション

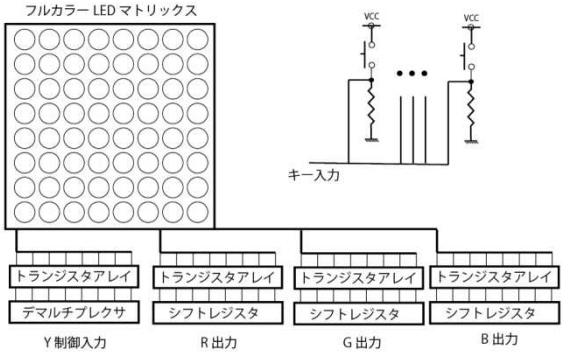

図３ オセロイルミネーション概略図

４．考察

本研究で開発したフレームワークはオセロの解法というそれなりに複雑なプログラムを非常に短い期間でコンパクトに記述することができた。その理由を考察する。

４．１　言語の多機能性

ミニマックス法は綺麗な木構造の再帰処理になり，スタック構造と相性がよく複雑なデータ構造を必要としない。しかし，それでも返せる駒の方向の数等，プログラミング中では予測し難いものが多く,最終的にプログラムが完成してはじめてGCが必要なかったと判明する。最終的にGCが必要であるかどうかの判断をプログラムの設計時ではなく本当に必要になる時点まで保留できるアドバンテージは大きい。

今回のような単純なミニマックス法では必要なかったが、クロージャを用いることによって動的に手続きを生成することができ、設計時よりも思考ルーチンが複雑になったとしても大きな設計変更なしに対応できると考えられる。

４．２　REPLの存在

REPL(Read-Eval-Print Loop)は、ユーザのキー入力を読み(Read)，それをプログラムとして評価し(Eval),結果の表示(Print)を行うことを繰り返す(Loop) 機構である。

アルゴリズムの検証，プログラムの正統性の評価等，プログラムの一部のみ動作させたい時が多々ある。

REPLを持たない言語の場合，プログラムの一部のみユーザの入力を与え，他の動作を行わないようプログラムを修正する必要がある。また，ユーザの入力をプログラムが解釈できるよう変換するプログラムも記述する必要がある。

REPLが存在することにより，関数等プログラム要素に与えるデータをad hocに作製し最適なプログラム検証作業を行うことができる。

本研究ではSchemeのサブセットの処理系であることで，他のREPLを持つScheme処理系(今回はGaucheを使った)を利用することにより，開発の効率を高めることができた。

Scheme言語の持つREPLは所謂LISP的開発を行うよう設計されたものである。Unix的なシェルからプログラムを起動するモデルとも異なるし，組み込み開発とは更に離れていると考えられる。組み込み開発に最適なREPL的なものの考案を今後の目標としたい。

５．まとめ

組み込みマイクロプロセッサを用い，周囲の環境の情報を様々なセンサで読み取りインテリジェントな反応を示すイルミネーション装置を開発するために，GCを搭載した高級言語処理系，マルチタスク等のOS機能を持つ組み込みマイクロプロセッサ用開発フレームワークを作製した。

そのフレームワーク上で人工知能(オセロのソルバ) を動作させた。

非常にコンパクトなプログラムを短期間で作製することができた。これはScheme言語の高機能性によるもののみならず，REPLの存在が大きいと考える。

今後は、従来のREPLから更に組み込み開発に特化した REPL的開発環境の考案を行いたい。