116 進化計算を用いたVHDLの自動構成と空調制御器設計への適用

Autmatically Generated VHDL Using Evolutionary Computation and its Application to Air-conditioning

○正 小島一恭（埼玉大院） 正 綿貫啓一（埼玉大院）

Kazuyuki KOJIMA, Saitama University, 255 Shimo-okubo, Sakura-ku, Saitama-shi, Saitama 338-8570
Keiichi WATANUKI, Saitama University

Recently, it has been needed to control a car air conditioner not only by merely heating and cooling based on the set temperature fixed by a passenger but also by considering of passenger's comfort. In order to design such a controller for the air conditioner, a cabin's thermal comfort in various situations has to be considered. Also, many parameters of all components such as sensors and/or actuators formed as a control system of the air conditioner have to be taken into consideration. It is for the reason that it has to take a lot of man-hours to design a program for the controller of the air conditioner. In this paper, designing support for a controller of car air conditioner using evolutionary hardware is described. First, a method to estimate the car air conditioner by using neural network is shown. Second, genetic algorithm is applied to generate a circuit of controller for the air conditioner. Next, an experiment is conducted to validate the automatically generated controller. As a result, a validation of our method is confirmed.

Key Words : Automatic control, Air conditioning system, Genetic algorithm, CPLD, VHDL

1. はじめに

近年の電子技術の著しい発展に伴い，身の回りのありとあらゆる制御システムに高機能・高性能な電子部品が組み込まれ，より複雑，高度な制御が行われるようになってきている．このようなシステムの多くは，センサから得られたアナログ信号をデジタル信号に変換し，それをMPU（Micro Processing Unit）やDSP（Digital Signal Processor），ASIC（Application Specific Integrated Circuit），CPLD（Complex Programmable Logic Device），FPGA（Field Programmable Gate Array）などを用いて演算処理し，制御量や制御シーケンスを決定している．演算処理後のデジタル信号は，アナログ信号に変換するなどして，駆動回路に入力し，アクチュエータを駆動する．このため，近年の制御器設計では，電気回路，電子回路などのハードウェアを設計するだけでなく，MPUで動作するソフトウェアの開発や専用ICの論理回路を設計することも要求される．

制御システムの制御器を設計する際，設計者はシステムを構成するセンサやアクチュエータ，演算器などの資源を機械的・電気的かつ有機的に結合し機能を発現させる．これは，制御システムの操作対象が直接的であったり，制御系が1入力1出力である場合には比較的容易に構成できるが，多入出力系では困難になる．

本論文では，このような多入出力制御システムの制御器設計支援のため，制御器の演算処理部にCPLD用い，その論理回路を記述するVHDL（VHSIC Hardware Description Language）を遺伝的アルゴリズムを用いて最適化することで，合目的的に機能発現を行うための枠組みを示す．制御システムの一例として，オートエアコンを取り上げ，本手法の適用の枠組みを示すとともに，オートエアコンを制御対象とした進化シミュレーションを行い，本手法の有用性について検討する．

2. 遺伝的アルゴリズム

図1に遺伝的アルゴリズム[1), 2)]の枠組みを示す．最適化問題の決定変数ベクトル $\boldsymbol{x}$ を，N 個の記号 s_j（j=1,…, N）の列で次のように表す．

$$\boldsymbol{x} : s = s_1 s_2 s_3 \ldots s_N \tag{1}$$

この記号列 s を N 個の遺伝子座からなる染色体とみなす．s_j は第 j 遺伝子座における遺伝子であり，s_j の取り得る値が対立遺伝子である．対立遺伝子としては，ある整数の組，ある範囲の実数値，単なる記号などを考えればよい．それぞれが式(1)で表される K 個の個体からなる集合を考え，世代 n における個体集合 $P(n)$ が遺伝子の複製・変異を経て次の世代 n+1 における個体集合 $P(n+1)$ に変わるものとする．このような世代の更新が繰り返され，更新のたびに，より最適値に近い解 $\boldsymbol{x}$ を表わす個体が選択されて，増殖するようにすれば，やがて最適解が得られるであろうというのが遺伝的アルゴリズムの基本的な考え方である．

3. FPGA/CPLDとVHDL

CPLD，FPGAはともにプログラム可能なLSIの一種で，論理ブロックの規模が比較的小さいSRAMベースのものをFPGA，

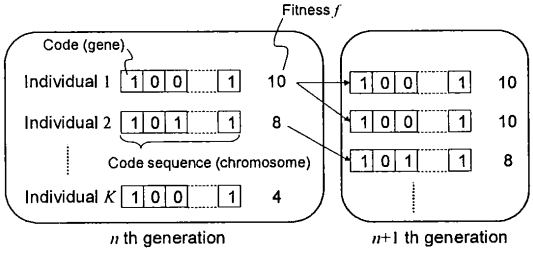


Fig.1 Framework of genetic algorithm

論理ブロックの規模が大きいEEPROMベースのものをCPLDと呼ぶ．双方ともHDL（Hardware Description Language）によって内部ロジックを設計することが可能である．これらのLSIと同様にHDLを用いて設計できるデバイスとしてASICがあるが，CPLDやFPGAなどのプログラマブル・デバイスは，設計した回路を直ちに実機評価できる，仕様変更に柔軟に対応可能であるなどのメリットがあるため，開発時の用途に適している．このため本論文では制御器としてCPLD（Xilinx XC9572）を使用するが，提案する枠組みはデバイスを限定するわけではなく，HDLで設計できるあらゆるデバイスに適用可能である．

現在，HDLとして利用されている標準的な言語にはVHDLとVerilog HDLの2種類ある．本論文ではVHDLを使用するが，2つのHDLはほぼ同等の記述能力を持ち，ここで示す枠組みは双方に対し重複して使用できる部分が多い．VHDLで記述したロジックはシミュレータや論理合成ツールなどを使用して検証および変換がなされ，デバイスに書き込み可能な形式となる．CPLDやFPGAがターゲットデバイスとなっている場合には，ダウンロードケーブルを介して，デバイスに論理回路を書き込み，簡単に目的のLSIを得ることができる．図2に簡単な論理回路を示すVHDLを示す．回路設計者はVHDLを記述することで，論理回路を設計する．図2の例では記述と回路との対応が見て取れる．

4．進化計算を用いた制御器設計支援

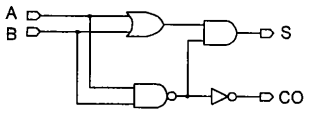
本論文ではFPGAやCPLDなどの書換え可能な論理回路ICを進化計算により適応的に自律的に構成する．ここでは，Xilinx XC9572を例に進化計算を用いた制御器設計支援の枠組みについて説明する．図3にXC9572の内部ブロックを示す[3]．XC9572は外部端子44本（ユーザI/O 34本），マクロセル72個，内部ゲート数1600個と小型のCPLDである．設計者は34本のユーザI/Oの中から選択的に入力および出力を決定し，それらのピン配置を定義する．さらに，入出力に対して信号を定義する．

```
library IEEE;
use IEEE.std_logic_1164.all;

entity HALF_ADDER is
  port(
    A,B : in std_logic;
    S,CO : out std_logic);
end HALF_ADDER;

architecture DATAFLOW of HALF_ADDER is
  signal C, D : std_logic;
begin
  C  <= A or B;
  D  <= A nand B;
  CO <= not D;
  S  <= C and D;
end DATAFLOW;
```

(a) VHDL source code



(b) Schematic

Fig.2 VHDL and its schematic

制御系の場合には，CPLDの入出力ピンに対してセンサやアクチュエータを関連付けることができる．図4に関連付けの例を示す．本例では，センサー1個，アクチュエータ2個を関連付けている．センサ値を8ビットのデジタル信号としてCPLDに入力し，2つのアクチュエータへの参照信号としてそれぞれ8ビットのデジタルデータを出力する．

5．染色体構造とVHDLの符号化

本論文では，CPLDを用いた制御器設計の一連のプロセスを自動化するために遺伝的アルゴリズムを用いてVHDLの記述を最適化する．

図5に生成されるVHDLの一例を示す．この例は図7に対応した記述となっている．VHDLは(a)エンティティ宣言部，(b)信号宣言部，(c)アーキテクチャ宣言部の3つの部分で構成される．(a)エンティティ宣言部ではCPLDの入出力信号が定義され，また，(b)信号宣言部ではCPLDの内部信号が定義される．

入出力信号，内部信号をすべて同一の **std_logic** 型にする場合，その数のみを染色体上に符号化すれば，VHDLの記述を復元できる．そこで，図6(a)に示す通り，染色体の先頭の遺伝子に入力信号数，出力信号数，内部信号数を符号化す

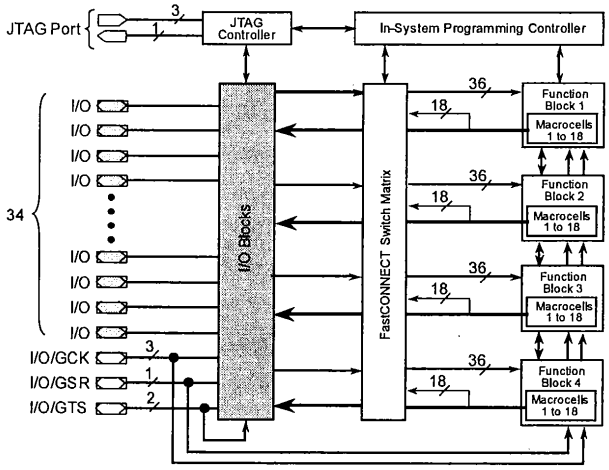


Fig.3 XC9572 architecture

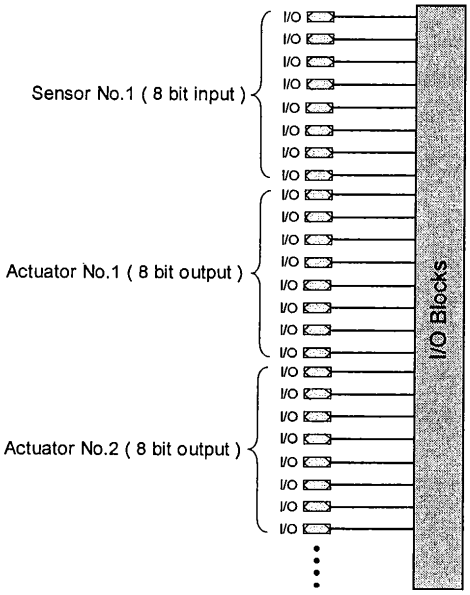


Fig.4 CPLD applications

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;

entity GA_VHDL is
    port(
        DI000 : in std_logic;
        DI001 : in std_logic;
        DI002 : in std_logic;
        DI003 : in std_logic;
        DI004 : in std_logic;
        DI005 : in std_logic;
        DI006 : in std_logic;
        DI007 : in std_logic;
        D0000 : out std_logic;
        D0001 : out std_logic;
        D0002 : out std_logic;
        D0003 : out std_logic;
        D0004 : out std_logic;
        D0005 : out std_logic;
        D0006 : out std_logic;
        D0007 : out std_logic;
        D0008 : out std_logic;
        D0009 : out std_logic;
        D0010 : out std_logic;
        D0011 : out std_logic;
        D0012 : out std_logic;
        D0013 : out std_logic;
        D0014 : out std_logic;
        D0015 : out std_logic
    );
end GA_VHDL;

architecture Behavioral of GA_VHDL is
    signal S000 : std_logic;
    signal S001 : std_logic;

begin
    S000 <= ((((not DI007 nand DI004) nor not DI005) or DI003) and not DI007) nand not DI003);
    S001 <= (((not DI007 nor not DI007) nor DI004) or not DI002) or DI002);
    process(S000, DI002) begin
        if(S000'event and S000='1')then
            D0000 <= (DI002 nand not S000);
        end if;
    end process;
    D0001 <= not DI000;

    process(S001) begin
        D0002 <= S001;
    end process;

    D0003 <= ((((not DI006 and not DI002) nand not DI001) nand not DI001) or not DI006) or DI005);
    D0004 <= ((((DI001 and not DI003) nand DI006) and DI002) nor not DI000);

    process(DI001, S000) begin
        D0005 <= (S000 nand DI001);
    end process;

    D0006 <= (((not DI000 and not DI002) nor DI003) and DI006);

    process(S000, DI001) begin
        if(S000'event and S000='1')then
            D0007<=DI001;
        end if;
    end process;

    D0008 <= (((((((S001 or DI000) and not DI006) or not DI001) nand not DI000) and not DI003) nor DI003) nand not DI002) or DI001);
    D0009 <= (((((not DI001 nand not DI001) nor DI006) nand DI001) nor not DI001) and S000);

    process(DI004) begin
        D0010 <= not DI004;
    end process;

    process(S001) begin
        if(S001'event and S001='0')then
            D0011<=S001;
        end if;
    end process;

    process(DI003, DI002) begin
        if(DI003='1')then
            D0012<=(DI003 nand not DI002);
        end if;
    end process;

    process(DI005) begin
        D0013 <= DI005;
    end process;

    D0014 <= (((((((not S000 nand not DI003) nand not DI006) or DI005) nand S001) nand S001) or not S001) or DI005) nand DI007);
    D0015 <= (((((((not DI001 nor not DI003) or not DI007) nor not DI003) nand not DI000) or DI005) and not DI007) nor not DI007) nand not DI001);

end Behavioral;
```

(a) Entity declaration

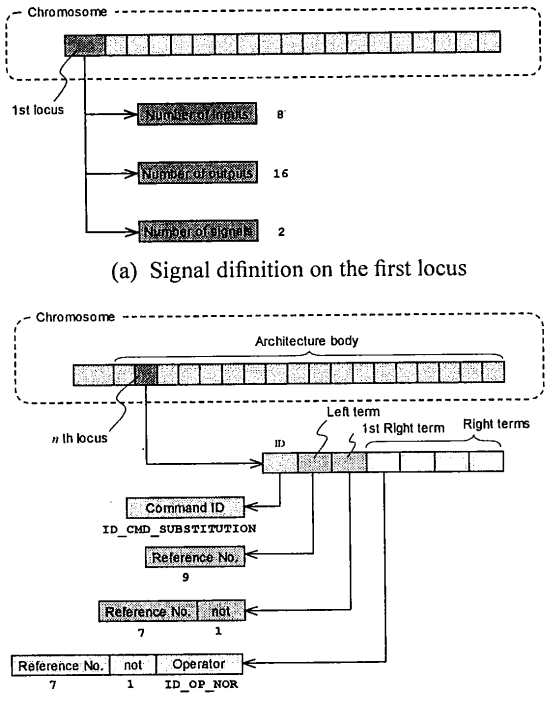
(b) Signal declaration

(c) Architecture body

(d) Substitution

(e) Process statement

Fig.5 Automatically generated VHDL



(a) Signal difinition on the first locus

(b) Substitution

Fig.6 VHDL genetic coding onto chromosome

る．図 5 (c) のアーキテクチャ宣言部では，代入文とプロセス文を使用して制御ロジックを記述している．VHDL においてアーキテクチャ宣言部に記述可能な表現は非常に多くあるが，ここでは，代入文の染色体上への符号化のみ示すこととする．図 6 (b) は代入文の符号化を図示したものである．図に記されている値は，図 5 (d) の 2 番目の代入文（左辺が「**S001**」の文）に相当している．染色体の第 2 番目以降の遺伝子はすべて同等の扱いであり，ここにアーキテクチャ宣言部の記述が符号化される．染色体は多重のリスト構造になっており，階層の数は特に制限しない．新たな VHDL の記述に対応させる場合には，必要な数だけ階層を設けることになる．2 階層目のリストの先頭は必ず ID になっており，この ID により，代入文であるのか，プロセス文であるのか，あるいは他の記述であるのかを判別することができる．ID が「**ID_CMD_SUBSTITUTION**」の場合は，代入文を表し，同時に，下層のリスト構造も図示する通りに一意に決まる．

また，本論文では遺伝的操作により染色体に矛盾が生じないように次の制約を設ける．

1) 最上位層における染色体長は，内部信号数と出力信号数に1だけ加えた長さとする．

2) すべての信号を参照番号で符号化する．

3) すべての内部信号に対し，参照番号の大きいものほど高くなるように優先度を決め，優先度の高い内部信号は優先度の低い内部信号と入力信号のみで記述する．

4) 最上位層の染色体のエンティティ宣言部には内部信号，出力信号の順で，かつ，優先度の低いものから，それぞれ一つの遺伝子上に一回だけ符号化する．

5) 交叉は最上位層の染色体のみで行い，交叉点は選択された2つの染色体のうち，短い側の染色体長を超えない位置で行い，交叉点が奇数の場合は，先頭の遺伝子を入れ替える．

6. 空調制御器設計への応用

図4の構成で，センサの入力を車室内のPMVとし，出力を2つのアクチュエータへの参照信号とみなすと，空調制御系を構成できる．本論文では，車室内空調装置を制御対象として本手法の適用を試みる．図7に空調制御システムの模式図を示す．自動車用エアコンで一般的なリヒートエアミックス方式のシステムである．ブロワの回転数に応じた量の外気が空調装置内に取り込まれ，一度，すべての空気が5℃まで冷却され，同時に除湿される．その後，一部がヒータにより80℃まで再加熱される．再加熱された暖気と冷気との混合比はエアミックスドアの開度により調整される．車室内の温度は，ブロワの回転数とエアミックスドアの開度を制御することにより行われる．制御器への入力は車室内のPMV，制御器からの出力はブロワ開度，ミックスドアの開度とし，それぞれ8ビットのデータとして取扱う．PMV(Predicted Mean Vote)[4]は，人体の定常熱収支式と温冷感や快適感に関する1000人の被験者の実験結果を基に導出された温熱指標で，PMV=0のとき温冷感が中立とされ，+3では暑い，-3では寒いと温冷感が対応づけられている．

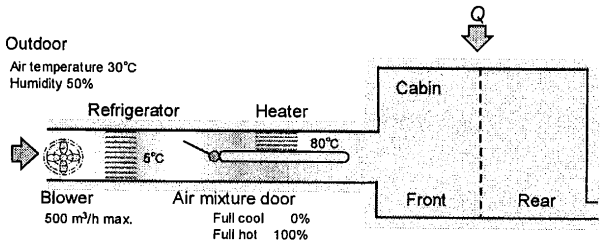
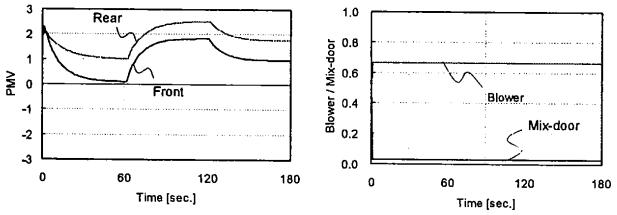
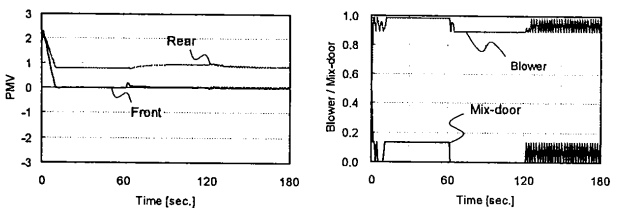


Fig.7 Air conditioning system



(a) 0 generations



(b) 10000 generations

Fig.8 Simulation results

評価関数は目標PMVを0として，180秒間の空調制御シミュレーションを行い，目標PMVとの偏差を積算し，積算値の小さいものほど適応度fitnessが高くなるようにする．ここで，tを時間[s]，ΔEを目標PMVとの偏差，T_{end}をシミュレーションの終了時間（本論文の場合は180[s]）とすると，適応度は次式で表される．

$$\text{fitness} = -\int_0^{T_{end}} \Delta E \, dt \tag{2}$$

また，シミュレーションでは，外乱として冷房負荷の変動を与える．0秒から60秒までは変動分の負荷を0[W]，60秒から120秒までは1100[W]，120秒から180秒までは550[W]を与える．

7. 進化シミュレーション

進化シミュレーションの結果を図8に示す．車室内前室のPMVを制御器にフィードバックすることとし，評価関数の計算を目標PMVと前室のPMVとの誤差で計算した場合の結果である．全てのグラフにおいて60秒ごとの負荷変動に伴って，グラフの傾向が変化していることが確認できる．図8 (a)の0世代のグラフを見るとPMVは冷房負荷の変化に伴って，上昇下降を繰り返している．冷房負荷が60秒ごとに変化しているにもかかわらず，ブロワ，ミックスドア開度の制御に変化が見られない．これは，負荷変動に伴い空調制御を行う機構がまだ形成されていないことを意味する．進化計算の世代が10000世代（図8 (b)）になると，PMVは一定値を示すようになり，目標値と前室のPMVとの誤差が減少している．また，ブロワ，ミックスドアの開度は冷房負荷の変化に応じて制御が切替わる様子が確認できる．このことから，本手法により制御機構が獲得されることが示唆される．

8. おわりに

本論文では，制御システムの制御器設計支援のため，制御器の演算処理部にCPLDを用い，その論理回路を記述するVHDLを遺伝的アルゴリズムを用いて最適化することで，合目的的に機能発現を行うための枠組みを示した．制御システムの一例として，オートエアコンを取り上げ，本手法の適用の枠組みを示すとともに，オートエアコンを制御対象とした進化シミュレーションを行い，本手法の有用性について検討した．進化シミュレーションを行い空調制御器の自動生成を試みた結果，目的に合致する形で制御機構が形成されることを確認した．このため，本枠組みは制御器設計支援のために有用であると考えられる．

参考文献

1) D.E. Goldberg, Genetic Algorithms in search, Optimization and Machine Learning, Addison-Wesley, (1989).
2) 三宮信夫・喜多一・玉置久・岩本貴司，遺伝的アルゴリズムと最適化，朝倉書店（1998），13.
3) Xilinx, XC9572 In-System Programmable CPLD Product Specification, Xilinx, (1998), 2.
4) P.O. Fanger, Thermal Comfort, McGraw-Hill, (1970).