計測自動制御学会東北支部 第 222 回研究集会(2005.6.29)
資料番号 222-13

自走 ROBOCUBE のためのアセンブリ言語について

Assembly Language for self-propelled ROBOCUBE

○熊谷龍太†，一條健司††，成田明子††，吉岡良雄††
○Ryuta Kumagai†，Kenji Ichijo††，Akiko Narita††，Yoshio Yoshioka††
†弘前大学大学院理工学研究科　††弘前大学理工学部
†Graduate Course of Science and Technology, Hirosaki University ，
††Faculty of Science and Technology, Hirosaki University
e-mail: gs05407@si.hirosaki-u.ac.jp

キーワード：　ROBOCUBE，機能分散制御，アセンブリ言語，パケット通信，共通バス

連絡先：　〒036-8561　弘前市文京町 3　弘前大学理工学部電子情報システム工学科情報メディア工学講座
吉岡良雄，Tel : 0172-39-3667，Fax : 0172-39-3667，E-mail : slyoshi@si.hirosaki-u.ac.jp

1.　はじめに

1999 年 6 月に学生実験用として非常に有用な機能分散型ロボット ROBOCUBE が(株)システムワット社によって開発された[1)]。この ROBOCUBE は，1 つのブロックに 1 つの機能のみを割り当てて，複数のブロックを積み木のように自由に組み立てることができる(図 1)。このため，ハードウェアが煩雑にならない特徴を持っている。(株)システムワット社から提供されたソフトウェア開発ツールは，リモコンおよび自走プログラムを作成できるタイル言語によるプログラム開発ツール（図式プログラム開発ツール)，およびリモコン用関数ライブラリである。前者の開発ツールは，習得に時間がかかり，習得後も他に応用ができない。そして，複雑なプログラムを作る際，煩雑になりすぎる。また，後者は自走用プログラムの作成ができない。そこで，アセンブリ言語風のプログラミング言語を開発し，そのアセンブラを作成した[2)]。

本報告では開発した ROBOCUBE のための自走アセンブリ言語仕様と実行例について述べる。

2.　ROBOCUBE とは

ROBOCUBE は次のような特徴を持っているロボットである。

- 1 つのブロックに 1 つのプロセッサを搭載し，1 つの機能のみを割り当ててある。
- ブロックを組み合わせることにより必要な機能を追加できる。
- ブロックを接続するネットワークは共通バス接続方式を採用している。
- ブロック間，およびブロックとパソコン(ホストコンピュータ)間の情報交換はパケットを使用している。

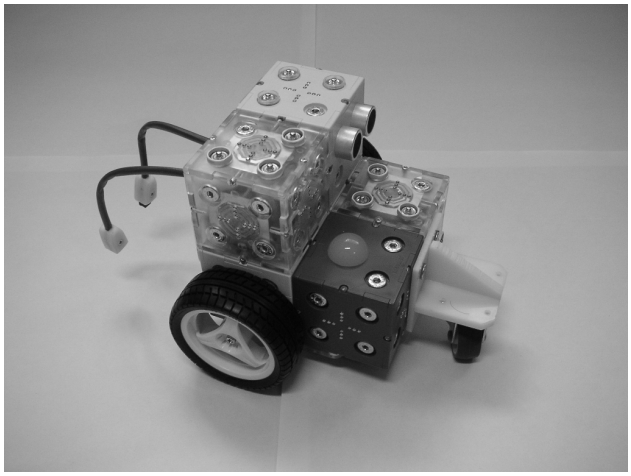

図 1 ROBOCUB 外見図

次に，ブロックの種類としては以下のものがある。

- 各ブロックを制御するコントロールブロック（制御ブロック)。
- アクチュエータブロック。例としては車輪ブロックやライトブロックなど。

・ センサブロック。例としてはタッチセンサブロックや光センサブロック，超音波センサブロックなど。

これらの特徴により，ハードウェアの構成が煩雑にならず，パケットの特徴も学ぶことができる。各種ブロックの働きは，次のようになっている。コントロールブロックが各ブロックからの情報を受け取り，判断，制御する。アクチュエータブロックでは命令に従って動作する。センサブロックは各種センサにより周りの情報をコントロールブロックに知らせる。また，センサブロックからの信号は，センサが感知した情報の他に，割り込み信号にもなっている。

動作方法としては，プログラムを予め転送しておき，それに従って動く自走方式と，人がリアルタイムに操作するリモコン方式とがある。本報告では前者の自走方式のために開発したアセンブリ言語仕様を取り扱う。

3. ROBOCUBE の内部構成

前述の通り ROBOCUBE とパソコン（ホストコンピュータ）の通信にはパケットが用いられており，ROBOCUBE にプログラムを転送する場合も同様である。よって付属のタイル言語ツールが転送するパケットを調べ，解析することで，どのような命令があるのか，どのようなパケット形式なのかがわかる。本報告ではその結果を元にアセンブリ言語を開発した。

ROBOCUBE の自走プログラムは，格納メモリ領域レベルで図 2 のようにサブルーティン領域とメインルーティン領域に分かれており，それぞれ 256 ステップと 255 ステップを格納することができる。

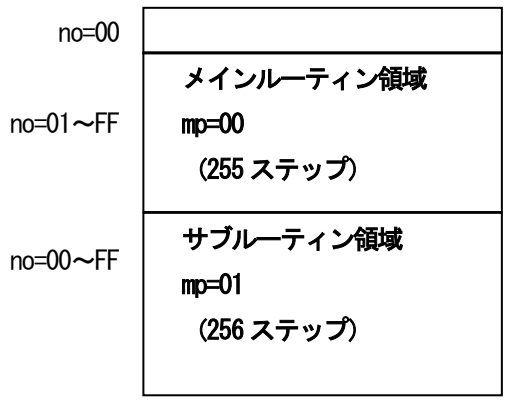

図 2 主ブロックのプログラム格納領域

*1500	mp	no	op	dt	data	it	jp	ck

*1500：5Byte の固定文字列

[mp]：2Byte サブルーティンは 01，メインルーティンは 00

[no]：2Byte アドレス番号

[op]：2Byte オペレーションコード

[dt]：2Byte パラメータ

[data]：4Byte データ

[it]：2Byte 各ブロックのノード番号

[jp]：2Byte 次のパケットのアドレス

[ck]：2Byte チェックサム

図 3 パケットの構成

そして，それぞれの領域に図 3 に示すようなパケットによって格納することができる。

4. ROBOCUBE 用自走アセンブリ言語

パケットとアセンブリ言語の関係は図 4 の様になっている。メインルーティンは主にサブルーティンを呼び出す命令で構成されている。このようなパケットを生成できるようにし，図 5 に示すようなアセンブリ言語風の記述法を開発した。

ここで，開発したアセンブリ言語の仕様を示す。

・ 行頭が「*」の行は注釈行

・ 行頭から始まる行はラベル

  ➢ ラベルはジャンプ用であるため，同じラベル名があってはならない。

・ 空白またはタブから始まる行は命令部

命令の種類としては，一般的なアセンブリ言語にある命令に加え，ブロックにデータ，状態をセットする命令や，特殊命令として SendStrobe，PacketGen，Nop，Execution，BindBlock，各ブロックの初期化等の命令がある。その他の詳細についてはホームページ[2)] を参照。

```
        mp no op dt data it jp ck *comments
    *1500 01 00 00 00 0000 00 00 C7   Subroutine
                                     forward
    *1500 01 0F 07 06 0000 00 10 EB   Motor    0,MOVE_CW
    *1500 01 10 07 07 0000 01 11 D9   Motor    1,MOVE_CCW
    *1500 01 11 07 00 0000 FF 12 FF   SendStrobe
    *1500 01 12 16 00 0000 00 00 D1   Return
          ---------------------
                                     * main program
    *1500 00 00 00 00 0000 00 00 C6   Mainroutine
    *1500 00 01 06 00 0000 00 02 CF   Nop
    *1500 00 02 00 01 0000 00 03 CC   CallSub   initialize
                                     loopback
    *1500 00 03 00 23 01FF 05 04 04   CallSub   white_A,$01ff,int
    *1500 00 04 00 0F 01FF 05 00 12   CallSub   forward,$01ff,int,00
```

図 4 ROBOCUBE への自走プログラムのパケット形式

```
  DefineBlock   10,BUZZER
  DefineBlock   10,LIGHT
  DefineBlock   0,TOUCH
  DefineBlock   0,INTERFACE
  BindBlock
  Subroutine
*
initialize
  Light    10,SET_RED,0
  Light    10,SET_GREEN,0
  Light    10,SET_BLUE,0
  Light    10,ACTION,DIM_OFF
  Light    10,ACTION,ON
  Return
*
red0
  Light    10,SET_RED,0
  Return
*
red255
  Light    10,SET_RED,255
  Return
*
blue0
  Light    10,SET_BLUE,0
  Return
*
blue255
  Light    10,SET_BLUE,255
  Return
*
touch0
  LoadAx TOUCH,0,2
  PopOneData
  TestAx   1,0
  Return
*
touch1
  LoadAx TOUCH,0,2
  PopOneData
  TestAx   5,0
  Return
*
*
  Mainroutine
  Nop
  CallSub   initialize,$01ff,int,00
* interrupt
int
  CallSubAndTest   touch0,flg01
    CallSub   red255,$01ff,int
    Jump TestTouch1
flg01
    CallSub   red0,$01ff,int
TestTouch1
  CallSubAndTest   touch1,flg02
    CallSub   blue255,$01ff,int,00
flg02
    CallSub   blue0,$01ff,int,00
*
  Execution
  end
```

図 5 ROBOCUBE 用自走アセンブリ言語プログラム例

4.1. アセンブリ言語

図 5 は，センターブロック，タッチセンサブロック，センターブロック内のライトブロックの 3 つだけを使用し，タッチセンサによって光り方が変わるプログラムである。メインルーティンは図 5 に示すように，センササブルーティンから取り込んだ状態をテストして，それぞれに応じた制御サブルーティンをコールするプログラムとなる。なお，図 5 はセンサの状態が変化したとき，割り込みが入り，その状態に応じてライトの色を変えるプログラムである。具体的には何も触らないと黒（無灯）で，タッチセンサの一方を下に倒すと青になり，もう一方を倒すと赤になる。両方を倒すと紫（ピンク）に光るプログラムである。

サブルーティンをコールする命令について説明する。記述法は以下の通りである。

CallSub label[,nodx,it,jp1]

ここで,label はジャンプするサブルーティンのアドレス（ラベル）である。以降の[]内は省略可能である。nodx にて割り込みを許可するブロックを選択し，it は割り込みが発生した場合に実行するアドレスである。jp1 はサブルーティンを終えて次に実行するアドレスとなる。jp1 に 00 を入れると，サブルーティン終了後，プログラムが一時停止し，割り込みを待つ。

4.2. マクロ命令

マクロ命令とは，複数の命令を 1 つの文字列で表現する記法である。特に，複数の命令の実行順序が決まっている場合や，わかりにくい命令が並ぶ場合にマクロ命令が利用される。例えば図 5 中の一部は図 6 のように短縮され見やすくなる。

マクロ命令未使用

```
touch0
  LoadAx TOUCH,0,2
  PopOneData
  TestAx   1,0
  Return
*
touch1
  LoadAx TOUCH,0,2
  PopOneData
  TestAx   5,0
  Return
```

マクロ命令使用

```
touch0
  TouchGetState 1,0
  Return
touch1
  TouchGetState 5,0
  Return
```

図 6 アセンブリとマクロ命令

5. まとめ

以上，本報告で示した ROBOCUBE 自走用アセンブリ言語を開発したことによって，タイル言語では煩雑になるため困難であった大きなプログラム記述が可能になったこと，ROBOCUBE を制御するためのアセンブリ言語の教育が可能になったこと，ハードとソフトの関係がわかりやすくなるなど高学年での学生実験に利用できるようになった。なお，ROBOCUBE 自走用アセンブリ言語の公開および本報告に関しては（株）システムワット社[1)] の許可を得て行っている。

参考文献

1） http://www.watt.co.jp/

2） http://earth.si.hirosaki-u.ac.jp/~slyoshi/

3） 吉岡良雄： ROBOCUBE のための自走アセンブリ言語の開発 平成 16 年 6 月 4 日版, http://earth.si.hirosaki-u.ac.jp/~slyoshi/ に掲載

4） 熊谷龍太等： ROBOCUBE 用自走アセンブラの開発，情報処理学会第 67 回全国大会 4N-8