# ソフトウェア演習5 レポート4 提出日 2016/02/01

13122029 高瀬正典

# (1)作成したプログラムの設計情報

# (1-1)全体構成

構文解析用の各関数にアセンブリ言語を出力するコードを挿入する。アセンブリ言語での出力ファイルは.csl 形式で出力する。

# (1-2)各モジュールごとの構成

FILE \*csl\_fp出力するファイルのファイルポインタint label\_numラベル番号char str\_C出力するアセンブリ言語を一時的に保存しておくバッファ

char StringStack[] アセンブリの最後に出力する DC 関連を一時的に保存しておくバッファ

#### (1-3)各関数の外部仕様

関数 int initCompiler(char \*mplname) 引数として作成するファイル名を受け取る。fopen により

そのファイルのファイルポインタを作成する。作成に成功すれば

0を返す。

関数 void CompilePrintStack(char \*str)

引数として char 配列を受け取る。str を str\_C に格納もしくは連結する。

関数 void CompileOutputCSL(char \*str)

引数として char 配列を受け取る。str を fprintf で csl\_fp に書き出す。

関数 void CompileOutputLine()

引数を取らない。str\_C に文字が格納されていれば csl\_fp

に書き出す。

関数 int CompileGetLabel()

引数を取らない。label\_num をインクリメントした数値を

返す。

関数 void CompileStack(int l, char \*str, int num\_string)

引数にラベル番号、char 配列、文字か数字かの判別用数字を取る。StringStack にアセンブリ末尾に出力する DC

関連の文字列を格納、連結する。

関数 void CompileOutputString()

引数を取らない。StringStackをcsl\_fpへ書き出す。

関数 void CompileDeclareStack(Id\_c \*\*id)

引数に id を取る。id を探索し、今回宣言された変数を アセンブリで書き出す。

関数 void CompileFormalPS(Id\_c \*\*id)

引数にidを取る。idを探索し、今回宣言された仮引数を

アセンブリで書き出す。

関数 void CompileLibrary()

引数を取らない。ライブラリとして使用できるアセンブリを 書き出す。

#### その他、以前作成した関数

pripri.c	p_program()	アセンブリを出力するコードの追加
pripri.c	block()	アセンブリを出力するコードの追加
pripri.c	SubProgram()	アセンブリを出力するコードの追加
pripri.c	CompositeSentence()	アセンブリを出力するコードの追加
pripri.c	AssignmentSentence()	アセンブリを出力するコードの追加
pripri.c	DivergencySentence()	アセンブリを出力するコードの追加
pripri.c	CallSentence()	アセンブリを出力するコードの追加
pripri.c	InputSentence()	アセンブリを出力するコードの追加
pripri.c	OutputSentence()	アセンブリを出力するコードの追加
pripri.c	OutputSpecification()	アセンブリを出力するコードの追加
pripri.c	Fomula()	アセンブリを出力するコードの追加
pripri.c	SimplicityFomula()	アセンブリを出力するコードの追加
pripri.c	Kou()	アセンブリを出力するコードの追加
pripri.c	Factor()	アセンブリを出力するコードの追加
pripri.c	Constant()	アセンブリを出力するコードの追加
pripri.c	RelationalOperator()	アセンブリを出力するコードの追加

cross.c void CompareID(Id\_c \*\*id)

末尾にすでに宣言されている変数についての、LD、LAD に関するアセンブリを出力するコードを加えた。

その他の関数については以前と同じである。

#### (2)テスト情報

(2-1)テストデータ sample11pp.mpl sample22.mpl	送信日時 sample35.mpl sample23.mpl	-	16:35 e12.mpl e24.mpl	sample21.mpl
(2-2)テスト結果 sample11pp.csl sample22. csl	sample35. csl sample23. csl	*	e12. csl e24. csl	sample21. csl

#### (2-3)テストデータの十分性

sample11pp.csl については正しい動作が確認できた。実行時のシミュレータの動作を図1に示す。 sample35.csl については正しい出力が得られなかった。配布資料と似た結果は得られたが時間が足りず実装し終えることができなかった。

その他、sample12.mpl, sample21.mpl, sample22.mpl, sample23.mpl, sample24.mpl はエラー無く動作することが確認できた。writeln 関数で表示される sample24.mpl についての動作を図2に示す。

```
Input File Name :C:¥Users¥thinkpot¥Downloads¥sample11pp.cs|
Run All Button was pressed.
Input the number of data
2
1
2
3
4
Sum of data = 10
All Simulation Result : 0
(Normal:65536, SYC:0 - 65535, Illegal:-1)
```

図1:sample11pp.csl の実行結果

```
Input File Name: C: ¥Users¥thinkpot¥Downloads¥sample24.csl
Disasm Button was pressed.
Run All Button was pressed.
{It's OK?
All Simulation Result: 0
(Normal:65536, SVC:0 - 65535, Illegal:-1)
```

図2:sample24.csl の実行結果

# (3)進捗状況

# (3-1)事前計画

開始予定日	終了予定日	見積もり時間	作業内容
1/1	1/7	10	課題達成のための方針を立てる
1/7	1/21	30	アセンブリの出力
1/21	1/28	10	シミュレータでの確認、他サンプルへの対応
1/28	2/1	10	レポートの作成

# (3-2)事前計画の立て方についての前課題からの改善点 冬休みの間に方針を立て、余裕のある予定にした。

#### (3-3)実際の進捗状況

	<u>\</u> \ \sigma \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \			
開始予定日	終了予定日	見積もり時間	作業内容	
1/1	1/14	20	課題達成のための方針を立てる	
1/14	1/31	40	アセンブリの出力	
1/31	2/1	5	シミュレータでの確認、他サンプルへの対応	
2/1	2/1	5	レポートの作成	

# (3-4)当初の事前計画と実際の進捗との差の原因

何をするのか、について理解するのに時間がかかってしまった。また、課題自体も難しく、時間がかかった。また、 提出前に期末試験や TOEIC が重なってしまい、追い込みのための時間が前日しか確保できなかった。そのため対応するサンプルが非常に少なくなってしまった。総じて、課題を甘く見積もっていた。