システムプログラミング 2
期末レポート

氏名: 重近 大智 (SHIGECHIKA, Daichi)
学生番号: 09501527

出題日: 2020 年 12 月 07 日
提出日: 2021 年 1 月 12 日
締切日: 2020 年 1 月 25 日

1 概要

本レポートでは，MIPS 言語と C 言語を用いて，提示された 5 つの課題に取り組み，その解答を報告する．実行結果は xspim 及び gcc により生成された 32bit バイナリによる結果である．

本レポートで報告するシステムプログラミング 2 の課題は次の 5 つである．

1. SPIM が提供するシステムコールを C 言語から実行できるようにしたい．A.6 節「手続き呼出し規約」に従って，各種手続きをアセンブラで記述せよ．ファイル名は，`syscalls.s` とすること．[1] また，記述した `syscalls.s` の関数を C 言語から呼び出すことで，ハノイの塔 (`hanoi.c` とする) を完成させよ．
2. `hanoi.s` を例に `spim-gcc` の引数保存に関するスタックの利用方法について，説明せよ．そのことは，規約上許されるスタックフレームの最小値 24 とどう関係しているか．このスタックフレームの最小値規約を守らないとどのような問題が生じるかについて解説せよ．
3. プログラム `report2-1.c` をコンパイルした結果をもとに，`auto` 変数と `static` 変数の違い，ポインタと配列の違いについてレポートせよ．
4. `printf` など，一部の関数は，任意の数の引数を取ることができる．これらの関数を可変引数関数と呼ぶ．MIPS の C コンパイラにおいて可変引数関数の実現方法について考察し，解説せよ．
5. `printf` のサブセットを実装し，SPIM 上でその動作を確認する応用プログラム (自由なデモプログラム) を作成せよ．フルセットにどれだけ近いか，あるいは，よく使う重要な仕様だけをうまく切り出して，実用的なサブセットを実装しているかについて評価する．ただし，浮動小数は対応しなくてもよい (SPIM 自体がうまく対応していない)．加えて，この `printf` を利用した応用プログラムの出来も評価の対象とする．

2 プログラムの説明

使用した MIPS アセンブリ言語のソースコードは，6 章に示す．

2.1 課題 2-1

まず，6.1 節に示す `syscalls.s` について説明する．

処理でスタックを確保する必要があるため，`.text` によりテキストセグメントにプログラムを配置する．続いて`.align 2` により次の命令が配置されるメモリ上のアドレスを 4 バイト境界に整列する．4 行目の`_print_int` ラベルから始まる一連の処理は，C 言語のソースコードから `print_int()` で呼び出せる処理に相当する．ま

ず subu 命令でスタックを 24 バイト確保し，sw 命令で$ra レジスタの値を$sp + 20 のメモリ上のアドレスに退避する．li 命令で$v0 レジスタに 1 を代入し，システム・コール・コード 1 である print_int を指定する．syscall により，システム・コールを行う．その後，lw 命令で$ra レジスタの値をスタックから復元し，addu 命令でスタックを開放する，j $ra により呼び出し元に処理を戻す．

他の print_string，read_int，read_string においては，システム・コール・コードがそれぞれ 4，5，8 となっていること以外は共通の処理を行っているため，ここでは触れない．

2.2 課題 2-5

続いて，6.2 節に示すデモプログラムについて説明する．

3 プログラムの使用法と実行結果

プログラムは，CentOS 7.6.1810 (Core) の xspim で動作を確認している．まず，ターミナルに xspim &と打ち込んで，xspim を実行する．実行後に load の機能を使い，拡張子が.s のアセンブリファイルを読み込む．run の機能で読み込んだプログラムを走らせる．プログラムを走らせた後，もう一度プログラムを走らせる場合には clear でメモリとレジスタの値を初期化した後，再度ロードする必要がある．syscalls.s を用いるプログラムの場合は，最後にこれを読み込ませる．

4 考察

4.1 課題 2-2

4.2 課題 2-3

4.3 課題 2-4

5 感想

6 作成したプログラムのソースコード

使用したプログラムを以下に添付する．

6.1 課題 2-1 で用いたコード

```
 1      .text
 2      .align 2
 3
 4  _print_int:
 5      subu    $sp,    $sp,    24  # スタックの積立
 6      sw      $ra,    20($sp)     # $ra レジスタの値をスタックに退避
 7
 8      li      $v0,    1           # syscall 用に print_int を指定
 9      syscall                     # システムコールの実行
10
11      lw      $ra,    20($sp)     # $ra レジスタの値の復元
12      addu    $sp,    $sp,    24  # スタックを解放
13      j       $ra                 # 呼び出し元に戻る
14
15  _print_string:
16      subu    $sp,    $sp,    24  # スタックの積立
17      sw      $ra,    20($sp)     # $ra レジスタの値をスタックに退避
18
19      li      $v0,    4           # syscall 用に print_string を指定
20      syscall                     # システムコールの実行
```

```
21
22      lw      $ra,    20($sp)     # $ra レジスタの値の復元
23      addu    $sp,    $sp,    24  # スタックを解放
24      j       $ra                 # 呼び出し元に戻る
25
26  _read_int:
27      subu    $sp,    $sp,    24  # スタックの積立
28      sw      $ra,    20($sp)     # $ra レジスタの値をスタックに退避
29
30      li      $v0,    5           # syscall 用に read_int を指定
31      syscall                     # システムコールの実行
32
33      lw      $ra,    20($sp)     # $ra レジスタの値の復元
34      addu    $sp,    $sp,    24  # スタックを解放
35      j       $ra                 # 呼び出し元に戻る
36
37  _read_string:
38      subu    $sp,    $sp,    24  # スタックの積立
39      sw      $ra,    20($sp)     # $ra レジスタの値をスタックに退避
40
41      li      $v0,    8           # syscall 用に read_string を指定
42      syscall                     # システムコールの実行
43
44      lw      $ra,    20($sp)     # $ra レジスタの値の復元
45      addu    $sp,    $sp,    24  # スタックを解放
46      j       $ra                 # 呼び出し元に戻る
```

6.2 課題 2-5 で用いたコード

xspim で実行する場合は，1 行目の`#include "spim.h"`は不要である．

```
1       #include "spim.h"
2
3       void print_char(char c)
4       {
5           char s[2]; // バッファ（2 文字目は終端文字）
6
7           s[0] = c;    // 1 文字目代入
8           s[1] = '\0'; // 終端文字代入
9
10          print_string(s); // 文字列表示
11      }
12
13      void print_big_str(char *s)
14      {
15          int i = 0; // オフセット指定用
16          char c;    // 1 文字バッファ
17
18          for (i = 0; *(s + i * sizeof(char)) != '\0'; i++)
19          {
20              c = *(s + i * sizeof(char)); // 次の文字情報を c に代入
21              if (c >= 97 && c <= 122)     // 小文字なら大文字へ
22                  c -= 32;
23              print_char(c); // 1 文字表示
24          }
25      }
26
27      void print_small_str(char *s)
28      {
29          int i = 0; // オフセット指定用
30          char c;    // 1 文字バッファ
31
32          for (i = 0; *(s + i * sizeof(char)) != '\0'; i++)
33          {
34              c = *(s + i * sizeof(char)); // 次の文字情報を c に代入
35              if (c >= 65 && c <= 90)      // 大文字なら小文字へ
```

```
36                  c += 32;
37              print_char(c); // 1 文字表示
38          }
39      }
40
41      char read_char() // 1 文字入力関数
42      {
43          char buf[1025]; // 入力文字数は最大 1024 文字
44          char c;         // 最初の 1 文字を格納
45
46          read_string(buf, 1025); // 入力受付
47          c = buf[0];             // 最初の 1 文字を c に代入
48          return c;
49      }
50
51      void myprintf(char *fmt, ...)
52      {
53          int i;        // 引数から受け取った int 値を代入
54          int argc = 0; // 何番目の引数か
55          char c;       // 引数から受け取った ASCII コードを代入
56          char *s;      // 引数となる文字列の先頭アドレスを格納
57
58          while (*fmt)
59          {
60              if (*fmt == '%')
61              {
62                  fmt++;  // 検索対象文字列を 1 文字右へ
63                  argc++; // 引数のカウント数を 1 増やす
64                  switch (*fmt)
65                  {
66                  case 'd': // 数値の表示
67                      i = *((int *)((char *)&fmt + argc * sizeof(void *)));
68                      print_int(i); // 数値表示
69                      break;
70                  case 's': // 文字列の表示
71                      s = *((char **)((char *)&fmt + argc * sizeof(void *)));
72                      print_string(s); // 文字列表示
73                      break;
74                  case 'c': // 1 文字表示
75                      c = *((char *)((char *)&fmt + argc * sizeof(void *)));
76                      print_char(c); // 1 文字表示
77                      break;
78                  case 'b': // すべて小文字で表示 *s は char *
79                      s = *((char **)((char *)&fmt + argc * sizeof(void *)));
80                      print_small_str(s);
81                      break;
82                  case 'B': // すべて大文字で表示 *s は char *
83                      s = *((char **)((char *)&fmt + argc * sizeof(void *)));
84                      print_big_str(s);
85                      break;
86                  }
87              }
88              else
89              {
90                  print_char(*fmt); // 1 文字表示
91              }
92              fmt++; // 検索対象文字列を 1 文字右へ
93          }
94      }
95
96      void myscanf(char *fmt, ...) //引数は 1 個まで
97      {
98          int *i;  // 引数となる変数のアドレスを格納
99          char *c; // 引数となる変数のアドレスを格納
100         char *s; // 引数となる変数のアドレスを格納
101
102         while (*fmt)
103         {
```

```
104                 if (*fmt == '%')
105                 {
106                     fmt++; // 検索対象文字列を 1 文字右へ
107                     switch (*fmt)
108                     {
109                     case 'd':                                           // 数
値の入力
110                         i = *((int **)((char *)&fmt + sizeof(void *))); // 代
入先情報
111                         *i = read_int();
112                         break;
113                     case 's':                                            // 文
字列の入力
114                         s = *((char **)((char *)&fmt + sizeof(void *))); // 代
入先情報
115                         read_string(s, 1025);
116                         break;
117                     case 'c':                                            // 1 文
字入力
118                         c = *((char **)((char *)&fmt + sizeof(void *))); // 代
入先情報
119                         *c = read_char();
120                         break;
121                     }
122                 }
123                 fmt++; // 検索対象文字列を 1 文字右へ
124             }
125         }
126
127         int main() // 整数専用の電卓
128         {
129             int out = 0;        // 計算結果
130             int in;             // 計算用の入力数値
131             char mode = 'f';    // mode 選択用
132             char flag;          // y or n フラグ用
133             int checkflag = -1; // in の入力の是非 (-1:初回時のみ)
134             char his_operand;   // 履歴を 1 回分保存
135             int his_num = 0;    // 履歴を 1 回分保存
136
137             myprintf("Starting %b...\n", "CALCULATOR");
138
139             while (1)
140             {
141                 myprintf("Please select the calc mode. (\"+\" or \"-\" or \"*\"
or \"/\" or \"0\" or \"c\" or \"h\" or \"q\")\nMode? : ", out);
142                 myscanf("%c", &mode); // mode 選択
143                 if (mode == 'q')      // q を選択した場合
144                     break;            // while ループを抜ける
145
146                 if (mode == '0') // '0' を選択した場合
147                 {
148                     myprintf("Do you want to reset calculation result? (y or N)\
n");
149                     myscanf("%c", &flag); // フラグ選択
150                     if (flag == 'y')      // y を選択した場合
151                     {
152                         myprintf("Reset calculation result.\n\n");
153                         out = 0; // 計算結果を 0 にリセット
154                     }
155                     else
156                         myprintf("Operation cancelled.\n\n");
157
158                     continue;
159                 }
160
161                 if (mode == 'c') // c を選択した場合
162                 {
163                     myprintf("Result : %d\n\n", out); // 確認用に結果を出力
```

```
164                     continue;                           // ループ先頭に戻る
165                 }
166
167                 if (mode == 'h') // h を選択した場合
168                 {
169                     if (checkflag == -1)
170                     {
171                         myprintf("Cannot use history func before calculating onc
e.\n\n");
172                         continue;
173                     }
174
175                     myprintf("Do you want to calc %c%d again? (y or N)\n", his_o
perand, his_num);
176                     myscanf("%c", &flag); // フラグ選択
177                     if (flag == 'y')      // y を選択した場合
178                     {
179                         myprintf("Calculated %c%d again.\n", his_operand, his_nu
m);
180                         mode = his_operand;
181                         in = his_num;
182                         checkflag = 1; // in に値を代入したため
183                     }
184                     else
185                     {
186                         myprintf("Operation cancelled.\n\n");
187                         continue;
188                     }
189                 }
190
191                 if (mode != '+' && mode != '-' && mode != '*' && mode != '/') //
 モードを正しく選択しなかった場合
192                 {
193                     myprintf("Please select the correct mode.\n\n");
194                     continue;
195                 }
196
197                 if (checkflag <= 0)
198                 {
199                     myprintf("Please input the number.(int type ONLY)\nNumber? :
 ");
200                     myscanf("%d", &in); // 整数の入力値受付
201                 }
202
203                 his_operand = mode; // history に入力モードを登録
204                 his_num = in;       // history に入力数値を登録
205
206                 if (mode == '+') // 加算モード
207                     out = out + in;
208                 if (mode == '-') // 減算モード
209                     out = out - in;
210                 if (mode == '*') // 乗算モード
211                     out = out * in;
212                 if (mode == '/') // 除算モード
213                     if (in != 0) // 0 除算は禁止
214                         out = out / in;
215                     else
216                         myprintf("Cannot divide by zero.\nOperarion denied.\n");
217
218                 myprintf("Result : %d\n\n", out); // 演算後に結果出力
219                 checkflag = 0;                    // in は未入力
220             }
221             myprintf("%B : %d", "final result", out); // 最終結果出力
222             myprintf("\nQuit.\n");
223             return 0;
224         }
```

参考文献

[1] David A. Patterson，John L. Hennessy，コンピュータの構成と設計　第 5 版 [下]　-ハードウエアとソフトウエア-，日経 BP 社，2014.