Минобрнауки России

Санкт-Петербургский государственный политехнический университет

Институт информационных технологий и управления

Кафедра «Информационные и управляющие системы»

КУРСОВОЙ ПРОЕКТ

Разработка учебной системы программирования

Вариант 7

Абсолютный загрузчик, эмулятор и отладчик

по дисциплине «Системы программирования»

Выполнили

студенты гр.5084/12 А.А.Лукашин

К.С.Шубин

Руководитель

доцент В.Я.Расторгуев

«__» _____ 2013 г.

Санкт-Петербург

2013

Оглавление

Задание

Воспользовавшись результатами второго этапа курсовой работы, доработать существующий загрузчик объектного представления программы. Произвести пошаговое выполнение программы, отслеживая состояние регистров и памяти. Это действие необходимо, чтобы убедиться в правильности второго этапа.

Вариант №7:

Код на языке PL1

```
EX07: PROC OPTIONS (MAIN);

            DCL A BIT (3) INIT ( 10B );

            DCL B BIT (3) INIT ( 101B );

            DCL C BIT (16);

            C = SUBSTR((B !! A),2,3);

END EX07;
```

Необходимо доработать загрузчик, дополнив его новой функциональностью. В новую функциональность входит поддержка новых команд (LH, STH, SRL, SLL, OR, NR). Данные команды необходимо внести в таблицу машинных операций, а так же написать соответствующие обработчики.

Модификация кода загрузчика

В рамках курсовой работы были проведены модификации и дополнения, которые позволяют решать поставленную в условии задачу.

Таблица машинных операций

```
T_MOP [NOP]  =                      {    {
{{'B','A','L','R',' '} , '\x05' , 2 , FRR} ,
{{'B','C','R',' ',' '} , '\x07' , 2 , FRR} ,
{{'S','T',' ',' ',' '} , '\x50' , 4 , FRX} ,
{{'L',' ',' ',' ',' '} , '\x58' , 4 , FRX} ,
{{'A',' ',' ',' ',' '} , '\x5A' , 4 , FRX} ,
{{'S',' ',' ',' ',' '} , '\x5B' , 4 , FRX} ,
{{'L','H',' ',' ',' '} , '\x48' , 4 , FRX} ,
{{'S','R','L',' ',' '} , '\x01' , 4 , FRX} ,
{{'S','L','L',' ',' '} , '\x02' , 4 , FRX} ,
{{'O','R',' ',' ',' '} , '\x16' , 2 , FRR} ,
{{'N','R',' ',' ',' '} , '\x14' , 2 , FRR} ,
{{'S','T','H',' ',' '} , '\x40' , 4 , FRX} ,
};
```

Серым цветом выделены строки, соответствующие добавленным операциям.

Функции обработки операций

1) Программные обработчики введенных команд

```
switch (T_MOP[k].CODOP{
   case '\x05' : P_BALR break
   case '\x07' : { i = P_BCR();
             getch();
             if (i == 1)
              return 8;
           }
            break;
    case '\x50' : P_ST();
            break;
    case '\x58' : P_L();
            break;
    case '\x5A' : P_A();
            break;
    case '\x5B' : P_S();
            break;
    case '\x48' : P_LH();
            break;
    case '\x01' : P_SRL();
            break;
    case '\x02' : P_SLL();
            break;
    case '\x16' : P_OR();
            break;
    case '\x14' : P_NR();
            break;
    case '\x40' : P_STH();
   }
```

2) Функция обработки операции типа RX

```
int FRX(void)
{
  int i, j;

  for (i = 0; i < NOP; i++)
  {
    if (INST[0] == T_MOP[i].CODOP)
    {
      waddstr(wgreen, "  ");
      for (j = 0; j < 5; j++)
        waddch(wgreen, T_MOP[i].MNCOP[j]);
      waddstr(wgreen, " ");

      j = INST[1] >> 4;
      R1 = j;
      wprintw(wgreen, "%.1d, ", j);

      j = INST[2] % 16;
      j = j * 256 + INST[3];
      D = j;
      wprintw(wgreen, "X'%.3X'(", j);
```

```
        j = INST[1] % 16;
        X = j;
        wprintw(wgreen, "%1d, ", j);

        j = INST[2] >> 4;
        B = j;
        wprintw(wgreen, "%1d)", j);

        ADDR = VR[B] + VR[X] + D;
        wprintw(wgreen,"        %.06lX       \n", ADDR);
        break;
      }
    }
```

3) Функция обработки операции SLL

```
int P_SLL()
 {
   int sm = D;
   VR[R1] = VR[R1] << sm;
   return 0;
 }
```

4) Функция обработки операции SRL

```
int P_SRL()
 {
   int sm = D;
   VR[R1] = VR[R1] >> sm;
   return 0;
 }
```

5) Функция обработки операции OR

```
int P_OR()
 {
   VR[R1] = VR[R1] | VR[R2];
   return 0;
 }
```

6) Функция обработки операции NR

```
int P_NR()
 {
  VR[R1] = VR[R1] & VR[R2];
  return 0;
 }
```

7) Функция обработки операции LH

```
int P_LH()
{
   int sm;
   ADDR = VR[B] + VR[X] + D;
   sm = (int)(ADDR - I);
   VR[R1] = OBLZ[BAS_IND + CUR_IND + sm] * 0x100L + OBLZ[BAS_IND +
CUR_IND + sm + 1];
   return 0;
}
```

8) Функция обработки операции STH

```
int P_STH()
 {
  int sm,i;
  char bytes[2];
```

```
 ADDR = VR[B] + VR[X] + D;
 sm = (int) (ADDR -I);
 bytes[0] = ((VR[R1] % 0x10000L) - ((VR[R1]%0x10000L)%0x100))/0x100;
 bytes[1] = (VR[R1] % 0x10000L) % 0x100;
 for (i=0; i<2; i++)
   OBLZ[BAS_IND + CUR_IND + sm + i] = bytes[i];
 return 0;
}
```

Выводы

В рамках проведенной работы был модифицирован абсолютный загрузчик и отладчик. Выполненные изменения позволили провести выполнение сформированного во втором этапе объектного представления:

1) Расширена таблица машинных команд
2) Модифицированы обработчики новых машинных команд
3) Осуществлена проверка работы модуля

В рамках отладочного запуска скомпилированной объектной карты была произведена проверка правильности результатов второго этапа. Результат соответствует требованиям задания.