Домашняя работа №4

ISA

Цель работы: знакомство со системой набора команд RISC-V.

Инструментарий и требования к работе: работа может быть выполнена на любом из следующих языков: C, C++, Python, Java.

Порядок выполнения работы:

1. Изложить в письменной форме:
    a. описание системы кодирования команд RISC-V;
    b. структуру elf-файла;
2. Написать программу, которая будет находить и дизассемблировать секцию кода (.text).

Содержание отчета

1. Теоретическая часть (пункт 1 из Порядка выполнения);
2. Описание работы написанного кода (пункт 2 из Порядка выполнения, экспериментальная часть);
3. Результат работы написанной программы на приложенном к заданию файле (также в экспериментальной части, для лучшей видимости выделить подпись жирным);
4. Листинг кода с указанием компилятора/интерпретатора (подробнее Оформление кода в отчёте).

Примечания:

1. Ссылка на файл (пункт 3 Содержания отчета): https://vk.cc/bH7DIA;
2. Файл с отчётом подгружаем в саму форму: https://vk.cc/bH7NdO;
3. В форму также необходимо загрузить результат работы программы в виде файла решения, в котором будет содержаться вывод программы (подробнее);

4. В поле «Ссылка на отчет» требуется ссылка на ответ, а не на диск, где лежит отчет. Данное поле заполняется, если по каким-либо причинам не удалось приложить файл в форму. Также можно подстраховаться на случай, если файл приложится некорректно, заполнив это поле. Если ссылка на отчет приложена неверно и в форме нет приложенного файла, то отчет не принимается;
5. В поле «Ссылка на код» можно вставлять как ссылку на файл (исходного кода либо архив с исходниками), так и на диск (где именно вы будете хранить файлы не принципиально, главное – открытый доступ по ссылке до обозначенного времени) или репозиторий (git – репозиторий должен быть закрытым и расшаренным с мной (RonoveRaum));
6. «Шаблон отчета»: https://vk.cc/aAWqZm;
7. В отчете нужно оставлять комментарии, почему вы выбираете ту или иную схему для построения и прочие комментарии. Поскольку очных защит работ у нас нет, то, по сути, отчеты и есть ваша защита выполненной домашней работы. Поэтому чем больше пояснений и комментариев (уместных конечно же) вы оставляете в работе, тем в ходе проверки будет проще понять ход ваших мыслей, что а) упросит проверку и б) в случае неоднозначных трактовок ТЗ позволит сохранить за вами баллы, если схемы по итогу будут работать.

Дополнительные сведения (код)

1. Аргументы программе передаются через командную строку:
   `hw4.exe <имя_входного_elf_файла> [<имя_выходного_файла>]`
   Если указано последнее, то результат работы логируется в этот текстовый файл;
2. Корректно выделяется и освобождается память, закрываются файлы, есть обработка ошибок: не удалось открыть файл, формат файла не поддерживается.

Если программе передано значение, которое не поддерживается – следует сообщить об ошибке;

3. В программе можно вызывать только стандартные библиотеки (например, <bits/stdc++.h> таковой не является и ее использование влечет за собой потерю баллов);
4. Если программа использует библиотеки, которые явно не указаны в файле с исходным кодом (например, <algorithm>), то за это также будут снижаться баллы;
5. Если во входном файле встречается команда, которая не распознается программой, то следует выводить unknown command.

Дополнительные сведения (дизассемблер)

ISA: **RISC-V RV32I, RV32M**.

Для каждой строки кода указывается её адрес в hex формате (16 СС).

Обозначение меток можно найти в Symbol Table (.symtable). Если же название метки там не найдено, то используется следующее обозначение: LOC_%08x, например, LOC_00000000, LOC_00000034.

Для каждой метки перед названием указывается адрес (пример ниже).

Комментарии идут в конце строки через ‘#’.

Пример дизассемблера (symbol table для этого кода приведено ниже):

```
00010078: <_start> addi      a0, zero, 0
0001007a:          lui       a1, 65536
0001007c:          addi      a1, a1, 158      # 0x0001009e <msg>
00010080:          addi      a2, zero, 12
00010082:          addi      a3, zero, 0
00010084:          addi      a7, zero, 64
00010088:          ecall
0001008c:          addi      a0, zero, 0
```

```
0001008e:          addi       a1, zero, 0
00010090:          addi       a2, zero, 0
00010092:          addi       a3, zero, 0
00010094:          addi       a7, zero, 93
00010098:          ecall
0001009c: <loop> jal    zero, pc + 0      # 0x0001009c <loop>
```

Symbol Table (.symtab)

```
Symbol Value              Size  Type     Bind     Vis       Index Name
[   0] 0x0                   0 NOTYPE   LOCAL    DEFAULT   UNDEF
[   1] 0x100b0               0 SECTION  LOCAL    DEFAULT       1
[   2] 0x10158               0 SECTION  LOCAL    DEFAULT       2
[   3] 0x11168               0 SECTION  LOCAL    DEFAULT       3
[   4] 0x0                   0 SECTION  LOCAL    DEFAULT       4
[   5] 0x0                   0 SECTION  LOCAL    DEFAULT       5
[   6] 0x0                   0 FILE     LOCAL    DEFAULT     ABS test.c
[   7] 0x11168               4 OBJECT   LOCAL    DEFAULT       3 counter.0
[   8] 0x11967               0 NOTYPE   GLOBAL   DEFAULT     ABS
__global_pointer$
[   9] 0x11167               0 NOTYPE   GLOBAL   DEFAULT       2 __SDATA_BEGIN__
[  10] 0x100b0              40 FUNC     GLOBAL   DEFAULT       1 _puts
[  11] 0x10126              50 FUNC     GLOBAL   DEFAULT       1 _start
[  12] 0x11170               0 NOTYPE   GLOBAL   DEFAULT       3 __BSS_END__
[  13] 0x11167               0 NOTYPE   GLOBAL   DEFAULT       3 __bss_start
[  14] 0x100d8              78 FUNC     GLOBAL   DEFAULT       1 _isr
[  15] 0x11167               0 NOTYPE   GLOBAL   DEFAULT       2 __DATA_BEGIN__
[  16] 0x11167               0 NOTYPE   GLOBAL   DEFAULT       2 _edata
[  17] 0x11170               0 NOTYPE   GLOBAL   DEFAULT       3 _end
[  18] 0x1009c               0 NOTYPE   GLOBAL   DEFAULT       1 loop
```

Оформление кода в отчёте

1. Никаких скринов кода – код в отчет добавляется только текстом;
2. Шрифт: `Consolas` (размер 10-14 на ваше усмотрение);

3. Выравнивание по левому краю;
4. Подсветка кода допустима. Текст должен быть читаемым (а не светло-серый текст, который без выделения на белом не разобрать);
5. В раздел Листинг код вставляется полностью в следующем виде:

   <Название файла>

   <Его содержимое>

   Файлы исходных кодов разделяются новой строкой.

   Например,

   main.cpp

```
int main()
{
  return 0;
}
```

   tmain.cpp

```
int tmain()
{
  return 666;
}
```

6. Фон белый (актуально для тех, у кого копипаста кода идет вместе с фоном темной темы из IDE).

Оформление дизассемблера в отчёте

1. Результат работы программы оформляется `Consolas` (размер 10-14 на ваше усмотрение);
2. Интервал: 1.0;
3. Выравнивание по левому краю;
4. Подпись кода (в теории) – слева без нумерации с двоеточием в конце: Пример дизассемблера (symbol table для этого кода приведено ниже):
5. Для строк кода используется следующее форматирование (приведено оформление в стиле Си): `"%08x: <%s>\t%s\t%s, %s, %s\n"` (с меткой); `"%08x: \t%s\t%s, %s, %s\n"` (без метки);

6. Регистр команд: нижний (строчные буквы);
7. Операнды отделяются друг от друга через `", ";`
8. Комментарий ставится на той же строке, что и код и отделяется от него `"\t".`