Архитектура вычислительных систем

Семинары № 6

Директивы. Макросы.
Многофайловые программы

План семинарского занятия

Цель и задачи

более глубокое погружение в разработку программа на ассемблере RARS
для повышения эффективности выполнения домашних и индивидуальных заданий.

Основные вопросы

1. Дополнительные директивы, повышающие эффективность написания кода.
2. Использование макросов.
3. Создание макробиблиотек.
4. Сочетание макросов и подпрограмм.
5. Создание многомодульных программ.

Дополнительные директивы

Повышающие эффективность написания кода

Директивы управления

Для более эффективного управления памятью при написании программ.

Обозначение	Соглашения по использованию
.align	Align next data item on specified byte boundary (0=byte, 1=half, 2=word, 3=double)
.ascii	Store the string in the Data segment but do not add null terminator
.asciz	Store the string in the Data segment and add null terminator
.byte	Store the listed value(s) as 8 bit bytes
.data	Subsequent items stored in Data segment at next available address
.double	Store the listed value(s) as double precision floating point
.dword	Store the listed value(s) as 64 bit double-word on word boundary
s1	Сохраняемый регистр 1 (saved register 1)
.eqv	Substitute second operand for first. First operand is symbol, second operand is expression (like #define)
.float	Store the listed value(s) as single precision floating point
.half	Store the listed value(s) as 16 bit halfwords on halfword boundary
.section	Allows specifying sections without .text or .data directives. Included for gcc comparability
.space	Reserve the next specified number of bytes in Data segment
.string	Alias for .asciz
.text	Subsequent items (instructions) stored in Text segment at next available address
.word	Store the listed value(s) as 32 bit words on word boundary

Псевдонимы (алиасы)

Псевдонимы обычно предназначены для подмены одного текста другим. В RARS для этого используется примитивный макрос, являющийся директивой .eqv.

.eqv *имя_псевдонима строка_заменяющая имя*

В результате препроцессорной обработки:

Макроподстановка

Механизм поиска шаблона в тексте и замены его другим текстом.

Полученный текст также может содержать шаблоны, так что процесс макроподстановки обычно рекурсивен.

Обозначение	Соглашения по использованию
`.end_macro`	End macro definition. See `.macro`
`.macro`	Begin macro definition. See `.end_macro`

Многофайловые программы

В реальных системах программирования программы собираются из множества модулей, которые хранятся в отдельных файлах, образуя проект.

Директивы для создания многофайловых проектов

Обозначение	Соглашения по использованию
`.extern`	Declare the listed label and byte length to be a global data field
`.global`	Declare the listed label(s) as globl to enable referencing from other files
`.globl`	Declare the listed label(s) as global to enable referencing from other files|
`.include`	Insert the contents of the specified file. Put filename in quotes.

Использование макросов

Особенности реализации в RISC-V

Пример алгоритма Евклида

```
# Ввод целого числа в заданный регистр
.macro read_int(%x)
   li a7, 5
   ecall
   mv %x, a0
.end_macro

# Печать содержимого регистра как целого
.macro print_int (%x)
    li a7, 1
    mv a0, %x
    ecall
.end_macro
```

```
# int euclid(a, b) {
#    while (a != b)
#        if (a > b) a = a - b;
#        else b := b - a;
#    return a;
# }
```

```
main:
    read_int (t1)
    read_int (t2)

loop:
    beq t1, t2, finish

    slt t0, t1, t2
    bne t0, zero, if_less

    sub t1, t1, t2
    j   loop

if_less:
    sub t2, t2, t1
    j   loop

finish:
    print_int (t1)
```

Создание макробиблиотек

Следующий пример с той же самой программой, но макросы собраны в виде некоторой библиотеки.

Текст программы /euclid1.

Макросы с локальными метками

В различны ассемблерах существуют различные подходы решению проблемы дублирования имен.

- В RARS сделано все просто.

Уникальность имен обеспечивается добавлением суффикса **_Mi**.

Сочетание макросов и подпрограмм

Вычисление НОД выделяется в подпрограмму.

Рассмотрим пример euclid2x

В данном примере результат становится ошибочным, так как макросы используют те же регистры, что и подпрограмма.

Если же регистры, занимаемые макросами, должны использоваться, то можно поступать как и с подпрограммами: сохранять их на стеке.

Рассмотрим пример euclid3

Перед вызовом конфликтующих макросов осуществляется сохранение в стеке. Для удобства дополнительно разработаны два макроса push и pop.

А эта программа работает уже правильно euclid4.

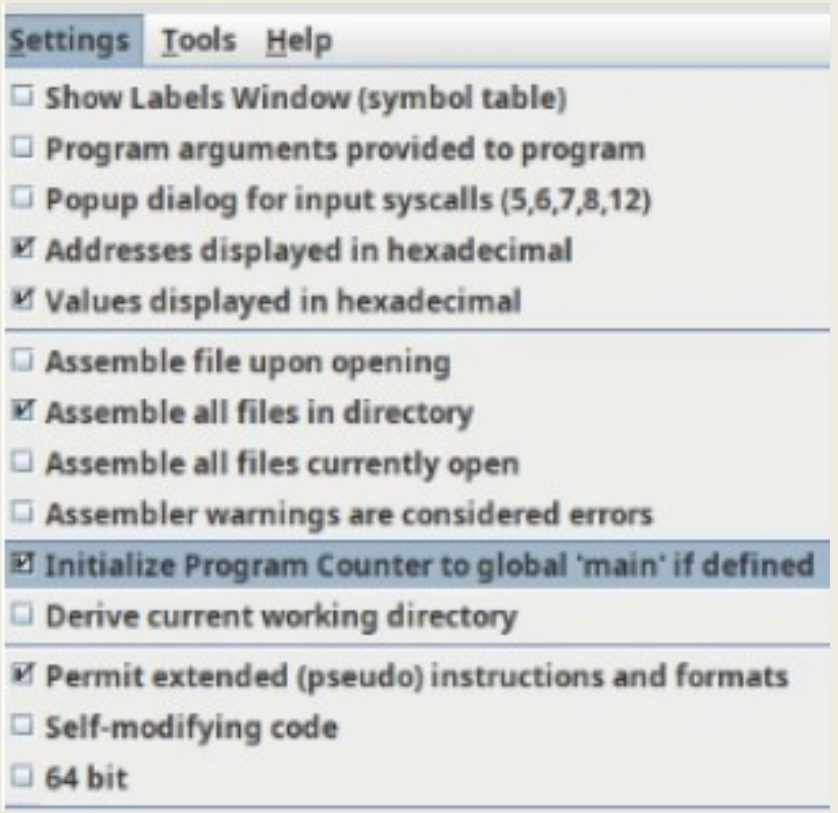
Проблемы макровзрыва и обертывание подпрограмм

Создание многомодульных программ

Многомодульные программы состоят из нескольких единиц компиляции, в каждой из которых сформирован некоторый код реализующий часть решаемой задачи.

Для того, чтобы подпрограммы могли быть видны из вне, необходимых имена отметить директивой .global (.globl)

Рассмотрим пример euclid05mod.



Установка опций **Assemble all files in directory** и **Initialize Program Counter to global 'main' if defined** для компиляции и выполнения многофайловых программ

Домашнее задание