CS 429H, Spring 2011
Y86 assembly exercises
Assigned: Fri Feb 11, Due: Friday Feb 18, 11:59PM

Christian Miller (`ckm@cs.utexas.edu`) is the lead person for this assignment.

1 Introduction

In this lab, you will transform three simple functions from C into Y86 assembly and test them against a simulator. The purpose of this is to give you practice with assembly level programming in general, and with the Y86 instruction set and tools in particular.

2 Logistics

You will work on this lab individually.

Any clarifications and revisions to the assignment will be posted on the course Web page.

3 Handout Instructions

You can get a copy of this handout and the assignment code from the CS429H labs webpage:

```
http://www.cs.utexas.edu/~fussell/courses/cs429h/labs/labs.shtml
```

Start by copying the file `asmlab-handout.tar` to a (protected) directory in which you plan to do your work. Then give the command:

```
tar xvf asmlab-handout.tar
```

In the newly-created directory, run `make` to build the distribution.

4 Assignment

Your task is to write and simulate the following three Y86 programs. The required behavior of these programs is defined by the example C functions in `examples.c`. Be sure to put your name and ID in a comment at the beginning of each program.

`sum.ys`: Iteratively sum linked list elements

Write a Y86 program (`sum.ys`) that iteratively sums the elements of a linked list. Your program should consist of a main routine that invokes a Y86 function (`sum_list`) that is functionally equivalent to the C `sum_list` function in Figure 1. Test your program using the following three-element list:

```
# Sample linked list
.align 4
ele1:
        .long 0x00a
        .long ele2
ele2:
        .long 0x0b0
        .long ele3
ele3:
        .long 0xc00
        .long 0
```

`rsum.ys`: Recursively sum linked list elements

Write a recursive version of `sum.ys` (`rsum.ys`) that recursively sums the elements of a linked list.

Your program should consist of a main routine that invokes a recursive Y86 function (`rsum_list`) that is functionally equivalent to the `rsum_list` function in Figure 1. Test your program using the same three-element list you used for testing `list.ys`.

`copy.ys`: Copy a source block to a destination block

Write a program (`copy.ys`) that copies a block of words from one part of memory to another (non-overlapping area) area of memory, computing the checksum (Xor) of all the words copied.

Your program should consist of a main routine that calls a Y86 function (`copy_block`) that is functionally equivalent to the `copy_block` function in Figure 1. Test your program using the following three-element source and destination blocks:

```
.align 4
# Source block
src:
        .long 0x00a
        .long 0x0b0
        .long 0xc00

# Destination block
dest:
        .long 0x111
        .long 0x222
        .long 0x333
```

To test your code, use the included simulator. Using `sum.ys` as an example, first enter `yas sum.ys` to produce the object file `sum.yo`. Then enter `yis sum.yo` to run the program. This will simulate your code and print out the number of steps to termination, the final register map, and a list of all the changes made to memory.

5 Evaluation

The lab is worth 35 points, 10 points for each Y86 solution program, plus 5 points for style. Each solution program will be evaluated for correctness, including proper handling of the `%ebp` stack frame register and functional equivalence with the example C functions in `examples.c`.

The programs `sum.ys` and `rsum.ys` will be considered correct if their respective `sum_list` and `rsum_list` functions return the sum `0xcba` in register `%eax`.

The program `copy.ys` will be considered correct if its `copy_block` function returns the sum `0xcba` in register `%eax`, and copies the three words `0x00a`, `0x0b`, and `0xc` to the 12 contiguous memory locations beginning at address `dest`.

Of course, your program will not be considered correct if it simply loads the right values into memory and returns.

6 Handin Instructions

To submit your code, use the following command:

```
turnin --submit ckm asmlab sum.ys rsum.ys copy.ys
```

Make sure you have included your name and UTCS ID in a comment at the top of each of your files.

```c
/* linked list element */
typedef struct ELE {
    int val;
    struct ELE *next;
} *list_ptr;

/* sum_list - Sum the elements of a linked list */
int sum_list(list_ptr ls)
{
    int val = 0;
    while (ls) {
        val += ls->val;
        ls = ls->next;
    }
    return val;
}

/* rsum_list - Recursive version of sum_list */
int rsum_list(list_ptr ls)
{
    if (!ls)
        return 0;
    else {
        int val = ls->val;
        int rest = rsum_list(ls->next);
        return val + rest;
    }
}

/* copy_block - Copy src to dest and return xor checksum of src */
int copy_block(int *src, int *dest, int len)
{
    int result = 0;
    while (len > 0) {
        int val = *src++;
        *dest++ = val;
        result ^= val;
        len--;
    }
    return result;
}
```

Figure 1: **C versions of the Y86 solution functions.** See `examples.c`