Handout for CS 372H
Class 11
23 February 2012

1. Implementing swtch()

Per−thread state in thread control block:

```
typedef struct tcb {
    unsigned long esp;      /* Stack pointer of thread */
    char *t_stack;          /* Bottom of thread’s stack */
    /* ... */
    };
```

Machine−dependent thread−switch function:

```
void swtch(tcb *current, tcb *next);
```

Machine−dependent thread initialization function:

```
void thread_init (tcb *t, void (*fn) (void *), void *arg);
```

Implementation of swtch(current, next):

```
pushl %ebp; movl %esp, %ebp             # Save frame pointer
pushl %ebx; pushl %esi; pushl %edi      # Save callee−saved regs

movl 8(%ebp),%edx                       # %edx = current
movl 12(%ebp),%eax                      # %eax = next
movl %esp,(%edx)                        # %edx−>esp = %esp
movl (%eax),%esp                        # %esp = %eax−>esp

popl %edi; popl %esi; popl %ebx         # Restore callee saved regs
popl %ebp                               # Restore frame pointer
ret                                     # Resume execution
```

2. Example of compiler output

foo.c:

```
#include <stdio.h>

int main(int argc, char** argv) {
    printf("hello world\n");
}

$ gcc −m32 −fno−builtin −S foo.c

# −m32 means 32−bit x86 code
# −S means don’t run the assembler
```

Here’s the output. note that there are two symbolic references:

foo.s:

```
                .file       "foo.c"
                .section    .rodata
        .LC0:
                .string     "hello world\n"
                .text
        .globl main
                .type       main, @function
        main:
                pushl       %ebp
                movl        %esp, %ebp
                andl        $−16, %esp
                subl        $16, %esp
                movl        $.LC0, (%esp)
                call        printf
```

```
3. Another example

    main.c:
        extern float sin();
        extern int printf(), scanf();
        float val = 0.0;

        int main(int argc, char** argv) {
            static float x = 0.0;
            printf("enter number: ");
            scanf("%f", &x);
            val = sin(x);
            printf("Sine of your value is %f\n", val);
        }

    C library:
        int scanf(char* fmt, ...) { ... }
        int printf(char* fmt, ...) { ...}

    math.c:
        float sin(float x) {
            float tmp1, tmp2;
            static float res = 0.0;
            static float lastx = 0.0;
            if (x != lastx) {
                lastx = x;

                /* compute sin */
            }
            return res;
        }

    [thanks to David Mazieres]
```