```
/*
 * Tests for prctl(PR_GET_TSC, ...) / prctl(PR_SET_TSC, ...)
 *
 * Basic test to test behaviour of PR_GET_TSC and PR_SET_TSC
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <inttypes.h>


#include <sys/prctl.h>
#include <linux/prctl.h>

/* Get/set the process' ability to use the timestamp counter instruction */
#ifndef PR_GET_TSC
#define PR_GET_TSC 25
#define PR_SET_TSC 26
# define PR_TSC_ENABLE      1   /* allow the use of the timestamp counter */
# define PR_TSC_SIGSEGV     2   /* throw a SIGSEGV instead of reading the TSC */
#endif

const char *tsc_names[] =
{
    [0] = "[not set]",
    [PR_TSC_ENABLE] = "PR_TSC_ENABLE",
    [PR_TSC_SIGSEGV] = "PR_TSC_SIGSEGV",
};

uint64_t rdtsc() {
uint32_t lo, hi;
/* We cannot use "=A", since this would use %rax on x86_64 */
__asm__ __volatile__ ("rdtsc" : "=a" (lo), "=d" (hi));
return (uint64_t)hi << 32 | lo;
}

void sigsegv_cb(int sig)
{
    int tsc_val = 0;

    printf("[ SIG_SEGV ]\n");
    printf("prctl(PR_GET_TSC, &tsc_val); ");
    fflush(stdout);

    if ( prctl(PR_GET_TSC, &tsc_val) == -1)
        perror("prctl");

    printf("tsc_val == %s\n", tsc_names[tsc_val]);
    printf("prctl(PR_SET_TSC, PR_TSC_ENABLE)\n");
    fflush(stdout);
    if ( prctl(PR_SET_TSC, PR_TSC_ENABLE) == -1)
        perror("prctl");

    printf("rdtsc() == ");
}

int main(int argc, char **argv)
{
    int tsc_val = 0;

    signal(SIGSEGV, sigsegv_cb);

    printf("rdtsc() == %llu\n", (unsigned long long)rdtsc());
    printf("prctl(PR_GET_TSC, &tsc_val); ");
    fflush(stdout);

    if ( prctl(PR_GET_TSC, &tsc_val) == -1)
        perror("prctl");

    printf("tsc_val == %s\n", tsc_names[tsc_val]);
    printf("rdtsc() == %llu\n", (unsigned long long)rdtsc());
    printf("prctl(PR_SET_TSC, PR_TSC_ENABLE)\n");
    fflush(stdout);

    if ( prctl(PR_SET_TSC, PR_TSC_ENABLE) == -1)
        perror("prctl");

    printf("rdtsc() == %llu\n", (unsigned long long)rdtsc());
    printf("prctl(PR_SET_TSC, PR_TSC_SIGSEGV)\n");
    fflush(stdout);

    if ( prctl(PR_SET_TSC, PR_TSC_SIGSEGV) == -1)
        perror("prctl");

    printf("rdtsc() == ");
    fflush(stdout);
    printf("%llu\n", (unsigned long long)rdtsc());
    fflush(stdout);

    exit(EXIT_SUCCESS);
}
```