컴퓨터학부 20152385 송민구

1. 소스 코드

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/time.h>

#include <sys/wait.h>

#include "ssu\_runtime.h"

void ssu\_echo\_exit(int status);

int main(void)

{

pid\_t pid;

int status;

gettimeofday(&begin\_t, NULL);

if ((pid = fork()) < 0) { // 자식프로세서 fork()

fprintf(stderr, "fork error\n");

exit(1);

}

else if (pid == 0) { // 자식프로세서는

exit(7); // 7번 신호로 exit()

}

if (wait(&status) != pid) { // 자식 프로세서가 종료될때까지 wait()

fprintf(stderr, "wait error\n");

exit(1);

}

ssu\_echo\_exit(status); // 함수 호출

if ((pid = fork()) < 0) { // 자식프로세서 fork()

fprintf(stderr, "fork error\n");

exit(1);

}

else if (pid == 0) { // 자식 프로세서는

abort(); // abort()로 강제 종료

}

if (wait(&status) != pid) { // 자식프로세서를 wait()

fprintf(stderr, "wait error\n");

exit(1);

}

ssu\_echo\_exit(status); // 함수 호출

if ((pid = fork()) < 0) { // 자식프로세서 fork()

fprintf(stderr, "fork error\n");

exit(1);

}

else if (pid == 0) { // 자식프로세서는

status /= 0; // 0으로 나누기 signal 발생

}

if (wait(&status) != pid) { // 자식프로세서 종료를 wait()

fprintf(stderr, "wait error\n");

exit(1);

}

ssu\_echo\_exit(status); // 함수 호출

gettimeofday(&end\_t, NULL);

ssu\_runtime(&begin\_t, &end\_t);

exit(0);

}

void ssu\_echo\_exit(int status) { // 프로그램이 어떤 signal로 종료되었는지

if (WIFEXITED(status)) { // coredump 파일은 남기는지 여부를 출력 하는 함수

printf("normal termination, exit status = %d\n",

WEXITSTATUS(status));

}

else if (WIFSIGNALED(status)) {

printf("abnormal termination, signal number = %d%s\n",

WTERMSIG(status),

#ifdef WCOREDUMP

WCOREDUMP(status) ? " (core file generated)" : "");

#else

"");

#endif

}

else if (WIFSTOPPED(status)) {

printf("child stopped, signal number = %d\n", WSTOPSIG(status));

}

}

1. 실행 결과

