Signal Assignment

- 작성자
  - 김영진 / 2016025241
  - 김정현 / 2014004448

- github: https://github.com/smilu97/system-hyu

#1 시그널 핸들러 바꿔보기

- Related file: src/stop.c => stop

목적

Ctrl-C가 눌려도 프로세스가 종료되지 않도록 한다.

sigaction

Signal이 Process에 전송되면, 비동기적으로 해당 Signal과 매칭 되는 Signal handler이 호출되게 된다.

이 signal handler는 메모리 영역 어딘가에서 해당되는 signal의 번호와 같이 매칭되어 저장되어있는 것으로 보이는데, `sigaction` 함수를 이용해서 이 테이블을 변경시킬 수 있다.

Arguments

- int signum
  - 바꿀 시그널의 번호
- const struct sigaction * act
  - 바꿀 시그널 핸들러를 담고 있는 `struct sigaction` 객체의 주소값
- struct sigaction * oldact
  - 기존에 존재하던 시그널 핸들러에 대한 정보를 기록할 `struct sigaction` 객체의 주소값
  - NULL일 경우 기록되지 않는다.

Example

```c
// Prepare struct sigaction
struct sigaction handler_sig_fn;
memset(&handler_sig_fn, 0x00, sizeof(struct sigaction));
handler_sig_fn.sa_handler = &sig_fn;

// Alter SIGINT signal handler to sig_fn
if(sigaction(SIGINT, &handler_sig_fn, NULL) < 0) {
  fprintf(stderr, "Failed to set SIGINT handler\n");
  return -1;
}
```

위 코드를 빌드한 프로그램은, 끊임없이 1초에 하나씩 양을 센다.

그런데 양을 세기 전에, SIGINT에 대한 핸들러를 변경시켰으므로 Ctrl-C 버튼이 눌렸을 때 sig_fn함수가 호출되게 된다.

#2 프로세스 끼리 돌아가며 카운트 하기

- Related file: src/count.c => count

목적

어떤 파일을 연다, 없으면 만들어서 열고, 첫 부분에 "0"을 쓴다.

프로세스 셋이서, 2명은 자고 1명은 깨어나서 파일에 string형식으로 쓰여진 숫자를 1증가 시켜서 다시 쓴 후, 다음 프로세스를 깨운다음 자기 자신을 다시 잠든다.

이 과정을 N번 반복하여, 결과적으로 파일에 N이 쓰여지도록 한다.

로직

1. 프로세스의 실행 인자로부터 N을 읽는다
2. SIGUSR1 시그널의 핸들러로, work함수를 설정한다.
    1. work함수는 파일을 읽어서 쓰여진 스트링이 N인지 보고,
        1. N이면 root_pid의 프로세스에 SIGUSR2 시그널을 보내고 끝낸다
        2. N이 아니면 N+1을 파일에 쓰고 next_pid의 프로세스에 SIGUSR1 시그널을 보낸다
3. SIGUSR2 시그널의 핸들러로, stop_count함수를 설정한다.
    1. stop_count함수는 자신이 루트면 child1, child2에게 SIGUSR2 시그널을 보내 종료시킨다
    2. 종료된다.
4. argv[2]` 로 들어온 파일경로를 가지고 파일을 열기 시도한다. 없으면 만들고, "0"을 쓴다.
5. 부모 프로세스는 자신의 pid를 root_pid 전역변수에 저장한다
6. 부모 프로세스가 fork해서 child1 프로세스를 만든다. 부모 프로세스는 후에, 자신이 깨울 프로세스(`next_pid`)로 child1을 지정한다.
7. child1 프로세스가 fork해서 child2 프로세스를 만든다. child1 프로세스는 `next_pid` 로 child2의 pid를 가진다.

8. child2 프로세스는 `next_pid` 로 부모 프로세스의 pid를 가진다. 그리고 `next_pid` 의 프로세스 에게 SIGUSR1 시그널을 보낸다.

위의 실행과정을 따라가게 되면, (parent, child1, child2) 프로세스가 만들어지고, child2가 parent에게 자기 자신이 다 만들어졌음을 알리는 동시에 counting의 시작을 알린다. 3개의 각 프로세스는 increment후에 next_pid의 프로세스에게 increment를 시키고. 목적지에 다다르게 되면 멈춘다.

#3 프로그램이 갑자기 종료되지 않도록 제어해보자

- Related file: src/mysignaltest.c => sigtest

목적b

프로그램이 5초 뒤에 종료된다. 그런데 Ctrl-C가 눌리면 특정 함수가 호출되는데, 이 함수는 5초가 더 넘게 걸린다. 함수가 하던 일이 멈춰지고 도중에 종료되면 에러가 날 수 있으므로, 함수가 실행중이면 5초가 지나도 종료되지 않고 이 함수를 기다리게 한다.

해결방법

- sig_int가 시작될때, 전역변수를 하나 잡아서 `doing_sig_int` 값을 1로 만든다.
- 알람에 대한 핸들러가 doing_sig_int가 1이면 종료시키지 않고 그냥 리턴한다.

```
gim-yeongjin@gim-yeongjin-ui-MacBook-Pro ~/hanyang/system/signal master ● bin/count 100000 test.txt
Opened existing file
0.722626 seconds passed
```