Xilinx Zynq FPGA, TI DSP MCU 기반의

프로그래밍 및 회로 설계 전문가

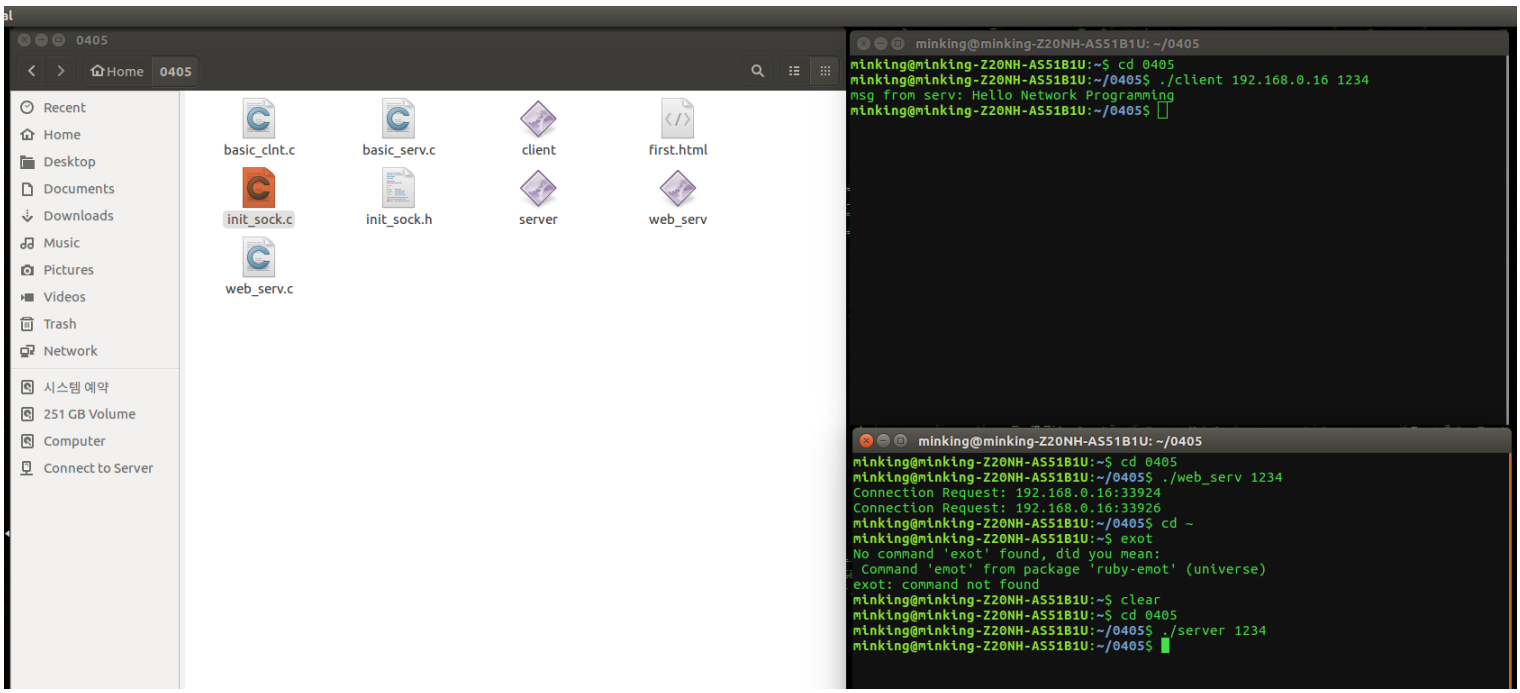
강사 이상훈

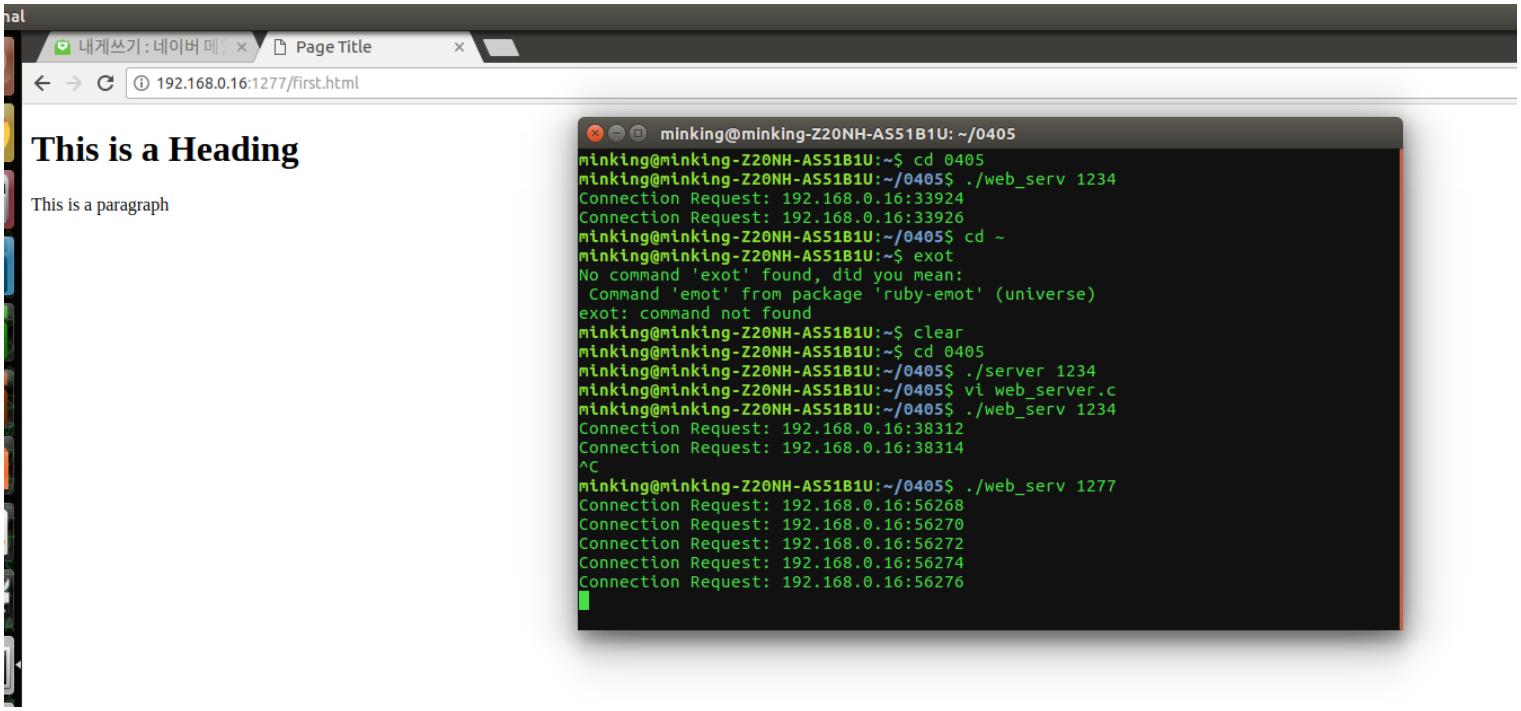
(Innova Lee)

Gcccompil3r@gmail.com

학생 김민호

minking12@naver.com


0405
Home
0405
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Network
시스템 예약
251 GB Volume
Computer
Connect to Server
basic_clnt.c
basic_serv.c
client
first.html
init_sock.c
init_sock.h
server
web_serv
web_serv.c
minking@minking-Z20NH-AS51B1U: ~/0405
minking@minking-Z20NH-AS51B1U:~$ cd 0405
minking@minking-Z20NH-AS51B1U:~/0405$ ./client 192.168.0.16 1234
msg from serv: Hello Network Programming
minking@minking-Z20NH-AS51B1U:~/0405$
minking@minking-Z20NH-AS51B1U: ~/0405
minking@minking-Z20NH-AS51B1U:~$ cd 0405
minking@minking-Z20NH-AS51B1U:~/0405$ ./web_serv 1234
Connection Request: 192.168.0.16:33924
Connection Request: 192.168.0.16:33926
minking@minking-Z20NH-AS51B1U:~/0405$ cd ~
minking@minking-Z20NH-AS51B1U:~$ exot
No command 'exot' found, did you mean:
Command 'emot' from package 'ruby-emot' (universe)
exot: command not found
minking@minking-Z20NH-AS51B1U:~$ clear
minking@minking-Z20NH-AS51B1U:~$ cd 0405
minking@minking-Z20NH-AS51B1U:~/0405$ ./server 1234
minking@minking-Z20NH-AS51B1U:~/0405$


내게쓰기 : 네이버 메
Page Title
192.168.0.16:1277/first.html
This is a Heading
This is a paragraph
minking@minking-Z20NH-AS51B1U: ~/0405
minking@minking-Z20NH-AS51B1U:~$ cd 0405
minking@minking-Z20NH-AS51B1U:~/0405$ ./web_serv 1234
Connection Request: 192.168.0.16:33924
Connection Request: 192.168.0.16:33926
minking@minking-Z20NH-AS51B1U:~/0405$ cd ~
minking@minking-Z20NH-AS51B1U:~$ exot
No command 'exot' found, did you mean:
Command 'emot' from package 'ruby-emot' (universe)
exot: command not found
minking@minking-Z20NH-AS51B1U:~$ clear
minking@minking-Z20NH-AS51B1U:~$ cd 0405
minking@minking-Z20NH-AS51B1U:~/0405$ ./server 1234
minking@minking-Z20NH-AS51B1U:~/0405$ vi web_server.c
minking@minking-Z20NH-AS51B1U:~/0405$ ./web_serv 1234
Connection Request: 192.168.0.16:38312
Connection Request: 192.168.0.16:38314
^C
minking@minking-Z20NH-AS51B1U:~/0405$ ./web_serv 1277
Connection Request: 192.168.0.16:56268
Connection Request: 192.168.0.16:56270
Connection Request: 192.168.0.16:56272
Connection Request: 192.168.0.16:56274
Connection Request: 192.168.0.16:56276

비유 1: 회사

fork , exec

fork 가 호출되면 프로그램이 1 개에서 2 개가 된다. 부모, 자식
자식이 추가되면 프로그램 실행은 부모에서 fork()리턴이 한번 실행되고 자식 fork()리턴이 실행된다.
현재 돌아가고 있던 프로세스를 복사해서 다른 프로세스를 만드는 것. 복사하기 때문에, 가지고 있던 메모리등의 시스템 자원을 모두 원래 프로세스와 공유한다.

exec 는 fork 처럼 새로운 프로세스를 위한 메모리를 할당하지 않고, exec 를 호출한 프로세스가 아닌 exec 에 의한 호출된 프로세스만 메모리에 남게 된다.
exec 는 실행의 결과로 생성되는 새로운 프로세스는 없고, exec 를 호출한 프로세스의 PID 가 그대로 새로운 프로세스에 적용되며, exec 를 호출한 프로세스는 새로운 프로세스에 의해 덮어 쓰여지게 된다.

비유 2: 카사노바 박씨 이야기

context switching

먼저 프로세스에 대한 설명이 필요하다. 간단히 말하면 프로세스란 프로그램이 메모리상에 실행중인 상태를 의미한다. 각각의 프로세스는 별도의 리소스를 할당하고, 운영체제가 대표적으로 하는 일 중 하나이다.

예를 들어서 프로세스 A 와 B 가 존재한다. A 가 running, B 는 ready 상태이다. 이 때 어떤 이유로 A 가 ready, B 가 running 상태가 되는 경우가 곧 발생할 것이다. 먼저 실행 중인 A 에 대한 데이터는 현재 레지스터에 존재할 것이다. 그리고 ready 상태인 B 는 메모리에 존재할 것이다. 하지만 이제 B 가 실행되어야 하기 때문에 A 는 레지스터를 B 에 양보해주어야 한다. 그리고 A 에 대한 데이터는 어떻게 해야할까? 그냥 버리면 될까? 아니다. A 에 대한 데이터는 B 가 실행을 마친 후에 A 가 실행될 수 있으므로 계속 가지고 있어야 한다. 즉, 현재 레지스터에 존재하는 A 에 대한 데이터는 메모리에 저장되어야 한다.
https://blog.naver.com/shumin/220559840274

비유 3: 철수와 영희 이야기

signal, block, socket, message queue

Signal 이란 Software interrupt 로, process 에 무엇인가 발생했음을 알리는 간단한 메시지를 비동기적으로 보내는 것이다.
: Signal 을 받은 프로세스는 Signal 에 따른 미리 지정된 기본 동작(default action)을 수행할 수도 있고, 사용자가 미리 정의해 놓은 함수에 의해서 무시하거나, 특별한 처리를 할 수 있다.
: 예를 들어 프로세스가 진행중에 Ctrl + c 를 누른다거나, 프로세스에 kill 명령이 내려지는것이 발생했을 때 Signal 이 발생한다.

출처: http://blockdmask.tistory.com/23 [개발자 지망생]

block

블로킹 입출력 방식은 요청한 작업이 완료되지 않으면 응답을 돌려주지 않는다.
버퍼가 비어있을 경우, 입력이 들어올 때 까지 대기하며 그 동안 다른작업을 할 수 없다.
그러므로 자원의 낭비가 발생하게 된다.

blocking i/o 와 달리 nio 방식은 다수의 클라이언트의 요청을 처리할 수 있는데,
blocking i/o 에서는 스레드를 사용하여 해결할 수 있다.

하지만 과도한 요청수 만큼 발생한 스레드(멀티쓰레드)는 OOM(Out Of Memory)를 발생 시킬 수 있다.
이를 방지하고자 스레드를 사용하고 반납하는 형식으로 스레드의 개수를 제한하는 스레드풀을 사용한다.

스레드풀의 크기를 정하는 것에는 두가지 관점에서 생각할 필요가 있다고 한다.

첫 번째는 스레드의 개수는 힙에 할당된 메모리의 크기에 의존하는데
클 수록 관리해야 할 스레드 개수가 늘어나므로 가비지 컬렉션의 수행속도가 감소한다.

두 번째는 많은 스레드가 Context Switching(CPU 를 점유하기 위한 상태변경)을 하면서
엄청난 오버헤드를 발생시킨다. 이는 cpu 성능저하의 원인이 된다.

쓰지 말라는 것이 아니고 용도에 맞게 적절하게 사용하는 것이 맞다.
하지만 멀티스레드를 사용하는 병렬처리는 공유자원에 대한 dead lock 의 위험 때문에 프로그래밍 하는것이 쉽지 않다.

출처: http://lelumiere.tistory.com/14 [Lelumiere]

socket

네트워크 프로그램 개발을 위한 TCP/IP 4 계층에서 더 추상화 해서 만들어진 계층이다. 프로그래머를 위해 만들어진 계층으로 소켓 계층은 소켓 함수 라이브러리를 말하며, 소켓 프로그래밍이란, 이 라이브러리를 사용해서 프로그래밍 하는 것을 말한다.

출처: http://eastroot1590.tistory.com/entry/socket-socket 이란 [글그리 블로그]

mesq

비유 4: 통신병 김군

tcp/ip

인터넷 표준 프로토콜으로 컴퓨터의 데이터 통신을 행하기 위해서 만들어진 프로토콜 체계
TCP/IP 는 응용 계층, 트랜스포트층, 인터넷층, 네트워크 인터페이스층의 4 개의 계층으로 구성되어 있다.
응용 계층은 사용자 응용 프로그램으로부터 요청을 받아서 이를 적절한 메시지로 변환하고 하위계층으로 전달하는 역할하는 기능을 담당한다.
트랜스포트층은 IP 에 의해 전달되는 패킷의 오류를 검사하고 재전송을 요구하는 등의 제어를 담당하는 계층으로 TCP, UDP 두 종류의 프로토콜이 사용된다.
그리고 인터넷층은 전송 계층에서 받은 패킷을 목적지까지 효율적으로 전달하는 것만 고려한다.

비유 5: 크게 성공할 미래의 주방장 이군

모르겠음 ...