시스템 프로그래밍 실습 팀 과제

팀 : 10조

조원 : 천윤서, 박규동, 이레

1. 개 요

두개의 라즈베리파이에서 앱을 실행하면 서로 실행했음을 확신하기 위해서 handshake를 실행한다. 양쪽은 앱을 실행하는 즉시 1010..을 반복해서 보내고 먼저 켜진 쪽이 10을 받으면 상대에게 10101을 전송한 뒤 잠깐 sleep에 빠진 뒤 본 통신으로 들어간다. 따라서 나중에 켜진 쪽도 10을 받고 상대가 앱을 실행함을 확신할 수 있다.

Handshake 후에 "Chat System Start! Press any key"라는 문구가 나오고 채팅을 시작한다. 우선적으로 ioctl의 CHAT_SET을 통해 인아웃모드를 활성화시킨다. 양쪽의 파이가 송수신을 모두 가능하게 하기 위해 app에서 send와 receive는 프로그램이 실행되면 서로 다른 스레드에 놓인다. 파이1이 "hi"를 치면 send_thread에서 한 char마다 ioctl의 CHAT_SEND_CHAR을 실행한다. Ioctl의 CHAT_SEND_CHAR는 먼저 Start Bit인 0을 CLOCK만큼 주고 드라이버에서 AscToStr의 함수를 통해 먼저 들어온 'h'에 해당되는 아스키코드를 1과0의 배열로 담아 send_buf 함수로 전송한다. 파이간 싱크를 맞추기 위해 각 비트마다 CLOCK만큼 msleep을 걸어준다. 파이는 모든 글자를 전송하고 Stop bit인 1로 Output을 바꾼다.

다른 스레드는 CHAT_RECIVE_CHAR에서 0이 들어오기를 대기하고 있다가 0이 들어오면 CLOCK+1만큼 쉬고, 비트를 받기 시작한다. receive_buf 함수가 이 역할을 수행하며, 비트를 버퍼에 담는다. 버퍼에 담긴 비트는 StrToAsc함수를 통해서 아스키코드로 변환되고, app의 recive_thread를 통해 파이2의 터미널에 'h'가 출력된다. 이 과정을 문자열 구분자가 올때까지 반복하여 문장을 완성한다.

2. 구현 사항

과제의 기본 사항을 전부 구현하였고, 멀티 스레드를 통해서 App하나로 송/수신을 전부 구현하였다. 제시된 3개의 선만을 이용하여 모든 문제를 해결하였고, 디바이스 드라이버 하나로 송/수신 기능을 구현하였다. 또한 ASCII코드를 비트로 치환하는 함수를 통해서 영문 소문자 외에 모든 아스키 문자를 송수신하도록 만들었다

3. 팀원별 역할 분담

천윤서 : 조장, 드라이버, 어플리케이션, Receive function, 멀티스레드, 디버깅

박규동 : 드라이버, 어플리케이션, Send function, 비트변환, 멀티스레드, 보고서

이레 : 드라이버, 어플리케이션, 비트변환, Send&Receive 버퍼, 디버깅, 보고서

4. 고 찰

박규동 : 이번 통신과제는 예전에 컴퓨터 통신이나. 컴퓨터 네트워크, 운영체제에서 배웠던것들을 활용하는 과제였다. 막히는 부분에서 구글링을 참고하려했으나 대부분 라이브러리에 정의된 함수들을 사용하는 코드라서 사용이 불가능했다. 과제 구현 중에서 아스키를 비트로 바꾸고 다시 변환하는 파싱과정이나, 그에 해당되는 비트를 보내고 받는 것은 어렵지 않았으나 양쪽의 파이간 송수신에서 클록을 이용해 싱크를 맞추는 과정이 어려웠다. 한비트만 잘못 전송되어도 글자가 깨졌기 때문에 이걸 조정하는게 가장 난해한 부분이었다고 생각한다. 적절하게 싱크를 맞추는 과정은 반복된 디버깅을 통해 해결했다.

천윤서 : 굉장히 짧은 시간 단위를 다룰 때에는 dmesg를 이용해서 양쪽의 디바이스 상태를 확인해서 디버깅을 정밀하게 해야함을 알았다. 파이를 오래 사용해야해서 열에 의해 쌓인 디바이스의 피로가 결과에 영향을 줄 수 있다는 점을 깨달았다. 또 핀에는 쓰레기값이 남는다는 것을 배워서 서로 다른 두 디바이스를 다룰 때에는 적절한 초기값 처리 방식이 들어가는 것이 꼭 필요하다는 점을 알았다. 중간에 출력과 입력의 함수를 각각 다른 조원이 짜서 처리 시간이 달랐는데 같은 방식으로 맞추니 성능이 향상되었다. 이로 인해 ms단위의 굉장히 짧은 시간 단위에서는 함수의 실행 속도가 결과에 영향을 미칠 수 있다는 점을 알았다.

이레 : 가산점의 규칙을 해결하기 위해선 gpio14와 gpio15만을 사용해서 해결해야했다. 이는 asynchronous방식 채택해 사용하는 것이다. 이를 위해서는 clock에 대한 정의를 잘 해내야 하며 start, stop bit가 필요하다. 한 char 문자를 보내기 위해서 start bit와 stop bit에 대한 정보를 붙여야 했다. 뿐만 아니라 초기값이 어떻게 정의되어 있는지를 정의해 놓은 상태로 시작해야했다. 실제로 다른 핀을 추가로 사용했다면 더 쉽게 구현 할 수 있었다.