일반 논문 ( ) / 학생 논문 (O)

원형 큐를 이용한 다중 터미널 에뮬레이터의 수신 성능 개선 방법

최홍석, 김윤수, 김진수, 전중남*
충북대학교 소프트웨어학과

Improving Receiver Performance of Multi-Terminal Emulator Using Circular Queue

Hong-Soek Choi, Youn-Su Kim, Jin-Su Kim, Joong-Nam Joen*
Department of Computer Science, Chungbuk University

요 약

기존의 터미널 에뮬레이터 프로그램들은 단일 연결만 지원하여 일대일 통신으로 주요 사용하였다 하지만 IoT의 발전으로 다양한 장치들과 데이터 송수신이 필요하게 되었으며, 그에 대한 테스트 도구의 필요성을 충족시키기 위해 일대다 연결이 가능한 다중 연결 터미널 에뮬레이터를 개발하였다. 또한 데이터의 안정성을 보장하기 위해 수신부에 원형 큐 버퍼를 추가하였고, 그 결과 극한 상황에서도 데이터를 안정적으로 송수신하여 IoT 장치의 통신 테스트에서 신뢰도를 보장받을 수 있다.

1. 서론

사물인터넷의 발전으로 다양한 장치가 통합관리 됨에 따라 기존의 일대일 터미널 에뮬레이터는 사용성적인 측면에서 불편한 점을 가져왔다. 따라서 통합관리 및 일대다 환경에서도 성능이 떨어지지 않는 터미널 에뮬레이터의 필요가 대두되었다. 하지만 다대일의 경우 통신 장치 간 간섭이 발생하거나 데이터 유실 등의 문제가 발생할 수 있다. 따라서 우리는 이러한 점을 보완하여 안정성을 보장하는 프로그램을 개발하였다.

개발된 프로그램은 TCP/UDP와 시리얼을 동시에 연결할 수 있으며, 연결된 상태에서 데이터의 송수신도 동시에 수행하는 기능을 가졌다. 그리고 동시 연결된 장치들과의 통신은 독립된 객체로 병렬적으로 데이터를 송수신 하도록 구성하였다. 따라서 각 연결은 서로의 간섭 없이 데이터 송수신이 가능하다. 하지만 장치에서 프로그램으로 다량의 데이터를 빠른 속도로 전송할 경우 데이터의 유실과 프로그램의 메모리가 부족한 문제가 발생하였기에, 본 프로그램은 데이터 수신부에 원형 큐로 구성된 버퍼를 추가하여 프로그램의 안정성을 높였다.

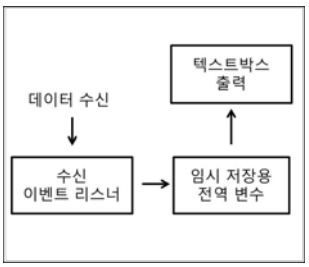
따라서 본 논문은 프로그램 개발의 목적이었던 안정성을 입증하기 위해 송수신 절차에서 원형 큐 버퍼 알고리즘을 적용하기 전과 후를 비교하고, 그 결과를 보이고자 한다.

2. 관련연구

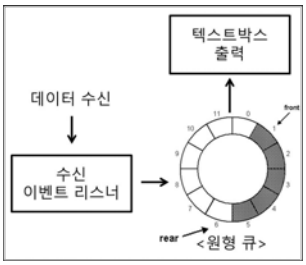
원활한 통신을 위해 버퍼를 사용하여 혼잡제어를 줄이는 방법이 존재한다. 정적 버퍼 용량을 동적으로 변경하고, 동시에 신속한 접근과 최적 용량을 통해서 네트워크에서 많이 발생되는 패킷의 폐기 과정을 최소화 할 수 있었다. 특히 오버플로 현상, 서버의 병목현상에서 자주 발생한다. 따라서 본 논문에서는 수신 버퍼를 완충 장치로 사용하여 프로그램이 다운되지 않고도 정확하게 패킷을 송수신 할 수 있는 방식을 제안하였다[1].

동적 버퍼를 사용하는 방법에 대해서 윈도우 기법이라는 방식을 사용한다. 이를 이용한 방법은 패킷을 전부 전송하지 않고, 나누어 송신 한다. 송신된 일부를 제외한 나머지는 추가적인 버퍼로 사용할 수 있어 효과적인 전송속도와 용량을 보장하며 다중 클라이언트로 전송 시 더욱 효과적이다. 전송 실패가 된 부분은 ACK를 이용하여 재전송 할 수 있기에 손실에 대응하고, 속도적인 측면과 안정적인 측면, 다수에게 효과적으로 전달될 수 있는 장점이 있다[2].

3. 원형 큐 버퍼를 이용한 데이터 수신 방법



[그림 1] 개선 전 수신 구조



[그림 2] 개선 후 수신 구조

그림 1의 개선 전 수신 구조에서는 데이터 송수신은 C#에서 제공하는 라이브러리의 리스너를 이용하여 임시 변수에 저장한 뒤 바로 출력하였다. 하지만 이 경우 전송장치에서 while(1) 의 반복주기로 데이터를 전송할 경우 출력하는데 문제가 발생하였

*Corresponding author : Joong-Nam Joen (chs93114@gmail.com)

다. 그 문제는 빠른 주기로 데이터 수신하면, 데이터는 계속해서 수신해도, 출력하는데 시간이 걸리기 때문에 메모리 부족문제가 발생하여 테스트를 지속할 수가 없었다. 따라서 문제를 해결하기 위해 그림 2와 같이 임시 변수를 원형 큐로 대체하여 문제를 해결하였다.

본 프로그램에서는 데이터 수신의 안정성을 보장하기 위하여 데이터를 출력하기 전 원형 큐에 데이터를 누적시켰다. 그 후 부하가 걸리지 않는 데이터 출력 주기를 테스트하여 최적 값을 설정하였다. 표 1은 원형 큐가 데이터를 출력하는 주기에 따른 부하시간을 측정한 결과이다. 그 결과 20ms의 주기로 설정했을 때, 부하까지 버티는 안정성과 데이터 출력 적절성의 균형을 잡을 수 있었다.

[표 1] 출력 주기에 따른 프로그램 부하까지의 시간

원형 큐 출력 주기	5ms	10ms	15ms	20ms
부하까지의 시간	69s	71s	75s	263s

여기서 부하란, 프로그램의 처리가 출력에 집중되어 CPU가 처리해야 하는 데이터가 증가하여 다른 명령을 처리하기 어려운 상태를 말한다. 데이터 수신 상태가 원활해지면 프로그램은 다시 원상태로 복귀된다.

데이터 수신이 계속되어 원형 큐에 데이터가 가득 차서 더 이상 데이터를 받을 수 없는 경우에는 원형 큐에 데이터를 넣을 공간이 생기기 전까지는 데이터의 수신을 거부한다. 또한 원형 큐 자료구조의 특성상 데이터의 앞뒤가 항상 구분되기 때문에 데이터가 뒤바뀌는 문제도 방지할 수 있다.

4. 구현 결과

4.1. 실험 환경 구축

본 논문에서는 간단한 데이터 송수신 부하 실험 모델을 만들기 위해서 아두이노와 HC-06 모듈을 사용하였다. 아두이노에 'while(1)'을 통해 반복적으로 데이터를 송신하도록 세팅한 뒤, PC와 아두이노를 블루투스를 통해 연결하였다. 사용된 PC의 사양은 "CPU: i5-4690, RAM:16G, Windows 10 Pro"이다.

테스트 방법은 원형 큐 버퍼의 사용 여부에 따른 안정성 보장을 정확히 측정하기 위해 버퍼를 사용하지 않았을 때와 사용했을 때 메모리 부족 오류가 발생하는 시간을 초단위로 측정하고, 똑같은 과정을 각각 10회 반복 진행하였다.

4.2. 테스트 결과 및 고찰

표 2의 버퍼를 적용하지 않았을 경우를 보면 대부분의 경우 4s 전후로 데이터 패킷 송수신 속도보다 화면에 입출력하는 속도가 늦기 때문에 프로그램에서 메모리부족 에러가 발생하며 멈추었다. 또 17s 이상 문제없이 송수신이 되는 경우가 테스트 결과 3회 있었지만, 끝까지 지속되지 못하고 결국은 에러가 발생하므로 안정성이 보장되고 있다고 할 수 없다.

버퍼를 적용했을 경우는 메모리 부족으로 프로그램에 에러가 생겨 멈추는 경우는 없었다. 하지만 수신이 계속되면, 출력부 처리에 따른 문제로 평균 29.5ms 시간에 프로그램에 부하가 발생하였다. 또한 버퍼가 가득 찬 후 수신한 데이터에 대해서는 내용이 단절되었다. 하지만 빠른 주기로 보내던 통신 장치에서 송신을 중지하면 프로그램은 무사히 원상태로 복귀되고, 원형 큐에 있던 나머지 내용들도 모두 출력되었다. 따라서 프로그램의 안정성을 보장할 수 있었다.

[표 2] 테스트 결과

버퍼 미적용	테스트 횟수	1회	2회	3회	4회	5회
	에러 발생 시간	3.3s	4.27s	7.61s	2.82s	3.28s
	테스트 횟수	6회	7회	8회	9회	10회
	에러 발생 시간	17.1s	21.5s	18.1s	4.26s	3.06s
버퍼 적용	테스트 횟수	1회	2회	3회	4회	5회
	부하 발생 시간	9.7s	8.8s	12.2s	57.1s	11.8s
	테스트 횟수	6회	7회	8회	9회	10회
	부하 발생 시간	23.5s	55.1s	34.8s	37.3s	44.7s

5. 결 론

이 논문에서는 원형 큐 버퍼를 이용한 데이터 수신 방법을 제시하였다. 그 결과 기존에는 극한 상황에서 프로그램이 강제 종료되어 테스트가 지속될 수 없었지만, 원형 큐 버퍼를 추가하여 비록 부하 상태지만 테스트는 지속될 수 있도록 개선하였다.

그 결과 매우 빠른 주기에서도 동작을 요하는 극한 환경의 장치도 테스팅을 할 수 있도록 안정성을 확보하였고, IoT 장치의 통신 테스트에서 신뢰도를 보장받을 수 있다.

이러한 방법의 도입은 비단 테스트 분야에서 뿐만 아니라, 극한 상황에서 프로그램을 구동시키기 위한 방법으로도 활용될 수 있을 것이다.

ACKNOWLEDGMENTS

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 서울어코드 활성화 지원사업(IITP-2017-2012-0-00598)의 연구결과로 수행되었음.

참고문헌

[1] H. J. Kim & I. H. Ra. (2005). A Dynamic Buffer Allocation and Substitution Scheme for Efficient Buffer Management. *Journal of the Korea Institute of Information and Communication Engineering, 9(1),* 128-133.

[2] C. C. Choi & S. Y. Lee. (2004). Study of Network Adaptation Buffer for Communication Model. *KOREA INFORMATION SCIENCE SOCIETY, 31(1A),* 454-456.