버킷 스케쥴러를 이용한 ATM 셀 스페이서

김영섭*· 박상택· 심재철

한국전자통신연구원, 교환·전송연구소, ATM정합팀

The ATM Cell Spacer Using Bucket Scheduler

Young-Sup Kim*, Jae-Chul Shim, Sang-Taick Park

ATM Interface Team, S-T Lab, ETRI

E-mail : youngs@etri.re.kr

요 약

ATM망에서는 악의의 가입자가 협상된 대역폭 이상을 사용하여 발생하는 망의 폭주에 의해 선의의 가입자의 서비스 품질이 저하되는 것을 방지하기 위해 사용자 파라메터 감시를 수행한다. 그런데 사용자가 대역폭을 준수하여 셀을 전송한다고 하더라도 교환기, 다중화기등을 거치면서 여러 연결의 셀들이 다중화 되면 CDV가 발생하여 사용자 파라메터 감시를 수행하는 장치에서 대역폭 위반으로 검출하여 셀이 손실되거나 셀 전달 우선 순위가 변경되어 서비스의 품질이 저하 될 수 있다. 이를 방지하기 위해 망의 장치는 셀을 출력할 때 장치에서 발생된 CDV를 보상하여 원래의 셀 스트림의 형태를 복원하여 전송함으로써 서비스 품질의 저하를 방지한다. 이를 스페이서라고 하는데, 본 논문에서는 이러한 스페이서의 구현 방식을 살펴보고, 메모리가 적게 사용되는 셀 도착 구동 방식의 스페이서 구조를 제안하고 실험을 통해서 그 성능을 평가 하였다.

I. 서 론

ATM 망에서 사용자가 점유하는 대역폭은 호 설정 시에 협상에 의해 설정된다. 망에서는 폭주를 예방하기 위해 허용 가능한 대역폭을 가입자에게 할당하여 가입자가 전송하는 ATM 셀이 폭주로 인해 셀 손실이 생긴다든지 지연이 크게 발생하는 것을 방지하여 고 품질의 통신 서비스를 제공하여야 한다. 그러나 악의의 가입자가 협상된 대역폭을 준수하지 않고 임의로 협상된 대역폭보다 높은 속도로 셀을 전송하게 되면 망에서는 폭주가 발생하게 되고 이로 인해 협상된 대역폭을 준수하여 셀을 전송하는 선의의 가입자가 셀 손실이나 지연의 품질 저하를 겪게 된다. 이를 방지하기 위해 ATM 망의 정합 장치에서는 사용자가 협상된 대역을 준수하여 셀을 전송하고 있는지 감시하여 위반한 셀을 검출하면 우선 순위를 변경하거나 셀을 폐기하는 등의 조치를 취함으로써 선의의 사용자들을 보호하여야 한다. 이러한 기능을 수행하는 것을 사용자 파라미터 감시(Usage Parameter Control : UPC)라고 한다.[1][2] 그런데 사용자가 협상된 대역을 준수하여 셀을 전송한다고 하더라도 망의 장치인 다중화기와 스위치 등을 거쳐 전송되는 과정에 여러 개의 연결들을 다중화 함으로써 사용자가 전송한 원래의 셀 스트림 형태에서 변형되어 셀 지터가 발생하게 되는데 이 지터를 흔히 CDV(Cell Delay Variation)이라고 한다.[1][2] 이 CDV로 인하여 UPC를 수행하고 있는 망의 장치들이 협상된 대역폭을 위반한 것으로 간주하게 될 수 있다. 이를 방지하기 위하여 망의 전송 장치는 UPC를 통과하여 입력된 셀 스트림을 다른 장치로 전송할 때 장치내에서 발생된 CDV를 보상하여 원래의 셀 스트림 형태로 복구하여 전송함으로써 사용자에게 고품질의 서비스를 제공하고, 망의 사용 효율을 높일 수 있다. 이와 같이 셀 지터를 보상하기 위한 장치를 셀 스페이서(spacer)라고 한다.[1-3]

셀 스페이서는 기본적으로 2개의 블럭으로 구성될 수 있다. 하나는 CDV를 측정하기 위한 CDV 계산기, 그리고 측정된 CDV 만큼 지연을 인가하여 원래의 셀 스트림 형태로 복원하는 스케쥴러로 구성된다.[3][4] CDV의 계산은 각 연결별로 협상된 대역폭의 파라메터를 저장하고 셀이 입력 혹은 출력될 때마다 계산하게 되는데 CDV를 계산하기 위한 방법으로는 VSA(Virtual Scheduling Algorithm)과 리키 버킷 알고리즘에 의한 계산 방식이 있다.[4][5] 또한 셀이 도착할 때 도착된 셀들의 CDV를 측정하여 이를 보상해 주는 셀 도착 사건 구동 방식과 셀이 출력 될 때 CDV를 측정하는 셀 출력 사건 구동 방식이 있다.[6] 그리고, 스케쥴링 방식에는 출력 셀 시간을 정렬하여 가장 작은 출력 시간을 갖는 연결의 셀

을 출력하는 방식과 카렌다를 이용하는 방식, 그리고 버킷 스케줄러를 이용하는 방식등이 있다.[5]

본 논문에서는 CDV를 계산하는 방식은 VSA로 하고, 셀 도착 사건 구동 방식과 셀 출력 사건 구동 방식에 대한 비교를 통해 셀 도착 사건 구동 방식을 사용하는 버킷 스케줄러를 이용하는 셀 스페이서에 대한 구조를 제시하고, 실험을 통해 제안된 셀 스페이서의 성능을 제시한다.

II. 셀 도착 사건 구동 방식

셀 도착 사건 구동 방식은 셀 스페이서로 셀이 입력될 때 이론적으로 이전 셀이 출력된 시간과 현재의 셀이 입력된 시간으로부터 현재의 셀이 이론적으로 출력될 시간을 계산하는 방식을 말한다. 이를 계산하는 방식으로는 VSA를 이용하는 방법과 리키 버킷 알고리즘을 이용하는 방법이 있지만 본 논문에서는 VSA를 이용하는 방식에 대해서만 검토한다. VSA를 이용한 이론적 출력 시간 계산 방식은 그림 1에 나타내었다. 셀이 입력되면 이전 셀의 이론적 출력 시간(Theoritical Departure Time : TDT)에 협상된 최소 셀간 간격 T를 더해 현재의 시간 t와 비교를 수행한다. 여기서 t가 큰 경우에는 이 셀을 바로 출력해도 협상된 최대 셀 속도를 위반하지 않는 것이 되기 때문에 새로운 TDT는 t가 된다. 그러나 t가 작은 경우에는 TDT+T에서 t를 뺀 만큼의 지연을 갖게 하기 위하여 새로운 TDT가 TDT+T가 된다.

셀 도착 사건 구동 방식은 스위치나 다중화 장치 등의 장치에서 발생하는 최대 CDV 만큼만 셀을 지연 시키면 되기 때문에 셀 스페이서가 가져야 하는 셀 버퍼는 허용하는 최대 CDV의 크기를 갖는 셀 버퍼를 구비하면 된다. 따라서 셀 버퍼로 사용되는 메모리의 양이 비교적 작다. 그러나 다중화된 연결의 셀들에서 TDT가 충돌하게 되면 실제 지연은 CDV보다 커질 수 있으며, 이로 인해 또 다른 CDV가 발생하여 서비스 품질의 저하를 가져올 수 있다.

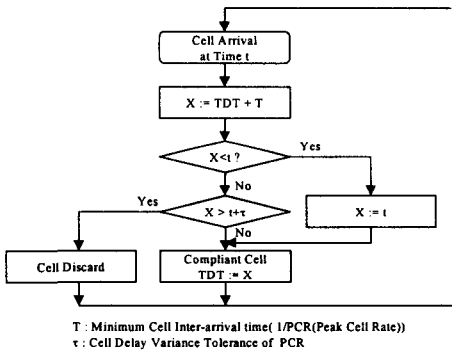

그림 1. VSA를 이용한 셀 도착 사건 구동 방식의 흐름도

III. 셀 출력 사건 구동 방식

셀 출력 구동 방식은 임의의 연결의 셀이 출력될 때 해당 연결의 다음 셀이 출력될 시간을 계산하여 해당 연결의 셀을 스케줄러로 전달하는 방식으로 해당 연결의 다음 셀을 찾아서 스케줄러로 전달하기 위해서는 각 연결 별로 셀 버퍼를 관리하여야 한다. 그리고 다음 셀이 출력될 시간이 계산되었지만 해당 연결의 버퍼가 비어 있을 때는 셀을 스케줄러로 전달하지 못하기 때문에 출력될 시간을 저장하였다가 새로운 셀이 입력될 때 그 시간과 계산된 셀 출력 시간과 비교하여 큰 값으로 도착된 셀의 출력 시간을 결정하여야

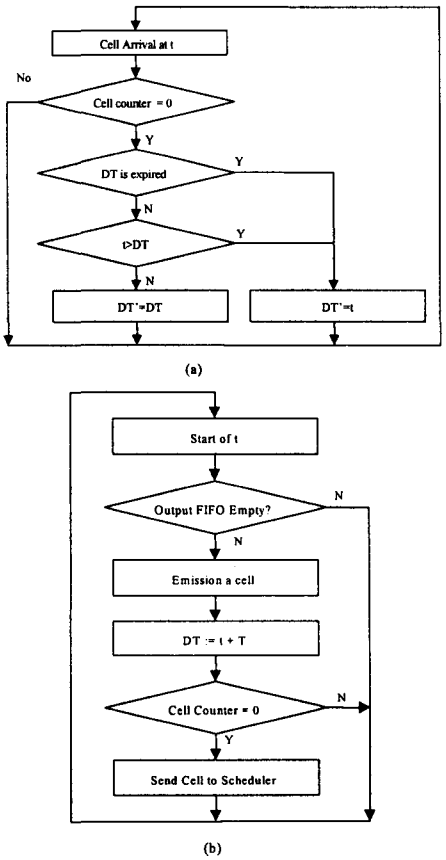

그림 2. 셀 출력 사건 구동 방식의 흐름도
(a) 셀 도착 프로세스, (b) 셀 출력 프로세스

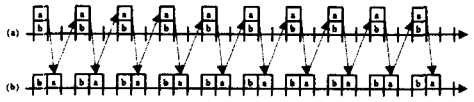

그림 3. 출력 셀 시간 충돌의 예 (a) 계산된 셀 출력 시간, (b) 스페이서 출력 셀 스트림

한다. 따라서 셀 도착 프로세스와 셀 출력 프로세스로 두개의 프로세스가 존재하여야 하며 연결별로 버퍼를 구성하여야 하기 때문에 실제 구현 상에서는 회로가 복잡하고 메모리가 많이 소요된다. 셀 출력 사건 구동 방식의 두가지 프로세스의 절차를 그림 2에 나타내었다.

한편 셀 출력 사건 구동 방식에서는 협상된 대역이 비슷한 2개의 연결에서 셀 출력시간의 충돌이 발생하면 어떤 하나의 연결에 계속적인 지연이 발생하게 되어 셀 버퍼를 아무리 크게 해도 셀 손실이 발생할 수 있다. 예를 들어 최소 셀간 간격이 2인 연결 a와 3인 연결 b의 셀이 그림 3(a)와 같이 계산되어 출력 셀 시간의 충돌이 발생하였다고 하면 출력은 그림 3(b)와 같이 된다. 따라서 연결 a는 셀 스페이서를 거치면서 최소 셀간 간격이 3인 셀 스트림으로 변형되고 결국 셀 손실을 초래하게 된다.

IV. 버킷 스케쥴러

버킷 스케쥴러는 각 셀 시간에 일정한 양의 버킷을 준비하고 CDV 계산기로부터 수신한 셀을 셀 버퍼에 기록하고 출력 셀 시간의 버킷에 셀 버퍼의 포인터를 기록하는 셀 쓰기 제어부와 현재 셀 시간의 버킷에 저장되어 있는 셀 버퍼 포인터를 출력 버퍼에 저장하고, 출력 버퍼에 저장되어 있는 셀 버퍼 포인터를 읽어 셀 버퍼로 부터 셀을 읽어 출력시키는 셀 읽기 제어부로 구성된다. 버킷 스케쥴러의 구성을 그림 4에 나타내었다.

버킷 스케쥴러의 동작은 우선 셀이 입력되면 셀 버퍼에 셀을 저장할 수 있는 유휴 버퍼 포인터를 갖고 있는 유휴 어드레스 FIFO(Idle Address FIFO)로부터 유휴 버퍼 어드레스를 읽어와서 해당 어드레스에 셀을 기록한다. 그리고 그 버퍼의 포인터를 TDT의 버킷에 저장한다. 버킷은 각 셀 시간별로 셀 버퍼 포인터를 기록할 수 있는 메모리와 각 버킷에 저장된 셀의 수를 계수하는 카운터로 구성된다. 칼렌다 제어부에서는 출력 시간과 셀의 버퍼 포인터를 수신하면 해당 출력 시간의 버킷에 셀 카운터를 하나 증가 시키고 해당 버킷 메모리에 셀 버퍼 포인터를 기록한다. 그리고 현재 시간 t의 버킷에 있는 셀 버퍼 포인터들을 읽어서 출력 버퍼(Emission FIFO)에 저장한다. 버킷에서 셀 포인터를 읽어 낼 때는 우선

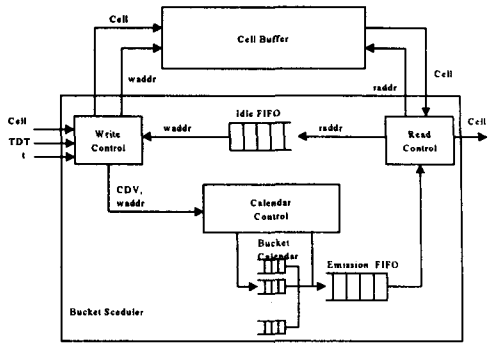

그림 4. 버킷 스케쥴러의 구성

버킷에 저장된 셀의 수를 읽은 후에 그 버킷에 있는 모든 셀 포인터를 출력 버퍼로 저장한다. 따라서 각 버킷 메모리는 FIFO와 같은 동작을 하게 된다.

출력 제어부는 한 셀 시간동안 출력 FIFO에 저장되어 있는 한 개의 셀 포인터를 읽어 셀 버퍼로부터 셀을 읽어 출력한다.

그림 4에 나타낸 버킷 스케쥴러는 FPGA를 이용하여 구현하였다. 버킷 메모리는 FPGA 내부의 SRAM을 이용하여 구성하였으며, 유휴 셀 버퍼 어드레스 FIFO와 출력 FIFO는 FPGA 내부에 SRAM을 이용한 FIFO로 구성하였다. 시스템에서 발생하는 CDV를 128셀 시간 이하로 고려하여 셀 버퍼로는 128개의 셀을 저장할 수 있는 8K 바이트 용량의 DPRAM으로 구성하였다.

버킷 메모리는 128개의 버킷을 가지며 각 버킷마다 7비트의 포인터를 저장하기 위한 메모리 7개와 카운터로 사용되는 메모리 1개로 구성되므로 7168비트의 SRAM으로 구성하였다. 또한 유휴 셀 버퍼 어드레스 FIFO와 출력 FIFO는 1287비트의 SRAM을 이용하여 구성되었다. 따라서 버킷 스케쥴러 내부에는 총 8960비트의 메모리를 사용하였다. 이는 셀 출력 사건 구동 방식을 사용하였을 때 구비하여야 하는 연결별 큐 및 연결별 큐를 제어하기 위해 사용되는 메모리에 비하면 아주 작은 메모리의 소요량이다.

V. 성능 평가

구현된 셀 스페이서는 ATM 교환기의 가입자 정합 장치에 적용되어 셀 스페이스 기능에 대한 실험을 수행하였으며 성능을 평가하기 위해 시뮬레이션을 수행하였다.

실험은 셀 스페이서가 적용된 ATM 교환기의 가입자 정합 장치에 ATM 분석기를 연결하고 셀을 루프백 시켜 ATM 분석기에서 셀을 수신하여 셀간 간격의 히스토그램을 확인하였다. 입력 셀 스트림의 형태는 그림 5에 나타낸 바와 같은 버

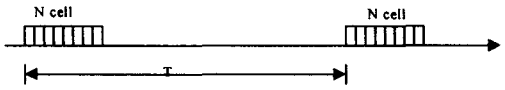

그림 5. 입력 셀 스트림

스트 트래픽을 인가하였다. 평균 셀 전송 속도와 버스트 길이 N을 지정하면 N개 셀이 연속적으로 발생되고, 발생을 멈추었다가 하는 식으로 평균 셀 전송율을 유지하는 트래픽이다.

입력 셀 스트림의 평균 전송율은 다음과 같다.

$$BW = \frac{N \times 424}{T} (bit/\sec)$$

실험은 평균 셀 전송 속도가 42.4Mbps이고 N이 16일때, 21.2Mbps이며 N=16인 경우 그리고 12.72Mbps이고 N=8인 경우에 대하여 실험하였으며 시험 결과를 각각 그림 6에서 8에 나타내었다. 스페이서에 설정된 PCR이 42.4Mbps인 경우에는 평균 최소 셀 간격이 3.67셀 시간인데 그림 6에 나타낸 실험 결과를 살펴보면 최소 셀 간격이 2셀이고 4셀 간격을 갖는 것이 약 59.3% 3셀 간격이 약 37.5%로 스페이싱 되고 있음을 알 수 있다. 또한 21.2Mbps인 경우에는 평균 최소 셀 간격이 7.34셀 시간인데 그림 7에 나타낸 실험 결과를 살펴보면 6셀 간격의 셀이 17.2%, 7셀 간격이 59.1%, 8셀 간격이 23.7% 정도의 분포를 보였다. 12.72Mbps인 경우에는 최소 평균 셀 간격이 12.23셀 시간이며 그림 8의 실험 결과는 11셀 간격의 셀이 33.6%, 12셀 간격이 55.3%, 13셀 간격이 11.1% 정도의 분포를 보였다.

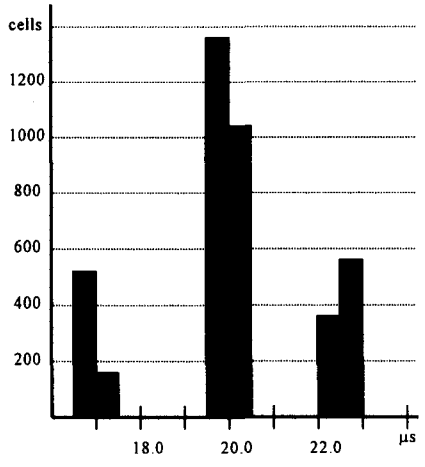

그림 7. PCR=12.72Mbps, N=8일 때의 실험 결과

실험 결과에 나타낸 바와 같이 버킷 스케쥴러를 이용한 셀 스페이서는 입력 셀 스트림이 버스한 특성을 갖더라도 설정된 PCR로 셀 스트림을 스페이싱함을 알 수 있다.

VI. 결 론

본 논문에서는 ATM 셀 스페이서의 필요성과 셀 스페이서를 구성할 때 CDV를 측정하는 방식에 대한 검토를 수행하였으며 이를 바탕으로 셀 도착 사건 구동 방식의 VSA를 이용한 CDV 계산

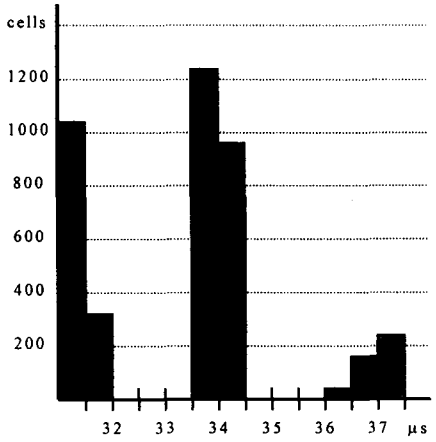

그림 6. PCR=21.2Mbps, N=16일 때의 실험 결과

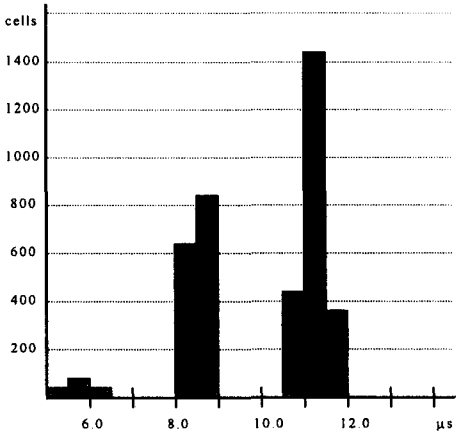

그림 8. PCR=42.4Mbps, N=16일 때의 실험 결과

기를 구현하였고, 계산 CDV에 의한 셀 스페이싱을 하기 위해 버킷 스케줄러를 사용한 스케줄러의 구성을 제시하고 일반적인 상용 FPGA를 이용하여 이를 구현하였으며 구현된 회로를 ATM 교환기의 가입자 정합 장치에 반영하였다. 그리고, ATM 교환기의 가입자 정합 장치와 ATM 분석기를 이용하여 셀 스페이서의 기능을 실험하여 그 결과를 제시하였다. 시험 결과 버스트한 셀 스트림이 입력되어도 설정된 PCR로 셀 스트림을 변형하여 출력하고 있음을 알 수 있었다.

한편 셀 도착 사건 구동 방식의 CDV 계산 알고리즘을 사용한 셀 스페이서는 연결별로 큐를 구성하지 않고 전송 장치에서 발생되는 최대 CDV 만큼의 셀 버퍼를 구비하기 때문에 아주 작은 메모리의 양으로도 셀 스페이서를 구성할 수 있었으며, 상용 FPGA로 구현할 수 있었다.

우리는 본 논문에서 PCR에 대한 셀 스페이서의 구현만을 검토하였으나 추후 SCR을 고려한 ATM 셀 스페이서를 FPGA를 이용하여 개발할 예정이다.

[참고문헌]

[1] ATM Forum, "Traffic Management Specification 4.0," af-tm-0056.000, Apr., 1996.

[2] ITU-T Recomendation I.371, "Traffic Control and Congestion Control in B-ISDN," Geneva, May, 1996.

[3] Pierre E. Boyer, Michel J. Servel, Fabrice P. Guillemin, "The Spacer-Controller : An Efficient UPC/NPC for ATM Networks," Paper A9.3 in Proceedings of ISS'92, Yokohama, Japan, pp316-320, Oct. 1992.

[4] Pierre E. Boyer, Fabrice M. Guillemin, Michel J. Sevel, and Jean-Pierre Coudreuse, "Spacing Cells Protects and Enhanceds Utiliztion of ATM Network Links," IEEE Network, pp38-49, Sep., 1992.

[5] 김철규, 전만영, 박홍식, " 셀 스페이서 구조 연구 및 버퍼 크기 분석," 96하계 학술 발표 대회 논문집 19권 1호, pp89-92, 대한 전자 공학회, 1996.

[6] Jun Shik Hong, "Design of an ATM Shaping Mutiplexer Algorithm and Architecture," Doctorial Thesis, Polytechnic University, Aug. 1996.