論文 96-1-1-12

VLSI-Implementation of the Virtual Scheduling Algorithm

Man-Yeong Jeon*, Hong-Shik Park* Regular Members

Virtual Scheduling Algorithm의 VLSI 구현

正會員 田 萬 泳*, 朴 弘 植*

ABSTRACT

Proposed numerous algorithms for the policing function have mainly focused on their performances. Besides their performance evaluation, however, the VLSI-implementation of these algorithms is worth consideration as well. Although, no algorithms for the policing function have been standardized up to now, ITU-T I.371 suggests two examples of algorithms, the Virtual Scheduling Algorithm (VSA) and the Continuous State Leaky Bucket algorithm. In this paper, we suggest the architecture of a policing device implementing the VSA among various algorithms for the peak cell rate policing and discuss some issues on the implementation. We also present how to select the policing modes of the two devices used to realize various policing schemes and show the experimental results obtained under four different peak cell rate values to confirm that the device performs the policing function satisfactorily. We exploit the priority encoder to run the algorithm in parallel instead of sequentially, which reduces the operation time to a great extent.

要 約

셀 속도 감시를 위하여 현재까지 제안된 여러 알고리즘들은 주로 그들의 성능에 관한 것이었다. 그러나 성능에 관한 논의와 더불어 이들 알고리즘의 VLSI 구현에 관한 논의 또한 고려해볼 가치가 있으리라 생각된다. 현재까지 셀속도 감시를 위하여 ITU-T에 의해 표준화된 알고리즘은 없으나 ITU-T는 셀속도 감시를 위한 알고리즘의 두가지 예로 Virtual Scheduling Algorithm(VSA)과 Continuous State Leaky Bucket Algorithm을 제시하고 있다. 본 논문에서는 최대 셀 속도 감시를 위한 여러 알고리즘 중 VSA를 구현하는 디바이스의 설계구조를 제시하고 설계시 고려해야할 몇가지 사안에 관해 논한다. 또한 다양한 셀 속도 감시 스킴을 실현하는데 사용되는 두 디바이스의 감시 모드를 선택하는 방법에 관해 기술하며 구현한 디바이스가 감시 기능을 만족스럽게 수행하는 것을 확인하기 위해 네가지 최대 셀 속도 계약치 하에서 얻어진 실험 결과를 제시한다. 본 논문에서는 VSA를 순차적으로 수행하는 대신 priority encoder를 이용하여 병렬적으로 수행하며 이로인해 알고리즘 수행시간을 대폭 단축할 수 있다.

* 한국전자통신연구소 ATM정합연구실
論文番號 : 95329-0921
接受日字 : 1995年 9月 21日

Ⅰ. INTRODUCTION

ATM technology adopts statistical multiplexing to use network resource efficiently, but it requires complicated traffic management [1]. Traffic management is accomplished using basic functions such as Connection Admission Control(CAC), Usage Parameter Control(UPC) and Congestion Control(CC). CAC decides whether network accepts a connection set up request through the estimation of the network's Quality of Service(QoS) based on the user-declared traffic parameters.

The peak cell rate policing function is one of the UPC/NPC (Network Parameter Control) functions, therefore it is located at the network entry point. It monitors the compliance of each user to the negotiated peak cell rate between the user and the network throughout the connections holding time in order to prevent the network from QoS degradation due to excess traffic coming from malfunctioning terminals or malicious users. When the peak cell rate policing function detects a non-compliant cell, it takes a proper policing action i.e. cell tagging or discarding on the non-compliant cell. Cell tagging means that the peak cell rate policing function converts CLP=0 cell identified as non-compliant to the CLP=1 cell.

From now, we will make brief description of VSA [2][3]. VSA is a peak cell rate policing algorithm applicable to any ATM connection cell stream provided that its peak emission interval T and cell delay variation tolerance τ are declared in the traffic contract. Fig. 1 shows the flow chart of VSA. For every connection, Four parameters used to run VSA are stored in the Connection Context Memory(CCM) i.e. peak emission T, cell delay variation tolerance τ, theoretical arrival time TAT and expiry bit E -- they will be explained in section 2. When a cell of any connection arrives at the policing device at time t, VSA reads expiry bit E belonging to the connection to examine whether TAT value of the connection is expired or not -- TAT value of the first cell is always considered expired, which will be explained in section 2.

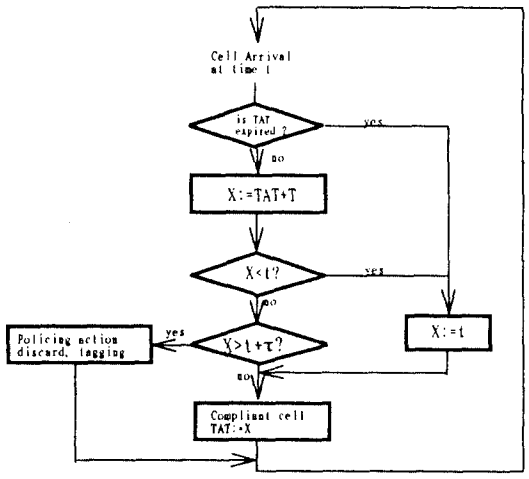


Fig. 1. Flow chart of VSA

If the E bit is true, that is, TAT value of the connection is expired, VSA considers the cell compliant. Otherwise, the cell should have arrived at time TAT+T. However, this is only a potential arrival time because the cell may possibly be declared as non-compliant. Therefore, VSA stores this value temporarily in a dummy variable X. When TAT value isn't expired, we compare X with cell arrival time t. After comparison, If X is smaller than t, the cell has experienced long queueing delay in the upstream or the user has sent the cell at the rate equal or less than the negotiated peak cell rate. Therefore, the cell is considered late and VSA declares the

cell compliant and TAT is set to cell arrival time t. If X is larger or equal to t, the cell is considered as an early arrived cell -- it belongs to cell clump, and probably it has experienced lower queueing delay than preceding one along the connection route. Therefore, VSA should decide whether the cell clump is due to cell delay difference between current cell and the preceding cell or due to violation of negotiated cell rate by the user. For that purpose, VSA compares X with the value of $t+\tau$ where τ is the cell delay variation tolerance value which VSA reads from the CCM for the connection. If X is smaller than $t+\tau$, VSA considers that the cell clump is due to cell delay difference. Hence, the cell is identified as compliant and TAT value is set to X. Otherwise, the cell clump is due to violation of the cell rate by the user. VSA therefore declares it non-compliant and takes a proper policing action. The value of TAT is left unchanged. There are two policing actions on non-compliant cell, that is, tagging and discarding. Tagging means conversion of non-compliant CLP=0 cell to compliant CLP=1 cell, whereas discarding is removal of all non-compliant cells from cell stream irrespective of their CLP bit values. Therefore, policing action on the non-compliant CLP=1 cell is always discard of it. It was proved that VSA can be mapped into the continuous-time Leaky bucket with capacity $T+\tau$ and leak rate 1/T [4].

Most of papers suggested various policing algorithms such as Moving Window, Jumping Window, Leaky Bucket and Exponentially Weighted Moving Average, and discussed their performances [5][6][7]. However, in this paper, we suggest the architecture for the design of a device implementing VSA and describe the device from the design point of view. We also discuss parameter coding size, a formula for Peak Cell Rate Granularity, how to realize expiry process that is essential to the implementation of VSA and how to select policing modes that are built in our device in order to realize the policing schemes recommended by ITU-T as well as other policing schemes. Section 2 presents some issues on the implementation of VSA. Section 3 describes the architecture for the implementation of VSA and suggestes simplified circuit for reducing the wasted time on performing the VSA to a great extent. Section 4 describes the experimental results obtained under four different peak cell rate values in order to confirm that the device performs the policing function satisfactorily. Finally, section 5 summarizes the discussion.

Ⅱ. SOME ISSUES ON IMPLEMENTATION

In this section, we describe coding size for four parameters mentioned in section 2, Peak Cell Rate Granularity, expiry process, and modes of operation built in our device.

2.1 Coding Size

For the implementation of VSA, we have to code time t used to measure cell arrival time, peak emission interval T, theoretical arrival time TAT and cell delay variation tolerance with the use of binary number. There is trade-off between coding accuracy and implementation complexity, so we have to make appropriate compromise between coding size and design complexity. It is necessary to normalize all parameter values by the cell slot time (Δ) -- 2.726 μs in the case of 155 Mbps link before we code each parameter value using binary number.

We use fixed point system when coding each

parameter to reduce design complexity. For the parameter TAT, we assigned 24 bits with 16 bit integer and 8 bit decimal part. For the parameter T, we assigned 22 bits with 14 bit integer and 8 bit decimal part. Twelve bits was used for coding τ which needs only integer part. Finally, for the time t, we assigned 24 bits (24 bit counter) in order to perform comparison of time t with potential cell arrival time X.

2.2 Peak Cell Rate Granularity

Since there is coding size limit when expressing the peak cell rate with binary number, it is impossible to express all possible values of the peak cell rate accurately. Instead, only a finite and discrete set of the peak cell rate values can be processed by an ATM system. Other values should be rounded up to the upper value closest to them. The set obtained by arranging them in ascending order is called ATM Peak Cell Rate Granularity by ITU-T I.371. As the peak cell rate is inverse of the peak emission interval T and it is coded using 22 bits with 14 bit integer and 8 bit decimal part, we can express any value Pc belonging to ATM Peak Cell Rate Granularity as following form [3]:

$$P_c = 10^6/2.726(k+m/256) \text{ [cells/s]}$$
$$\text{for } 0 \le k \le 2^{14}-1,\ 0 \le m \le 2^8-1,\ m+k \neq 0$$

From this formula, we know that the bit rate handled by our device ranges from 9.493 Kbps to 39.82 Gbps. From above equation, we can also formulate the relative variation between two consecutive values as following approximation:

$$\Delta P_c/P_c \simeq 1.0694 \times 10^{-8} P_c$$

This relationship tells us that the ATM Peak Cell Rate Granularity provided by our device is relatively accurate for the typical values used in the broadband network. For example, the two nearest values to 100 Mbps are 99.79 Mbps and 100.16 Mbps, and their relative variation is 0.371 %.

2.3 Expiry Process

When running VSA, we should record when the preceding cell of any connection arrived at the policer or when it should have arrived. The TAT parameter is intended for that purpose. By the way, this parameter value increases steadily along time until it reaches a maximum value decided by the coding size for TAT -- $2^{16}\Delta$ i.e. 178.65 ms, and then returns to 0. This procedure repeats itself infinitely as long as the connection holds. These returns to 0 make it difficult to simply compare TAT+T with other timing parameter such as t or t+τ. To overcome this problem, we run an expiry process on TAT value of every connection. All the TAT values are scanned cyclically by the expiry process and declared either 'valid' or 'expired' depending on a criterion built in the expiry process. Expiry bit E is associated with every TAT value to indicate whether it is 'valid' or 'expired'.

We declare every TAT value 'expired' upon connection admission. Therefore, only 'valid' TAT values are used when VSA performs comparisons.

There are many ways to implement the expiry process, but we followed the method in Reference [3]. Let us divide TAT value ranging from 0 to R = $2^{16}\Delta$ = 178.65ms into m intervals Ik, k = 0,1, ..., m-1, then I_k is expressed as follows:

$$I_k = [kR/m,\ (k+1)R/m]$$

The expiry process starts at the beginning

point of each interval Ik and stops after it scans all the TAT values in the CCM. CCM stores four parameters related to every connection that are required to run VSA. While it scans TAT values, it declares every TAT value subject to the interval $I_{(k+\lfloor m/2 \rfloor+1) \text{ Mod } m}$ 'expired'. In order to indicate such a TAT value 'expired', every E bit of the connection to which the TAT value belongs is set to 1.

According to the expiry process, the most recent TAT value that can be declared is Tmax = R/2 - 2R/m older than current time. Therefore, if the peak emission interval T of any connection is greater than T_{max}, a cell of the connection may find expired TAT upon arrival and it may be declared compliant even though it is not the compliant cell. Therefore, 1/Tmax is minimum peak cell rate that can be supported by a real policing device. We know that under the upper bound of R/2, the larger m is, the larger T_{max} is -- the smaller peak cell rate is., In our device, since R = $2^{16}\Delta$ = 178.65 ms and m = 8, T_{max} = R/4 = 44.663 ms, the smallest peak cell rate that can be policed by our device is $P_{c,\ min}$ = 22.39 cell/s = 9.493 Kbps.

Table 1. Mode for policing action

Mode	policing action on non-compliant cell		mlde bits (b_2 b_1 b_0)
I	pass		1 x x
II	CLP=0 cell	tag	0 1 0
	CLP=1 cell	pass	
III	CLP=0 cell	discard	0 1 1
	CLP=1 cell	pass	
IV	CLP=0 cell	tag	0 0 0
	CLP=1 cell	discard	
V	discard		0 0 1

2.4 Policing Mode

The device provides five kinds of policing modes that can be programmed by external microprocessor. Each connection selects its own policing mode by setting corresponding mode bits in the CCM to predefined values shown in Table 1. Table 1 summarizes the policing action corresponding to each policing mode. It should be noted that the mode II and III is intended to police only CLP=0 cell stream of a connection, whereas the mode IV and V are used to police CLP=0+1 cell stream.

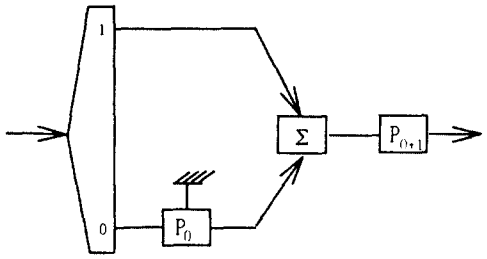
When a connection has the policing mode I, the device unconditionally passes all the cells of the connection without decision on violation. In the mode II, the device takes tagging action on the non-compliant CLP=0 cells, whereas passes CLP=1 cells unconditionally. In the mode III, the non-compliant CLP=0 cells are discarded, while CLP=1 cells are passed unconditionally. With policing mode IV, the device takes tagging action on the non-compliant CLP=0 cells and discards the non-compliant CLP=1 cells. In the mode V, all the non-compliant cells are discarded regardless of the CLP bit's value.

2.5 Realization of Policing Schemes

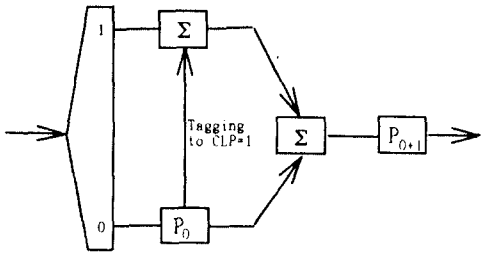
Fig. 2 shows two policing schemes recommended by ITU-T I.371 [2]. The no cell tagging scheme shown in Fig. 2 (a) discards CLP=0 cells identified as non-conforming by the policing function (P_0) performed on the CLP=0 cell stream, and the resultant CLP=0+1 stream consisting of compliant CLP=0 cells and user-submitted CLP=1 cells is tested by the policing function (P_{0+1}) and then the non-compliant CLP=0+1 cells are discarded. The cell tagging scheme in Fig. 2 (b) converts non-compliant CLP=0 cells to CLP=1 cells and merges those newly formed CLP=1 cells i.e.

tagged cells with user-submitted CLP=1 cells before CLP=0+1 cell stream consisting of compliant CLP=0 cells, user-submitted CLP=1 cells and tagged CLP=1 cells is offered to the conformance test (P_{0+1}). Each of above mentioned two policing schemes can be realized by cascading two devices as shown in Fig. 3. The device 1 is for the conformance test of CLP=0 cell stream and the device 2 is for that of CLP=0+1 cell stream. Every connection selects its own policing mode among five modes shown in Table 1 by setting its mode bits in each of the two cascaded devices to the predefined values shown in Table1.

When a connection employs the no cell tagging scheme in Fig. 2 (a), it can be realized by setting the mode bits of the connection in the device 1 to 011 (mode Ⅲ) and those bits in the device 2 to 001 (mode Ⅴ). If a connection selects the cell tagging scheme in Fig. 2 (b), it can be realized by setting the mode bits of the connection in the device 1 to 010 (mode Ⅱ) and those bits in the device 2 to 001 (mode Ⅴ). The TAT parameters in the policing function P_0 should be updated if and only if the CLP=0 cell is judged to be conforming by both P_0 and P_{0-1}. The control signal of update policing parameter in the Fig. 3 is used to indicate that the TAT of P0 can be updated since the CLP=0 cell identified as compliant by P0 is also judged to be conforming by P_{0+1}. Other possible schemes are about policing only the aggregate CLP=0+1 cell stream. The scheme that discards all the non-compliant cells regardless of the CLP bit values can be realized by employing mode I for the device 1 and adopting mode V for the device 2. The scheme that discards the non-



(a) No cell tagging



(b) Cell tagging

Fig. 2. Policing scheme

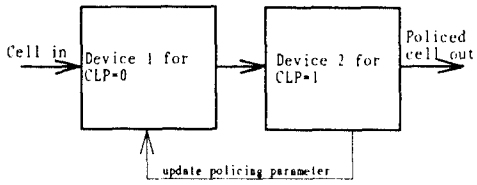


Fig. 3. Cascade of devices

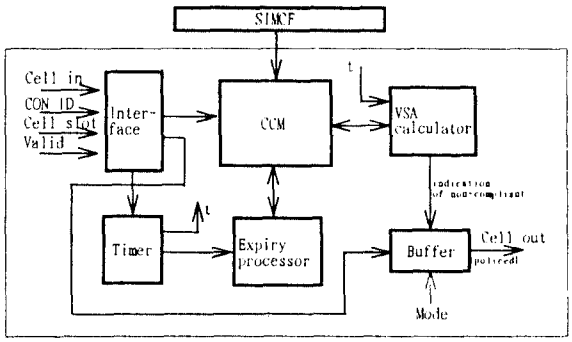


Fig. 4. Architecture of policing device

compliant CLP=1 cell but converts the non-compliant CLP=0 cell to the compliant CLP=1 cell can be realized by selecting mode Ⅰ for the device 1 and choosing mode Ⅳ for the device 2.

Ⅲ. ARCHITECTURE

Fig. 4 shows the architecture of the device to implement VSA. The brief description of each part is as follows.

· Interface

This part is responsible for reforming timing relationship among four signals such as Cell in, CON_ID, Cell slot and Valid, which are coming from the interface with the physical layer. Cell in is input cell data. CON_ID is used to identify a connection and used as the address to access four parameters in CCM i.e. TAT, T, τ and E pertaining to the connection. Cell slot is a signal representing the beginning of the cell time slot. Finally, Valid indicates whether a time slot contains an effective cell or not.

· Timer

Timer generates time t that is used to keep track of arrival times of every connections cells. It is 16 bit counter and increments by 1 whenever a new time slot begins. It increases steadily until it reaches 65535, the peak value and then returns to 0. This process repeats infinitely while the policing device is active.

· CCM

VSA requires four parameters(TAT, T, τ and E) in order to perform policing function. CCM(Connection Context Memory) stores the four parameters in the location assigned to each connection. Besides them, it also stores the mode of operation for each connection that has been explained in the section 2.4. CCM is accessed by VSA calculator performing the policing function actually whenever multiplexed cells from every connection arrive at the device. Table 2 is the configuration of the CCM.

Table 2. Configuration of CCM

CON_ID	parameter				
0	T_0	τ_0	E_0	$Mode_0$	TAT_0
1	T_1	τ_1	E_1	$Mode_1$	TAT_1
· · ·					
511	T_{511}	τ_{511}	E_{511}	$Mode_{511}$	TAT_{511}

At the connection set-up, the parameter value of T and τ is set by the external microprocessor. TAT is unceasingly updated during the connection holding time. E is set to 1 at the connection set-up time, and then reset to 0 whenever an arriving cell is identified as compliant. Again, it is set to 1 when 'expired' condition on TAT value occurs. The device can handle 512 connections simultaneously.

· Expiry processor

This part performs the expiry process depicted in section 2.3 using t from the Timer and TAT value from the CCM. It issues the address for reading every TAT and E value in the CCM. If the read TAT value is subject to such a region that makes it 'expired', E bit is set to 1 unless it has been already set to 1. If the read E bit has been already set to 1, the Expiry processor doesn't change its value regardless whether TAT value is 'expired' or not.

· VSA calculator

This part actually runs VSA by performing addition and comparison using TAT, T, τ and

t. Whenever a cell arrives, the parameter values are read from CCM in order to decide whether an incoming cell complies with the peak cell rate contracted between the user and the network at the connection set-up time. In the case that VSA calculator declares an arriving cell 'non-compliant', it sends to the Buffer a signal indicating non-compliant.

When we carefully look at Virtual Scheduling Algorithm in Fig. 1, we can find that among three decision parts, the decision part of 'is TAT expired ?' has the highest priority in deciding the final TAT value. The decision part of 'X< t ?' has the second higher priority and the decision part of 'X ≤ t+τ ?' has the third higher priority. In other words, If 'is TAT expired ?' is true, the incoming cell is declared compliant and VSA substitutes TAT value of the incoming cell with the current time t regardless of the results of other two parts. If 'is TAT expired ?' is false and 'X< t ?' is true, then the cell is identified as compliant and TAT value is set to the current time t irrespective of the result of 'X ≤ t+τ?'. If both results of 'is TAT expired ?' and 'X< t ?' are false and 'X ≤ t+τ ?' is true, then VSA also recognizes the cell as compliant and changes TAT value to TAT+T. When results of above three decision parts are all false, VSA declares the cell 'non-compliant' and leaves the TAT value unchanged. Therefore, based on this observation, we perform VSA by processing three decision parts in parallel instead of sequentially so that the time to complete the algorithm reduces to a great extent. To do this, we exploit a priority encoder as shown in Fig. 5 that represents the simplified VSA calculator.

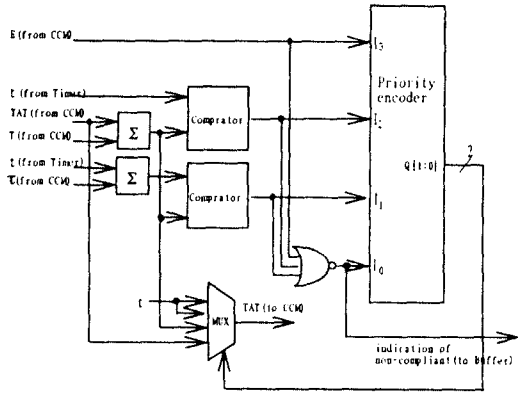


Fig. 5. Simplified VSA calculator

From the figure, we know that with the use of two adders and two comparators, the VSA calculator concurrently performs four arithmetic operations, that is, summation of TAT and T, summation of t and τ, comparison between t and TAT+T and comparison between t+τ and TAT+T. E bit indicating whether a connection is expired is connected to the I3, the input of the priority encoder with the highest priority since it has the highest priority in deciding the final TAT value. The outputs from above two comparators are directed to the input I2 and I1 that have the second and the third higher priority since they have the second and the third higher priority in deciding the final TAT value respectively. Finally, the output of NOR 3 gate indicating that the arriving cell of the connection is non-compliant is transferred to the I0, the input with the lowest priority since it has the lowest priority in deciding the final TAT value. The resultant output from the priority encoder is used to control the four input mux in order to produce the final TAT value by selecting one from the three candidates of t, TAT and TAT+T. The final TAT value is used by the

next arriving cell.

· Buffer

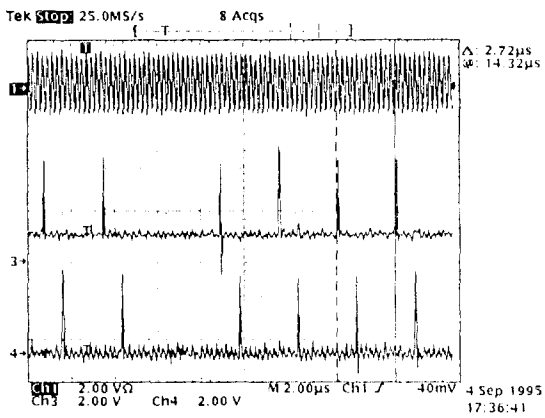
It takes about 550 ns for our device to decide whether an arriving cell is compliant or not. We therefore need a storage element to store incoming cell data for that duration. Buffer is intended for that purpose. In addition, it takes policing action on incoming cell according to the policing mode of individual connection.

Since we use VSA as a 'pick up' algorithm, our device can not prevent clusters of cells from entering the network in the presence of cell delay variation [8]. In order to avoid such clusters, we have to implement additional spacing function in our device.

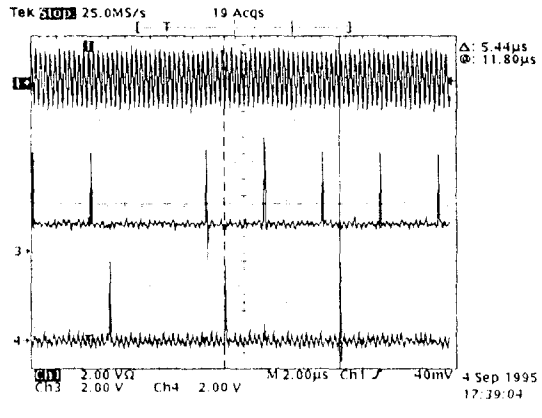
Ⅳ. EXPERIMENTAL RESULTS

Fig. 6 shows the snapshot of the master clock, the cell sync input and the cell sync output signal of a connection that are observed by the oscilloscope. The channel 1, 3 and 4 in the figure represent the master clock, the cell sync input and the cell sync output signal respectively. The master clock is referenced by all the signals in the device in order to be synchronized with it. The cell sync input signal represents the starting time of the arriving cell at the device. The cell sync output signal indicates the starting time of the policed cell by the device under the mode V.

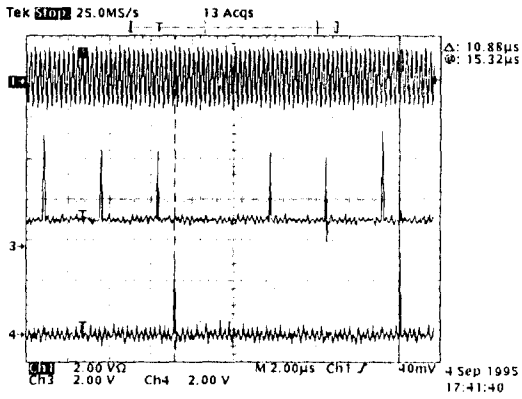
The Fig. 6 (a), (b), (c) and (d) show the waveforms at the instant the input has its peak cell rate. They represent the measured waveforms for the four cases each negotiated peak cell rate of which is 155, 155/2, 155/4 and 155/8 Mbps. For all the cases, the input of the connection is a VBR (Variable Bit Rate) traffic with the peak cell rate of 155 Mbps, the average cell rate of 70 Mbps and the CDV of 5.44 μs. The Fig. 6 (a) shows that the device passes all the incoming cells without any dropping of them since the user sends the cells at the same rate as the negotiated peak cell rate of 155 Mbps. From the Fig. 6 (b), we see that the device passes the half of the incoming cells since the user emits the cells at twice as the negotiated peak cell rate of 155/2 Mbps, which coincides with the expected result. The Fig. 6 (c) and (d) show



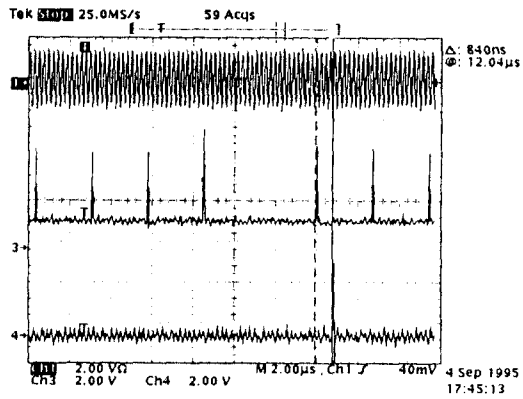
(a)



(b)



(c)



(d)

Fig. 6. Snapshot of Waveforms

that the device passes a quarter ((c)) and one eighths ((d)) of the incoming cells since the negotiated peak cell rate in each figure is 155/4 Mbps and 155/8 Mbps.

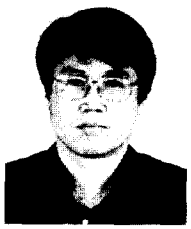
V. CONCLUSIONS

In this paper, we discussed the coding size for four parameters i.e. TAT, T, τ and E that are used by VSA and showed that the ATM Peak Cell Rate Granularity supported by the device is relatively accurate. We also described the architecture of the device for the implementation of Virtual Scheduling Algorithm and discussed some issues on the implementation. The smallest peak cell rate that the device can support was 9.493 Kbps. We presented how to choose the modes of two devices used to realize the two policing schemes recommended by ITU-T as well as other policing schemes. We exploited the priority encoder to run the algorithm in parallel instead of sequentially, which reduced the operation time to a great extent. We showed the experimental results under four different peak cell rate values to confirm that the device performs the policing function satisfactorily. We used CMOS gate array and standard cell technology for the fabrication of the device and consumed about 80,000 gates, including memory.

REFERENCES

1. A. Baiocchi, et al, "Loss Performance Analysis of an ATM Multiplexer Loaded with High-Speed ON_OFF Sources", IEEE J. Sel. Areas in Commun. vol. 9, no. 3, pp.388-393, Apr. 1991.
2. ITU-T SG XVIII, Recommendation I. 371, Geneva, March 1994.
3. Pierre E. BOYER, Michel J. SERVAL, Fabrice P. GUILLEMIN, "The SPACER-CONTROLLER : An Efficient UPC/NPC for ATM Networks", Paper A9.3 in Proceedings of ISS '92, Yokohama, Japan, October 1992.
4. "Mapping Non-conforming Cells with Losses in Peak Cell Rate Control Mechanisms", CCITT SG XVIII/8 Delayed contribution D. 209,

Geneva, June 1992.
5. E.P. Rathgeb, "Modeling and performance Comparison of Policing Mechanisms for ATM Networks", IEEE J. on Sel. Areas in Commun. vol. 9, No. 3, pp.325-334, Apr. 1991.
6. L. Dittman, S.B. Jacobsen, K. Moth, "Flow Enforcement Algorithms for ATM Networks", IEEE J. on Sel. Areas in Commun. vol. 9, No. 3, pp.343-350, Apr. 1991.
7. H. Hemmer, P.T. Huth, "Evaluation of Policing Functions in ATM Networks", Queueing, Performance and Control in ATM, ITC-13, pp.111-116, Copenhagen, June 1991.
8. Eugen Wallmeier and Tom Worster, "THE SPACING POLICER, AN ALGORITHM FOR EFFICIENT PEAK BIT RATE CONTROL IN ATM NETWORKS", Paper A 5.5 in proceeding of ISS '92, Yokohama, Japan, October 1992.



田 萬 泳(Man-Yeong Jeon) 정회원

1987년 2월 : 경북대학교 전자공학과(학사)
1991년 2월 : 경북대학교 대학원 전자공학과(석사)
현재 : 한국전자통신연구소 ATM 정합연구실 선임연구원

朴 弘 植(Hong-Shik Park) 정회원

1953년 8월 16일생
1977년 2월 : 서울대학교 공과대학 졸업(학사)
1986년 8월 : 한국과학기술원 전기 및 전자공학과 졸업(석사)
1995년 2월 : 한국과학기술원 전기 및 전자공학과 졸업(박사)
1977년 12월~현재 : 한국전자통신연구소 근무 현 ATM 정합연구실장