Design and Implementation of Two Rate Three Color Marker Based on FPGA

YanLong Zhang, YongTing Hu, Tao Li, ZhiGang Sun
School of Computer, National University of Defense Technology, Hunan 410073, Changsha, China
zhangyanlong126@163.com,807052086@qq.com,taoli.nudt@gmail.com,sunzhigang@nudt.edu.cn

Abstract - As a network QoS technique, two rate three color marker (trTCM) has been widely used in the implementation of traffic policing, traffic shaping and port rate limiting. The implementation of the trTCM is usually based on software, which is simple and easy to be configured. However, the software trTCM has low marking precision and it cannot be easily scaled to support high-speed network traffic. We propose a novel method to design and implement the trTCM based on FPGA. The key configuration parameters of the trTCM are optimized based on the thoroughly theoretical analysis. The experimental results show that the average marking precision of the designed trTCM is as high as 99.9734% with acceptable fewer hardware resources.

Index Terms - Two rate three color marker; QoS; FPGA

1. Introduction

With the rapid development of the Internet and computer technology, and constant diversity of multimedia services on the network, the best-effort service has become increasingly unable to meet the requirements of real-time business on the quality of service (QoS) nowadays. In the network, providing QoS control according to user needs has become the new challenges faced by the network service [1]. In this case, IETF RFC recommended standardized single-rate three color marker (srTCM) and two-rate three color marker (trTCM) two token bucket algorithms. These two algorithms can be widely applied to limit the access rate, general traffic shaping and physical interfaces rate limiting [2]. The srTCMs are a single-bucket or double-bucket structure [3], it is relatively simple in token added methods and packet processing procedure; The trTCMs double-bucket structure [4], it is relatively complex in add token and processes packets. The srTCM focuses on the size of packets burst, while the trTCM focuses on the burst rate, both have their advantages [5]. In practical applications, appropriate method for different services should be applied.

Since trTCM is usually implemented based on the software that is deficient in precision and performance. The main contribution of this paper is trTCM implemented by FPGA. On the basis of the theoretical analysis, by optimizing the configuration parameters, we have designed and implemented this trTCM with high precision, which occupies fewer hardware resources.

2. Two Rate Three Color Marker design implementation

A. Parameter Description

According to the IETF RFC for trTCM regulations, the trTCM has four parameters need to be set, including: CIR, CBS, PIR and PBS. In order to make the hardware easier to control marker, we define the following parameters:

1) Time granularity (Time_GRA): describes intervals added to the token.

2) Token granularity (Token_GRA): describes the number of bytes sent by a single token.

3) P bucket token increase granularity (P_Bucket_Add_GRA): describes the number of tokens want to add to P bucket in a single time granularity cycle.

4) C bucket token increase granularity (C_Bucket_Add_GRA): describes the number of tokens added to C bucket in a single time granularity cycle.

5) P bucket barrel depth (P_Bucket_Size): describes P bucket can accommodate the number of tokens.

6) C bucket barrel depth (C_Bucket_Size): describes C bucket accommodated the number of tokens.

According to the parameters defined above, we can have the following result:

\[ CIR = \text{Token_GRA} \times C_{\text{Bucket Add GRA}} / \text{Time GRA} \]

\[ PIR = \text{Token GRA} \times P_{\text{Bucket Add GRA}} / \text{Time GRA} \]

\[ CBS = C_{\text{Bucket Size}} \times PBS = P_{\text{Bucket Size}} \]

In this paper, time granularity, P bucket token increase granularity, C bucket token increase granularity, bucket barrel depth and C bucket barrel depth support external configuration via software to implement the flexible control.

For the purpose of simplification, we set the token granularity to be 1 that means a single token can send a byte packet data.

B. Theoretical Analysis

In order to verify the function and performance of the proposed trTCM, the trTCM has been implemented in self-developed network processing engine. The operating frequency of Network Processing Engine is 250 MHz (clock cycle 4 ns). We specify the range of token bucket rate-limit is from 1Mbps to 40Gbps when designed.

(1) Time granularity values. According to the formula:

\[ CIR = \text{Token_GRA} \times C_{\text{Bucket Add GRA}} / \text{Time GRA} \]

\[ PIR = \text{Token GRA} \times P_{\text{Bucket Add GRA}} / \text{Time GRA} \]

We have:

\[ \text{Time GRA} = \text{Token GRA} \times P_{\text{Bucket Add GRA}} / \text{PIR} \]

Or \[ \text{Time GRA} = \text{Token GRA} \times C_{\text{Bucket Add GRA}} / \text{CIR} \]

In order to make hardware reserved Time GRA register to meet the maximum parameter needs, we should take the maximum of P_Bucket_Add_GRA and C_Bucket_Add_GRA to get Time GRA maximum, while the maximum of
P_Bucket_Add_GRA and C_Bucket_Add_GRA are not sure, so we design a reference time granularity to determine TimeGRA value, which is the value when both of P_Bucket_Add_GRA and C_Bucket_Add_GRA have a value of 1. Thus, the maximum reference time granularity should satisfy:

\[
\text{Time}_{\text{GRA}} = \frac{\text{Token}_{\text{GRA}} \times P\_\text{Bucket\_Add\_GRA}}{\text{PIR}}
\]
or

\[
\text{Time}_{\text{GRA}} = \frac{\text{Token}_{\text{GRA}} \times C\_\text{Bucket\_Add\_GRA}}{\text{PIR}}
\]

Then, we have:

\[8 \times \frac{1}{2^{17}} = 2^{-17}\]

Converted to the hardware clock cycle: \(2^{17}/4 \times 10^{9} \approx 1908\), so the maximum reference time granularity can take an 11-bit wide register.

(2) Token increase granularity values. According to the formula:

\[
\text{CIR} = \frac{\text{Token}_{\text{GRA}} \times C\_\text{Bucket\_Add\_GRA}}{\text{Time}_{\text{GRA}}};
\]

\[
\text{PIR} = \frac{\text{Token}_{\text{GRA}} \times P\_\text{Bucket\_Add\_GRA}}{\text{Time}_{\text{GRA}}}
\]

We have:

\[
C\_\text{Bucket\_Add\_GRA} = \text{CIR} \times \frac{\text{Time}_{\text{GRA}}}{\text{Token}_{\text{GRA}}}
\]

and

\[
P\_\text{Bucket\_Add\_GRA} = \text{PIR} \times \frac{\text{Time}_{\text{GRA}}}{\text{Token}_{\text{GRA}}}
\]

In order to make the hardware reserved C_Bucket_Add_GRA, P_Bucket_Add_GRA register meet the parameters requirements, we set the maximum value of PIR and CIR to 40Gbps, the maximum value of TimeGRA is 1908, which is the maximum reference time granularity.

Formula:

\[
C\_\text{Bucket\_Add\_GRA}_{\text{max}} = 40 \times 2^{20} \times 4 \times 10^{-9} \frac{1908}{8} \approx 40974;
\]

\[
P\_\text{Bucket\_Add\_GRA}_{\text{max}} = 40 \times 2^{20} \times 4 \times 10^{-9} \frac{1908}{8} \approx 40974.
\]

Therefore, the C bucket token increase granularity and P bucket token increase granularity can take a 16-bit wide register.

(3) Bucket depth values. According to the RFC specification, we summed up the bucket depth value which should satisfy 1) greater than the maximum packet length, 2) more than the value of token increased granularity, 3) less than or equal to the maximum rate limit. The maximum of the bucket depth (Bucket_Size) should meet the condition 

\[
\text{Bucket\_Size} \leq 40\text{Gbps}/8=5\text{Gbps}.
\]

Therefore, barrel depth register can take a 33-bit wide register.

C. Design and Implementation

Function module of trTCM consists of the clock module, token management module, and metering module. The modular structure of implementation is shown in Figure 1. The signal definition list is shown in Table 1. It is based on the size of the packet (Packet_Size), P bucket and C bucket remaining number of tokens (P_Bucket_Sum, C_Bucket_Sum) to color. It is according to barrel depth, P bucket token increase granularity,C bucket token increases granularity and result of metering to update the number of tokens in bucket.

<table>
<thead>
<tr>
<th>Signal name</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_Bucket_Size</td>
<td>33</td>
<td>P bucket barrel depth</td>
</tr>
<tr>
<td>C_Bucket_Size</td>
<td>33</td>
<td>C bucket barrel depth</td>
</tr>
<tr>
<td>P_Bucket_Add_GRA</td>
<td>16</td>
<td>P bucket token increase granularity</td>
</tr>
<tr>
<td>C_Bucket_Add_GRA</td>
<td>16</td>
<td>C bucket token increase granularity</td>
</tr>
<tr>
<td>Time_GRA</td>
<td>11</td>
<td>Time granularity</td>
</tr>
<tr>
<td>Start_T</td>
<td>1</td>
<td>Start timing signal, 1:valid 0: Invalid</td>
</tr>
<tr>
<td>TokenAdd</td>
<td>1</td>
<td>Add token signal 1:valid 0: Invalid</td>
</tr>
<tr>
<td>P_Bucket_Sum</td>
<td>33</td>
<td>P bucket remaining number of tokens</td>
</tr>
<tr>
<td>C_Bucket_Sum</td>
<td>33</td>
<td>C bucket remaining number of tokens</td>
</tr>
<tr>
<td>P_BucketSub</td>
<td>1</td>
<td>C bucket subtraction signal 1:valid 0: Invalid</td>
</tr>
<tr>
<td>P_BucketSub</td>
<td>1</td>
<td>P bucket subtraction signal 1:valid 0: Invalid</td>
</tr>
<tr>
<td>Sub_Size</td>
<td>11</td>
<td>The size of token deletion</td>
</tr>
<tr>
<td>Packet_Size</td>
<td>16</td>
<td>The size of the packets</td>
</tr>
<tr>
<td>Color_Info</td>
<td>2</td>
<td>Color information: 00 red, 01 yellow, 10 green, 11 default</td>
</tr>
</tbody>
</table>

Clock module is responsible for timing. After external logic initial token parameters such as time granularity, token increase granularity and barrels depth, the start timing signal (Start_T) is set valid, and the clock counter starts from 0. When the counter reaches a time granularity period (arrived at TimeGRA), token-add signal (TokenAdd) is to be outputted. It notifies the token management module to add the token to bucket; counter is set back to 0 again.

Token management module is responsible for control and management of the token by listening to TokenAdd signal. This signal is sent by the clock module, let P bucket and C bucket add tokens. The amount of added tokens is the token increase granularity. By listening to meteringmodule to judge signal issued by C_BucketSub or P_BucketSub, let P bucket or C bucket delete tokens .The amount of deletion is equal to the value of Sub_Size signal. Its state machine ofimplementation is shown in Figure 2. State machine describes the C bucket managing state. Since the state machine of the P bucket is the same as C bucket, so there is no need to describe it. As is shown in Figure 2, the state machine consists of the initialization (C_initial), idle (C_idle), token-add (C_add) and token-subtract (C_sub). In order to avoid signal token-add and token-subtract conflict, we take precedence of token-subtract operation, which postpones the processing token-add operation. The state operation is specified as follows:
Figure 2 C bucket management state machine

C_initial: According to the external logic configuration, token increase granularity and barrel depth register are initialized. Then, Start_T signal is set valid, the state jumps to C_idle state; otherwise, the state remains in C_initial state.

C_idle: Listen to whether token-add or token-subtract signal is valid, if only the token-add signal is valid, state jumps to C_add state; If only the token-subtract signal is valid, the state jumps to C_sub state; If the token-add and token-subtract signals are both valid at the same time, keep with token-add signal current values, the state jumps to C_sub state.

C_add: Judge whether the sum of C bucket current number of tokens coupled and token increase granularity is greater than the bucket depth. If greater, set the current number of tokens equals to the barrel depth. Otherwise, the current number of tokens is equal to the sum of the number of remained tokens and token increase granularity. Then listen to whether token-subtract signal is valid. If so, the state jumps to C_sub state; If not, the state jumps to C_idle state.

C_sub: the current number of tokens minus the length of the packet, the state jumps back to C_idle state.

Metering module is based on the current number of tokens in P bucket, the current number of tokens in C bucket and packet length information to mark different packets with different colors. Metering module notifies token management module by C_BucketSub or P_BucketSub signal. Deletion of size of P bucket or C bucket is determined by the Sub_Size signal. (Sub_Size is equal to packet length). The specific code of two rate three color marker algorithm is shown in Algorithm 1.

Algorithm 1: Two rate three color marker

Input: C_Bucket_Sum, P_Bucket_Sum;
Input: Packet_Size;
Output:C_BucketSub, P_BucketSub, Sub_Size;
Output:Color_info;
Begin
If(P_Bucket_Sum<Packet_Size)
    Color_info<=red;
Else if(C_Bucket_Sum<Packet_Size)
    Begin
        Color_info<=yellow;
P_BucketSub<=1;
Sub_Size<= Packet_Size;
    End
Else
    Begin
        Color_info<=green;
P_BucketSub<=1;
C_BucketSub<=1;
Sub_Size<= Packet_Size;
    End
End
End

When the arrival packet length is greater than P bucket current number of tokens(P_Bucket_Sum<Packet_Size), packet is marked red; when the packet size is less than or equal to the current number of tokens in the bucket of the P and C, packet is marked yellow, P bucket-subtract signal is set valid, the size of token decrease is equal to the size of packet; when the packet size is less than the current number of tokens in the bucket C, packet is marked green, both P bucket-subtract and C bucket-subtract signal are set valid, the size of token decrease of both bucket is equal to the size of packet.

3. Experimental Analysis

In order to verify the function integrity and coloring effect of the trTCM in the real flow measurement, we use Verilog HDL to compile trTCM algorithm. The trTCM is implemented based on Altera FPGAs[6] (Stratix IV EP4SGX180KF40C2). We analyze the implementation code and occupied logical resources by using Quartus II hardware programming tools. We use the IXIA XM2 tester to analyze the mark error of trTCM in a real environment.

<table>
<thead>
<tr>
<th>Resource name</th>
<th>using total quantity</th>
<th>occupies proportion /%</th>
<th>Resources total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinational LUTs</td>
<td>76</td>
<td>0.054%</td>
<td>140600</td>
</tr>
<tr>
<td>Logic registers</td>
<td>96</td>
<td>0.068%</td>
<td>140600</td>
</tr>
<tr>
<td>Total block memory bits</td>
<td>2048</td>
<td>0.017%</td>
<td>11704320</td>
</tr>
</tbody>
</table>

Table 2 analyzes the resource consumption of token management module and metering module. In table 2, the token management module and metering module occupy less than 0.07% of the total storage and logic resource. Therefore, the realization of the two rate three color marker has the characteristic of utilizing fewer resources.

Figure 3 shows the error of marker marked in the case that P bucket and C bucket barrel depth is set to 40 k and 10 k respectively. As depicted in figure 3, the average committed burst rate error is only 0.0322%, the average peak rate error is
only 0.021%, the average error of both committed rate error
and peak rate error is only 0.0266%, so the average mark
accuracy is as high as 99.9734%. Therefore, it is a marker with
higher accuracy.

Experimental results summary: it verified that the trTCM
can effectively achieve color-coded and require fewer
resources when implemented on FPGA. Performance test
verified the mark error of trTCM. The experiment proves that
the trTCM has good color marked effect, and average mark
accuracy is as high as 99.9734%, the storage and logic
resources cost is less than 0.07%.

4. Conclusion

This paper proposed to use FPGA to realize trTCM.
Based on the theoretical analysis, the optimized configuration
parameters and value ranges of the trTCM are determined. The
logic structure and state-machine of trTCM are designed based
on the parameters, and it is further implemented based on
FPGA. The experimental results show that the implementation
of the trTCM is feasible and with high accuracy. The trTCM
implementation technique can be well applied to network
processing equipment such as switches and routers.

References
1. Xiaotong Zhang, Peiya Li, Qin Wang, Chong, A Multi-service Token
Bucket Algorithm for MAC QoS System in HFC access network, 2008
IEEE 8th International Conference on Computer and Information
2. Xiao-li Li, Yu-chun Guo, Comparison Between Token Bucket Algorithms
in QoS Technology. ZTE COMMUNICATIONS, Vol. 13(3), pp. 56-60,
June 2007 (In Chinese).
3. Heinanen J, Guerin R.A single rate three color marker. RFC 2697,
September 1999.
5. M. Feng, H. Zhang, D. Wang and Z. Sun, Design and implementation of
high-speed token bucket based on FPGA, 2nd International Conference
Chinese).