STATIC-THRESHOLD-LIMITED BuST PROTOCOL

Constance Kalu
Department of Electrical/Electronic and Computer Engineering
University of Uyo, Akwa Ibom
NIGERIA

Simeon Ozuomba
Department of Electrical/Electronic and Computer Engineering
University of Uyo, Akwa Ibom
NIGERIA

Umoren Mfonobong Anthony
Department of Electrical/Electronic and Computer Engineering
University of Uyo, AkwaIbom
NIGERIA

ABSTRACT

In this paper, Static-Threshold-Limited BuST (STLB) Media Access Control (MAC) protocol was developed for bandwidth allocation in Multiservice Local Area Network (MLANs). STLB protocol was developed from two existing versions of the timed token MAC protocols, namely; Static-Threshold-Limited On-Demand Guaranteed Service Timed Token (STLODGSTT) protocol and Budget Sharing Token (BuST) Protocol. The development and analysis of the STLB protocol are presented for a system that is heavily loaded with asynchronous traffic but with variable load of synchronous traffic. In all, STLB protocol improved on the ability of the STLODGSTT protocol to utilize available network bandwidth by improving on the protocol's spare bandwidth reclaiming mechanism. Also, a numeric example is used to demonstrate the improved performance of the STLB protocol over the existing STLODGSTT protocol.

Keywords: Multiservice, bandwidth, protocol, real-time, non real-time , network, traffic.

INTRODUCTION

Nowadays, efficient support for both time-critical and non real-time traffic in the same Local Area Network (LAN) is essential (Ricardo , 2010 and White , 1997). The MAC protocol for such multiservice LAN must provide not only bounded message transmission time, as required by the hard and soft real-time tasks, but also high throughput, as demanded by non real-time tasks (Indumathi and Murugesan , 2010; Zhang and Burns , 1994 a; Zhang and Burns , 1994 b ; and Regnier, and Lima, 2006). An attractive MAC approach for such networks is the timed token protocol. Consequently, various versions of the timed token protocol have been incorporated into several high-bandwidth network standards (Nicholas and Wei , 1994), such as, IEEE802.4 Token Bus LAN (IEEE ,1995); Fiber Distributed Data Interface (FDDI) (Biao and Wei , 1992; Shin . and Zheng , 1995; Kenneth and Marjory , 1987; Grow , 1982 and Chan , Chen ., Cao and Lee , 1992); SAFENET (Dept. of Defense US ,1992) ; Manufacturing Automation Protocol (MAP) (Mcguffin , Reid , and Sparks ,1998) ; High-Speed Ring Bus (Uhlhorn , 1991), in PROFIBUS (Tovar , and Vasques ,1991); and in wireless networks (Lee, Attias , Puri, Sengupta, Tripakis, and Varaiya ,2001; Malpani , Vaidya , and Welch , 2001, and Willig ,2002).

Recently, the Static-Threshold-Limited On-Demand Guaranteed Service Timed Token (STLODGSTT) protocol (Ozuomba , Chukwudebeb , Obot , 2011) and Budget Sharing Token (BuST) Protocol are developed to improve on the ability of timed token protocol to support diverse traffic classes on the same LAN (Franchino , Buttazzo , and Facchinetti , 2007). The two protocols adopted different spare bandwidth reclaiming mechanisms. Though different, the two mechanisms can be combined to give higher throughput in certain network and traffic situations than any of the two protocols operating separately. The new MAC protocol called Static-Threshold-Limited BuST Protocol (STLB) protocol is analyzed and compared with STLODGSTT protocol. The comparison is with respect to the ability of each of the protocol to support diverse traffic classes in the same LAN under varying network configurations.

The rest of the paper is organized as follows. The network and message models are presented in Section 2 along with brief review of the STLODGSTT and BuST protocols (Ozuomba , Chukwudebeb , Obot , 2011 and (Franchino , Buttazzo , and Facchinetti , 2007). The STLB protocol is presented and analyzed in Section 3. In section 4, the STLB protocol is compared with the STLODGSTT protocol. Finally, concluding remarks and recommendations for further studies are presented in Section 5.

The Timely-Token Protocol and its Parameters

Network Model

The network model, as presented in (Ozuomba, Chukwudebeb, Obot, 2011) consists of a token ring network with N nodes as shown in Fig 1. Each node has a unique number in the range 0, 1, 2…N-1. For each node i, the next node along the unidirectional medium is station (i+1) or more appropriately node (i+1) ***mod N***. The token frame circulates around the ring from node i to nodes i + 1, i + 2, … until node i + (N-1), then to nodes i , i + 1, i+2,…, helping to determine which node should send a frame of message among the contending nodes. A special bit pattern called token frame circulates around the ring from node i to nodes i + 1, i + 2, … until node i + (N-1), then to nodes i , i + 1, i+2,…, helping to determine which node should send a frame of message among the contending nodes. Let w_i denote the latency or walk-time between a node, i and its upstream neighbor node, (i + 1). The sum of all such latencies in the ring is known as the ring latency or the token walk-time,

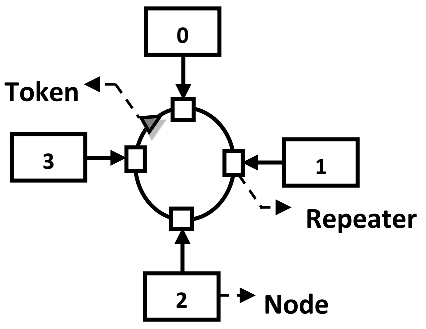


Fig1 A 4-Station Token Ring Network

W, where W =

$$\left(\sum_{i=0}^{i=N-1}(w_i)\right) \qquad (1)$$

The ring latency, W denotes the token walk time around the ring when none of the nodes in the network disturb it (Biao, and Wei, 1992 and Ozuomba, and Chukwudebe , 2003).

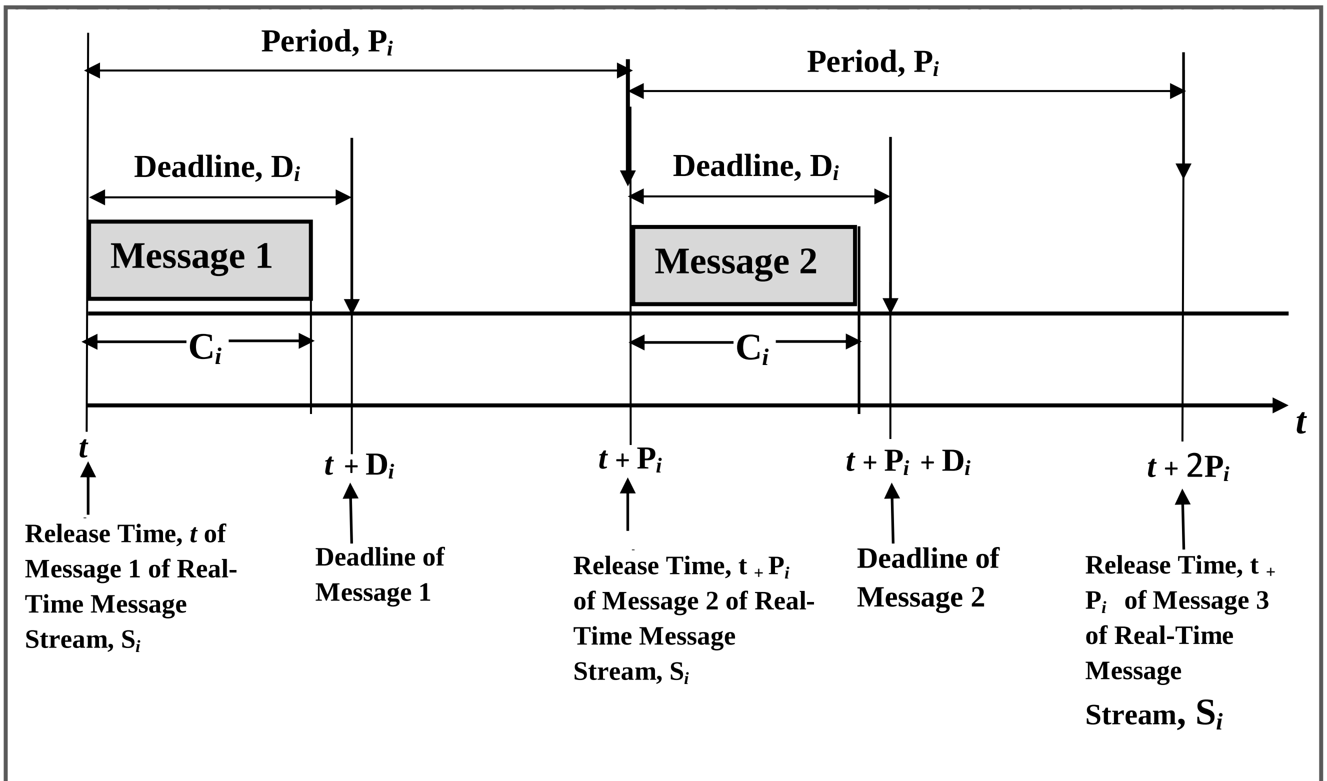
Message Model

Messages generated in the system at run time may be classified as either synchronous (real-time) messages or asynchronous (non real-time) messages (Ozuomba, Chukwudebeb, Obot, 2011). Furthermore, each node, i in the ring has a single stream of synchronous messages, $\mathbf{s}_i$, where $\mathbf{s}_i$ is defined in terms of the tuple ; $\mathbf{s}_i = \{ C_i , P_i , D_i\}$, where:

- *Message Length*, $\mathbf{C}_i$, is the maximum amount of time required to transmit a stream message. This includes the time required to transmit both the payload data and message headers.
- *Period Length*, $\mathbf{P}_i$, is the minimum inter-arrival period between consecutive messages in stream, s_i at node i. If the first message of node i is put in the transmission queue at time $t_{i,1}$, then the j-th message in stream s_i will arrive at time $t_{i,j} = t_{i,1} + (j - 1) P_i$,

where j >1. For instance, if the first message arrives at time *t*, then the second message will arrive at $t + P_i$ and the third message will arrive at $t + 2P_i$ as shown in Fig 2.

- *Message Deadline*, $\boldsymbol{D_i}$, is the relative deadline associated with messages in stream $\mathbf{s}_i$, that is, the maximum amount of time that can elapse between a message arrival and the completion of its transmission. Thus, the transmission of the j-th message in stream $\mathbf{s}_i$ that arrives at $t_{i,j}$ must be completed no later than $t_{i,j} + D_i$, which is the message's absolute deadline. Again, as an example, if the first message in the message stream, $\mathbf{s}_i$ arrives at time *t*, then it must be transmitted not later than $t + D_i$, as shown in Fig 2.


Period, P_i
Period, P_i
Deadline, D_i
Deadline, D_i
Message 1
Message 2
C_i
C_i
t
t
$t + D_i$
$t + P_i$
$t + P_i + D_i$
$t + 2P_i$
Release Time, *t* of Message 1 of Real-Time Message Stream, S_i
Deadline of Message 1
Release Time, $t + P_i$ of Message 2 of Real-Time Message Stream, S_i
Deadline of Message 2
Release Time, $t + P_i$ of Message 3 of Real-Time Message Stream, S_i

Note: $\mathbf{C}_i$ is the *Message Length*

Fig 2: Model for the Synchronous (or Real-Time) Message Stream, S_i in Node I (Ozuomba , Chukwudebeb, Obot, 2011).

The Timed-Token Protocol Parameters

The parameters as presented in (Ozuomba , Chukwudebeb, Obot, 2011) includes the following:

a) **Target Token Rotation Time, (TTRT):** TTRT is the time needed by the token to complete an entire round-trip of the network. The value of TTRT, denoted as τ.

b) **Synchronous Capacity of Node i (H_i):**
$\mathbf{H}_i$ represents the maximum time for which a station, i is allowed to transmit synchronous messages during every token receipt. Then, according to (Ozuomba , Chukwudebeb, Obot, 2011);

$$\mathbf{H_0 + H_1 + \ldots H_{N-1}} = \left(\sum_{i=0}^{i=N-1} (H_i) \right) = \mathbf{H} \qquad (2)$$

$$\mathbf{H_i + w_i = \sigma_i} \quad (3)$$

$$H + W = T \quad (4)$$

$$A = \tau - T \quad (5)$$

where ***A*** be defined as the total time units available to the asynchronous traffic in every cycle .

Constraints:

For proper operation of the timed-token protocol, the choice of values for τ and T parameters must satisfy the ***Protocol Constraint*** and the ***Deadline Constraint*** which according to (Ozuomba , Chukwudebeb, Obot, 2011) is given as,

$$\mathbf{T} \le \tau \le \min_{i=0,1,..N-1} (D_i) \quad (6)$$

c) **Token Rotation Timer of Node i (TRT*ᵢ*):**

TRT$_i$ is the cycle length or the time between two consecutive token receipts at node i.

d) **The Unused Synchronous Bandwidth (ε)**

According to (Ozuomba , Chukwudebeb, Obot, 2011),

$$h_i = H_i - \varepsilon_i \quad (7)$$

where h_i denotes the used portion of $\mathbf{H}_i$ and ε_i the unused portion of $\mathbf{H}_i$ time units reserved for the synchronous traffic in node i (where $\mathbf{h_i} \le \mathbf{H_i}$). Then, for a system that is lightly loaded with synchronous traffic, out of the $\mathbf{H_i}$ time units, only h_i time units are used in node i leaving ε_i time units unused

$$\varepsilon_i + \varepsilon_i + \ldots + \varepsilon_i = \varepsilon \quad (8)$$

e) **An Asynchronous-Limit Variable of Node i (THT$_i$):**

THT$_i$ is used to control the amount of time for which node i can transmit asynchronous messages.

2.4 Review of BuST Protocol and STLODGSTT Protocol

Generally, in the timed token protocols, there are two categories of spare bandwidth which can be reallocated to the asynchronous traffic. The two categories of spare bandwidth are;

i. The unallocated bandwidth per cycle, given as $A = \tau - T$ (which is the same as , $A = \tau - H - W$)

ii. The allocated but unused bandwidth, given $\varepsilon = \varepsilon_i + \varepsilon_i + \ldots + \varepsilon_{i.}$. ε is the portion of the bandwidth allocated to the synchronous traffic but is not used by the synchronous traffic due to drop in the expected load level of the synchronous traffic.

BuST and STLODGSTT protocols differ on how each protocol allows the nodes to exploit these spare bandwidth to transmit asynchronous traffic on each token receipt.

In the BuST (Franchino , Buttazzo, and Facchinetti, 2007), transmission of asynchronous traffic is done with the ε spare bandwidth unused by real-time (synchronous) traffic. In essence, the H_i timed units of a node is shared between synchronous and asynchronous traffic. When a node receives the token, it transmits its synchronous traffic for h_i time units where $h_i \le H_i$. Then, the asynchronous messages in the node i, are transmitted for $e_i = \varepsilon_i$ time units, where $\varepsilon_i = (H_i - h_i)$ time units . Essentially, the synchronous and the asynchronous traffic utilize the H_i time units in each node. The BuST protocol do not use the $\boldsymbol{A}$ time units (where $\boldsymbol{A} = \tau$ - H – W). In all, for the BuST protocol (Franchino , Buttazzo, and Facchinetti, 2007), the total time units used for the transmission of data (synchronous and asynchronous) frames in any cycle is given as $\sum_{i=0}^{i=N-1}(h_i + e_i)$ = (h + e) ≤ H . Hence, at its best, the BuST protocol can have an average cycle length of H + W which is less than $\boldsymbol{\tau}$, the average bandwidth available in every cycle (Ozuomba, Chukwudebeb, Obot, 2011).

On the other hand, the STLODGSTT protocol uses $\mathbf{a}_i$ time units in each node i, for the transmission of asynchronous traffic, where $\mathbf{a}_i$ = max(**0**, $\boldsymbol{\tau}$ - $\boldsymbol{\varepsilon}$ - ($\boldsymbol{t_i}$ - $\boldsymbol{t_{i\text{-}N}}$)) + $\mathbf{A_T}$, and A_T = $\frac{\tau \text{ - T}}{\text{N}}$ (Ozuomba, Chukwudebeb, Obot, 2011). According to (Ozuomba, Chukwudebeb, Obot, 2011) , in each cycle, the total time units used for the transmission of asynchronous traffic is given as $\sum_{i=0}^{i=N-1}(a_i)$ ≤ $\boldsymbol{A}$ time units (where A = $\boldsymbol{\tau}$ - H – W) . Also, in each cycle, the total time units used for the transmission of synchronous traffic is given as $\sum_{i=0}^{i=N-1}(h_i)$ = h ≤ H , where h = H - $\boldsymbol{\varepsilon}$. In all, for the STLODGSTT the total time units used for the transmission of data (synchronous and asynchronous) frames in any cycle is given as $\sum_{i=0}^{i=N-1}(h_i + a_i)$ = (h + a) ≤ **(**H + A). Hence, at its best , the STLODGSTT can have an average cycle length of A + H + W which is equal to $\boldsymbol{\tau}$, maximum value of bandwidth available in every cycle.

However, since the STLODGSTT protocol does not use the $\boldsymbol{\varepsilon}$ time units, under load load of the synchronous traffic (h ≤ H, and hence $\boldsymbol{\varepsilon}$ > 0) , the average cycle length for the STLODGSTT protocol drops by $\boldsymbol{\varepsilon}$ time units to a value given as, $\boldsymbol{\tau}$ - $\boldsymbol{\varepsilon}$ which is less that $\boldsymbol{\tau}$ (Ozuomba, Chukwudebeb, Obot, 2011). In order to solve the problem, in this paper, the BuST protocol's spare bandwidth allocation mechanism is incorporated into the STLODGSTT mechanism to give the new protocol with more robust bandwidth reclamation mechanism as presented in subsequent sections.

The Static-Threshold-Limited BuST (STLB) Protocol

The STLB Algorithm and Flowchart

The flowchart of the STLB protocol is presented in Fig 3 while the detailed algorithm is given as ***Protocol PSB MAC Algorithm***.

Protocol PSB (MAC Algorithm)

PSB1: NETWORK INITIALISATION CYCLE

During the first token rotation, to initialize timers, no station is allowed to transmit any packets. First, ***TTRT***, that is $\boldsymbol{\tau}$ is defined to satisfy the deadline requirements of every synchronous message in the network. Then, the following two parameters are also defined, $\mathbf{w_i}$, $\mathbf{H_i}$ for 0 ≤ i ≤ N-1. In addition, $\mathbf{h_i}$ is reset to zero. So, $\boldsymbol{\varepsilon} = \sum_{i=0}^{i=N-1} \boldsymbol{(H_i)} = \sum_{i}^{N-1} \text{H}_i$ where $\boldsymbol{\varepsilon_i} = \mathbf{H_i}$ for 0 ≤ i ≤ N-1, $\varepsilon_i = \boldsymbol{H_i}$;n = N and n$_i$ = 1 (for i = 0, 1,……N-1)

In summary, during the network initialization, the following parameters are defined, initialized or computed:

INITIALIZATION CYCLE :

PSB1.1 Define TTRT (that is τ) and N

PSB 1.2 Define w_i for i = 0,1,..........N-1

```
PSB1.3  Define $H_i$ for i = 0,1,............N-1
PSB1.4    Initialize hi = 0 for I = 0,1, …………..N-1
PSB1.4.1 Initialize $\varepsilon_i = \bar{\varepsilon}_i = H_i$ for i = 0,1,...............N-1
PSB1.4.2 Initialize $e_i = 0$ for i = 0,1,...............N-1
PSB1.4.3  $e = \left( \sum_{i=0}^{i=N-1} (e_i) \right)$
PSB1.5    Initialize $a_{(i-N)} = 0$ for i = 0,1,.............N-1
PSB1.6
          PSB1..6.1  Compute $\varepsilon = \left( \sum_{i=0}^{i=N-1} (\varepsilon_i) \right) = H$
          PSB1..6.2  Compute $\bar{\varepsilon} = \left(\sum_{i=0}^{i=N-1}(\bar{\varepsilon}_i)\right) = \left( \sum_{i=0}^{i=N-1} (\varepsilon_i) \right) - \left( \sum_{i=0}^{i=N-1} (e_i) \right) = H$
PSB1.7  Compute $T = \left( \sum_{i=0}^{i=N-1} (H_i + w_i) \right)$

PSB1.8  Compute $A_T = \frac{\tau - T}{N}$
PSB1.9 Initialize Token rotation timer ($TRT_i$)Timer
       PSB1.9.1  i = 0
       PSB1.9.2  $TRT_i = 0$
       PSB1.9.3  Start TRTi; $TRT_i$ Counts up
       PSB1.9.4  i = i + 1
       PSB1.9.5  Pass the Token to Node i + 1
       PSB1.9.6  IF (i < N) then
                     Goto Step PSB1.9.2
                 Else
                     Goto Step PSB2.1
                 End if.
```

DATA TRANSMITION CYCLE, PART I: TRANSMISSION OF SYNCHRONOUS FRAMES

```
PSB2.1  Check Frames that arrives at Node i
PSB2.2  IF (Frames is Token) then
            Goto Step PSB2.5
        Else
            Goto Step PSB2.3
        End if

    PSB2.3  Process Frame (Store, Ignore, etc.)
    PSB2.4  Goto Step PSB2.1
    PSB2.5  $THT_i = \mathbf{max}(TTRT - \varepsilon - TRT_i ,\ 0)$
    PSB2.6
        PSB2.6.1  $\varepsilon' = \varepsilon - \varepsilon_i$
        PSB2.6.2  $\bar{\varepsilon}' = \bar{\varepsilon} - \bar{\varepsilon}_i$
    PSB2.7.1   $TRT_i = 0$
    PSB2.7.2   Start $TRT_i$
    PSB2.7.3   $TRT_i$ counts up
    PSB2.8 IF ($TRT_i \le H_i$ ) then
                Goto StepPSB2.9
                Else
                Goto Step PSB2.12
                End if
    PSB2.9    IF (Synchronous Frames are Available) then
                    Goto Step PSB2.10
               Eles
                    Goto StepPSB2.11
               End if
PSB2.10   Transmit Synchronous Frames
PSB2.11   Goto Step PSB2.8
PSB2.12   $h_i = TRT_i$
PSB2.13
                PSB2.13.1  $e' = e - e_i$ ;
                PSB2.13.2  $e_i = 0$
                PSB2.13.3 TX =  $TRT_i$ , $TRT_i$  counts up ;
                PSB2.13.4 IF (($TRT_i \le H_i$ ) and (Asynchronous Frames are Available) )
then                 Transmit Asynchronous Frames ;
                          $e_i = H_i - TX$;
                        End if
     PSB2.14    $e = e' + e_i$
     PSB2.15    $\varepsilon_i = H_i - h_i$
     PSB2.16    $\varepsilon = \varepsilon' + \varepsilon_i$
     PSB2.17    $\bar{\varepsilon}_i = H_i - h_i - e_i \therefore \bar{\varepsilon}_i = \varepsilon_i - e_i$
     PSB2.18    $\bar{\varepsilon} = \bar{\varepsilon}' + \bar{\varepsilon}_i$  (Goto Q3.1 )
```

<u>DATA TRANSMITION CYCLE, PART II: TRANSMISSION OF ASYNCHRONOUS FRAMES</u>

```
PSB3.1  $THT_i = THT_i + \min(a_{1-N}, A_T)$
PSB3.2  $a_i = THT_i$
PSB3.3  Start $THT_i$, $THT_i$ counts down
PSB3.4  IF ($THT_i > 0$) then
          Goto Step PSB3.5
          Else
          Goto Step PSB3.8
          End if
    PSB3.5    IF (Asynchronous Frames are Available) then
               Goto Step PSB3.6
               Else
               Goto Step PSB3.8
              End if
    PSB3.6   Transmit Asynchronous Frame
    PSB3.7   Goto Step PSB3.4
    PSB3.8   $a_i = a_i - THT_i$
    PSB3.9   $a_{i-N} = a_i$
```

```
PSB3.10    i = i + 1
PSB3.11    i = (i mod N)
PSB3.12    Pass the Token to Node i   (Goto PSB2.1 )
```

The flowchart of STLB Protocol

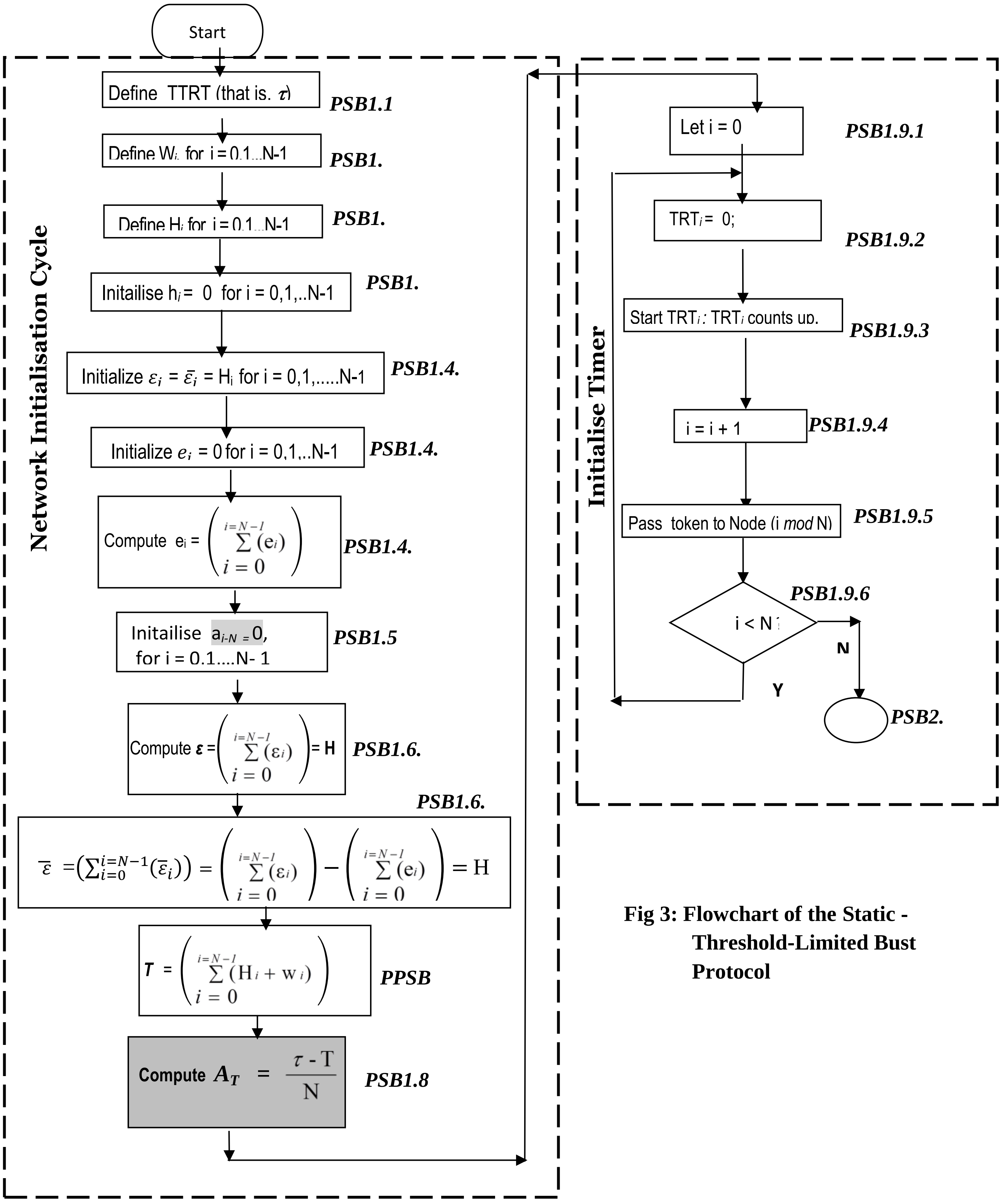
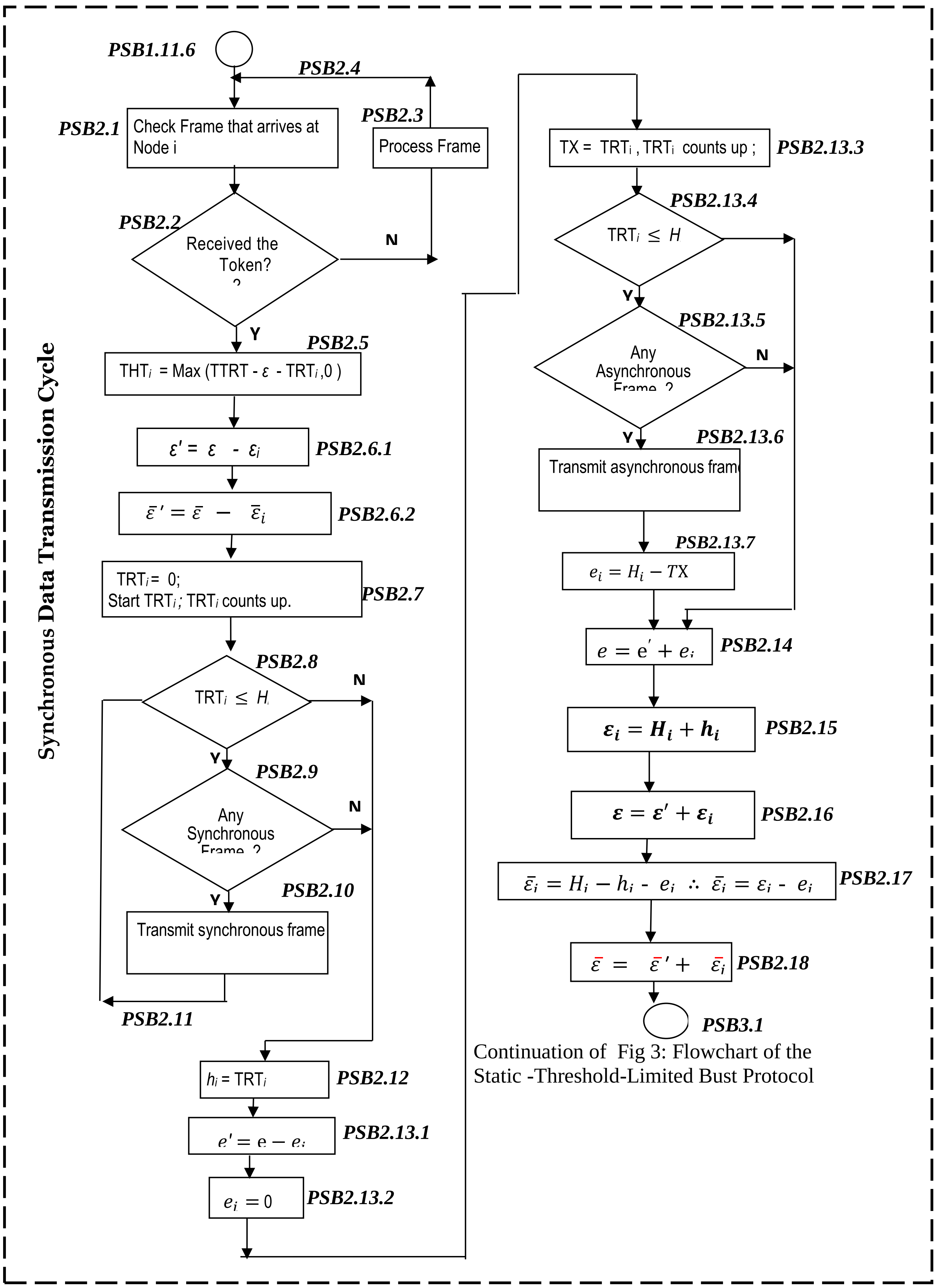
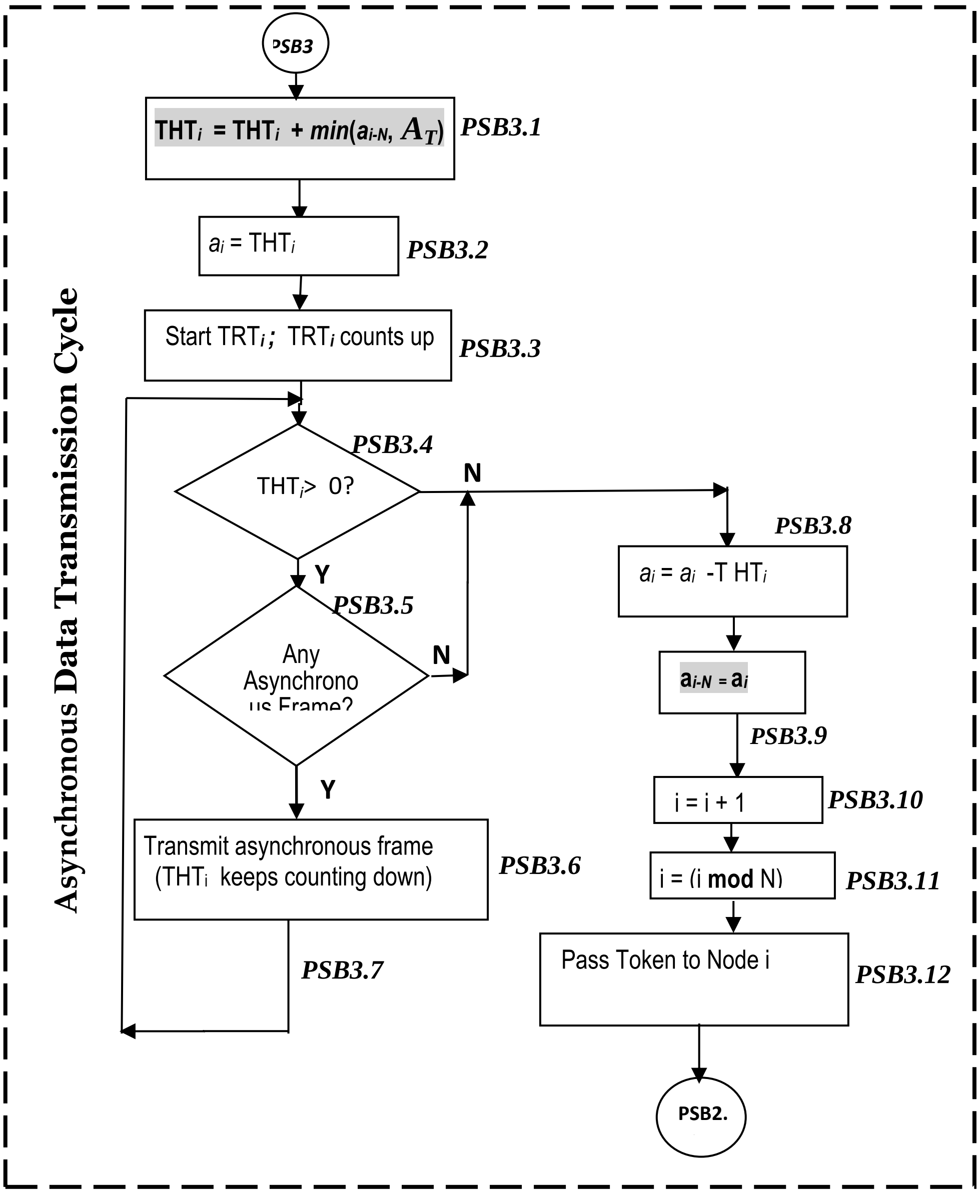

Start
Network Initialisation Cycle
Define TTRT (that is. τ)
PSB1.1
Define W_i for i = 0,1...N-1
PSB1.
Define H_i for i = 0,1...N-1
PSB1.
Initailise h_i = 0 for i = 0,1,..N-1
PSB1.
Initialize $\varepsilon_i = \bar{\varepsilon}_i = H_i$ for i = 0,1,.....N-1
PSB1.4.
Initialize e_i = 0 for i = 0,1,..N-1
PSB1.4.
Compute $e_i = \left(\sum_{i=0}^{i=N-1} (e_i)\right)$
PSB1.4.
Initailise a_{i-N} = 0, for i = 0,1....N- 1
PSB1.5
Compute $\varepsilon = \left(\sum_{i=0}^{i=N-1} (\varepsilon_i)\right) = H$
PSB1.6.
PSB1.6.
$\bar{\varepsilon} = \left(\sum_{i=0}^{i=N-1} (\bar{\varepsilon}_i)\right) = \left(\sum_{i=0}^{i=N-1} (\varepsilon_i)\right) - \left(\sum_{i=0}^{i=N-1} (e_i)\right) = H$
$T = \left(\sum_{i=0}^{i=N-1} (H_i + w_i)\right)$
PPSB
Compute $A_T = \frac{\tau - T}{N}$
PSB1.8
Initialise Timer
Let i = 0
PSB1.9.1
TRT$_i$ = 0;
PSB1.9.2
Start TRT$_i$: TRT$_i$ counts up.
PSB1.9.3
i = i + 1
PSB1.9.4
Pass token to Node (i *mod* N)
PSB1.9.5
PSB1.9.6
i < N ?
N
Y
PSB2.

Fig 3: Flowchart of the Static - Threshold-Limited Bust Protocol


Synchronous Data Transmission Cycle
PSB1.11.6
PSB2.4
PSB2.1
Check Frame that arrives at Node i
PSB2.3
Process Frame
PSB2.2
Received the Token?
N
Y
PSB2.5
THT$_i$ = Max (TTRT - ε - TRT$_i$,0)
$\varepsilon' = \varepsilon - \varepsilon_i$
PSB2.6.1
$\bar{\varepsilon}' = \bar{\varepsilon} - \bar{\varepsilon}_i$
PSB2.6.2
TRT$_i$= 0;
Start TRT$_i$; TRT$_i$ counts up.
PSB2.7
PSB2.8
TRT$_i$ ≤ H$_i$
N
Y
PSB2.9
Any Synchronous Frame ?
N
Y
PSB2.10
Transmit synchronous frame
PSB2.11
h_i = TRT$_i$
PSB2.12
$e' = e - e_i$
PSB2.13.1
$e_i = 0$
PSB2.13.2
TX = TRT$_i$, TRT$_i$ counts up ;
PSB2.13.3
PSB2.13.4
TRT$_i$ ≤ H
Y
PSB2.13.5
Any Asynchronous Frame ?
N
Y
PSB2.13.6
Transmit asynchronous frame
PSB2.13.7
$e_i = H_i - TX$
$e = e' + e_i$
PSB2.14
$\varepsilon_i = H_i + h_i$
PSB2.15
$\varepsilon = \varepsilon' + \varepsilon_i$
PSB2.16
$\bar{\varepsilon}_i = H_i - h_i - e_i \therefore \bar{\varepsilon}_i = \varepsilon_i - e_i$
PSB2.17
$\bar{\varepsilon} = \bar{\varepsilon}' + \bar{\varepsilon}_i$
PSB2.18
PSB3.1

Continuation of Fig 3: Flowchart of the Static -Threshold-Limited Bust Protocol



Continuation of Fig 3: Flowchart of the Static-Threshold-Limited Bust Protocol

Analysis of the STLB Algorithm

The analysis of STLODGSTT protocol as presented in (Ozuomba , Chukwudebeb , Obot , 2011) applies to STLB protocol except that;

a) In STLB, $\mathbf{e_i}$ denote the portion of the H_i time units that are actually used for transmitting asynchronous traffic in station i; then **e** denote the total of e_i per cycle .

b) In the analysis of the STLB protocol, every occurrence of ε_i is replaced with $\bar{\varepsilon}_i$; and ε is replaced with $\bar{\boldsymbol{\varepsilon}}$; where

$$\varepsilon_i = H_i - h_i \quad (9)$$

$$\bar{\varepsilon}_i = \varepsilon_i - e_i = H_i - h_i - \mathbf{e_i} = H_i - (h_i + \mathbf{e_i}) \quad (10)$$

$$\varepsilon = \sum_{i=0}^{i=N-1}(\varepsilon_i) = H - h \quad (11)$$

$$e = \sum_{i=0}^{i=N-1}(e_i) \quad (12)$$

$$\bar{\boldsymbol{\varepsilon}} = \sum_{i=0}^{i=N-1}(\bar{\varepsilon}_i) = \varepsilon - e = H - h - e = H - (h + e) \quad (13)$$

Consequently, following the analytical expressions for the STLODGSTT protocol as presented in (Ozuomba , Chukwudebeb , Obot , 2011), the performance parameters for STLB protocol can be stated as follows;

Upper Bound On Cycle Length, ***max (t_i - t_{i-N})***

According to (Ozuomba , Chukwudebeb , Obot , 2011) , for STLODGSTT protocol

Upper Bound On Cycle Length STLODGSTT protocol = τ - ε when ε > 0 (14)

Upper Bound On Cycle Length STLODGSTT protocol = τ when ε = 0 (15)

Similarly, replacing ε in Equation(14) with $\bar{\boldsymbol{\varepsilon}}$ gives

Upper Bound On Cycle Length STLB protocol = τ - $\bar{\boldsymbol{\varepsilon}}$ when $\bar{\boldsymbol{\varepsilon}}$ > 0 (16)

Now, $\bar{\boldsymbol{\varepsilon}}$ = **e** – ε where **e** ≤ ε, then ;

Upper Bound On Cycle Length STLB protocol = τ + **e** – ε where e ≤ ε and ε ≥ 0 (17)

Upper Bound On Cycle Length STLB protocol = τ when e = ε and ε ≥ 0 (18)

For a network that is heavily loaded with asynchronous traffic, when the synchronous traffic fails to use up its ***H*** time units per cycle, in that case , ε ≥ 0 **,** for the STLODGSTT protocol, the cycle length reaches a maximum value of τ - ε time units since the STLODGSTT protocol does not allow the asynchronous traffic to use the spare bandwidth , ε left by the synchronous traffic. However, for the STLB protocol, the cycle length reaches a higher maximum value of τ time units since the STLB protocol allows the asynchronous traffic to use up all the spare bandwidth , ε left by the synchronous traffic; this occurs in network that is heavily loaded with asynchronous traffic in which case, **e** = ε , for all values of i.

Average Cycle Length, (Ĉ)

Let $\hat{C}_{\mathrm{STLODGSTT}}$ denote the Average Cycle Length, for STLODGSTT protocol and $\hat{C}_{\mathrm{STLB}}$ denote the Average Cycle Length for STLB protocol. Then, according to (Ozuomba , Chukwudebeb , Obot , 2011),

$$\hat{C}_{\mathrm{STLODGSTT}} \le (\tau - T) + (T - \varepsilon) \quad (19)$$

$$\hat{C}_{\mathrm{STLODGSTT}} \le \tau - \boldsymbol{\varepsilon} \quad (20)$$

Similarly, replacing ε in Equation(19) with $\bar{\boldsymbol{\varepsilon}}$ gives

$$\hat{C}_{\mathrm{STLB}} \le (\tau - T) + (T - \bar{\boldsymbol{\varepsilon}}) \quad (21)$$

$$\hat{C}_{\mathrm{STLB}} \le \tau - \bar{\boldsymbol{\varepsilon}} \quad (22)$$

Now, T − $\bar{\boldsymbol{\varepsilon}}$ = T + **e** – ε where **e** ≤ ε

$$\hat{C}_{\mathrm{STLB}} \le (\tau - T) + (T - \varepsilon) + e \text{ where } e \le \varepsilon \quad (23)$$

$\hat{C}_{\text{STLB}} \leq \tau - \varepsilon + e$ where $e \leq \varepsilon$ (24)

Hence, $\hat{C}_{\text{STLB}} = \hat{C}_{\text{STLODGSTT}} + \mathbf{e}$ (25)

The Average Asynchronous Traffic Time Units Per Cycle (AV)

Let $\mathbf{AV}_{\text{STLODGSTT}}$ denote the average time units used in the STLODGSTT protocol to transmit asynchronous in every cycle and $\mathbf{AV}_{\text{STLB}}$ denote the denote the average time units used in the STLB to transmit asynchronous traffic in every cycle. Now, according to (Ozuomba , Chukwudebeb , Obot , 2011),

$$\hat{C}_{\text{STLODGSTT}} \leq (\tau - \text{T}) + (\text{T} - \varepsilon) \quad (26)$$

In the expression for $\hat{C}_{\text{STLODGSTT}}$, $(\tau - \text{T})$, denote the average value of the asynchronous traffic delivered per cycle through the best-effort mechanism , while $(\text{T} - \varepsilon)$ denote the average value of the synchronous traffic transmitted per cycle through the guaranteed service mechanism. Consequently, the average time units used in the STLODGSTT protocol to transmit asynchronous traffic in every cycle is given as (Ozuomba , Chukwudebeb , Obot , 2011),

$$\mathbf{AV}_{\text{STLODGSTT}} = (\tau - \text{T}) \quad (27)$$

Similarly, for the SLTB protocol,

$$\hat{C}_{\text{STLB}} \leq (\tau - \text{T}) + (\text{T} - \bar{\boldsymbol{\varepsilon}}) \quad (28)$$

In the expression for $\hat{C}_{\text{STLB}}$, $(\tau - \text{T})$, denote the average value of the asynchronous traffic delivered per cycle through the best-effort mechanism, while $(\text{T} - \bar{\boldsymbol{\varepsilon}})$ denote the average value of the synchronous and asynchronous traffic transmitted per cycle through the guaranteed service mechanism. The two components of $(\text{T} - \bar{\boldsymbol{\varepsilon}})$ can be separated by replacing $\bar{\boldsymbol{\varepsilon}}$ with $\mathbf{e} - \varepsilon$ where $\mathbf{e} \leq \varepsilon$. This gives,

$$\text{T} - \bar{\boldsymbol{\varepsilon}} = \text{T} - \varepsilon + \mathbf{e} \text{ where } \mathbf{e} \leq \varepsilon \quad (29)$$

In this case, **e** is the asynchronous traffic delivered through the guaranteed service mechanism while the $\text{T} - \varepsilon$ is the synchronous traffic delivered through the guaranteed service mechanism. Hence,

$$\hat{C}_{\text{STLB}} \leq (\tau - \text{T}) + \mathbf{e} + (\text{T} - \varepsilon) \text{ where } e \leq \varepsilon \quad (30)$$

Therefore, $\mathbf{AV}_{\text{STLB}} = (\tau - \text{T}) + \mathbf{e}$ (31)

It can be seen that STLB protocol delivers additional **e** time units of asynchronous traffic on every cycle more than the STLODGSTT protocol.

Comparison Of The Performance Of *STLB Protocol and STLODGSTT Protocol*

Numerical Example

Consider a ring network with any number of stations; the network uses the STLODGSTT protocol and the STLB protocol for its MAC , where the protocol parameters are given as follows: τ = 100, W = 4 for all the nodes, H = 80 and hence T = 84. Now, the network operates under heavy load of asynchronous traffic but with variable load of synchronous traffic. With these given parameters and load configurations, the performance of the protocols vis avis the Average Cycle Length and the Average Asynchronous Traffic Per Cycle computed with respect to the variation in the Ɛ, are given in Table 1 and Table 2 respectively, as well as in Fig 4a and Fig 4b respectively.

RESULTS

Table 1: Average Cycle Length Vs. ℇ (where τ = 100, H = 80, T = 84)

ℇ	h	e	$\hat{C}_{STLODGSTT}$	$\hat{C}_{STLB}$
0	80	0	100	100
8	72	8	92	100
16	64	16	84	100
24	56	24	76	100
32	48	32	68	100
40	40	40	60	100
48	32	48	52	100
56	24	56	44	100
64	16	64	36	100
72	8	72	28	100
80	0	80	20	100

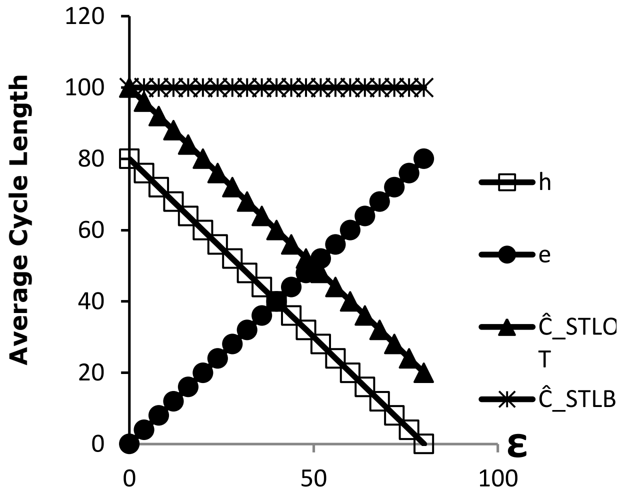


Fig 4a Graph of Average Cycle Length, (ĉ) Vs. ε (where τ = 100, H = 80, T = 84)

Table 2 : Average Asynchronous Traffic Per Cycle Vs ℇ (where τ = 100, H = 80, T = 84)

ℇ	h	E	$AV_{STLODGSTT}$	AV_{STLB}
0	80	0	16	16
8	72	8	16	24
16	64	16	16	32
24	56	24	16	40
32	48	32	16	48
40	40	40	16	56
48	32	48	16	64
56	24	56	16	72
64	16	64	16	80
72	8	72	16	88
80	0	80	16	96

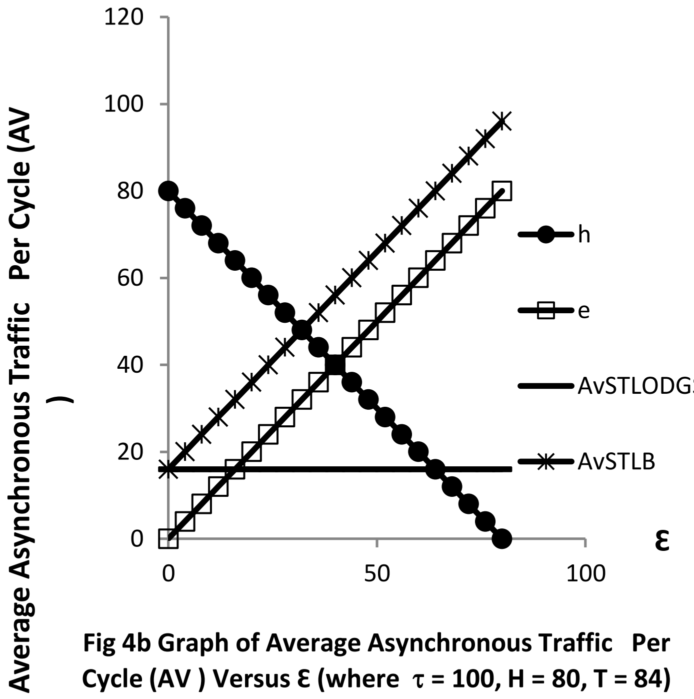


Fig 4b Graph of Average Asynchronous Traffic Per Cycle (AV) Versus ℇ (where τ = 100, H = 80, T = 84)

DISCUSSION OF RESULTS

a. The Average Asynchronous Traffic Time Units Per Cycle

Table 1 shows that STLB protocol maintains maximum value for average cycle length irrespective of the variations in the load level of the synchronous traffic from a maximum value of 80 (h = H = 80) to its minimum value of 0 (h = 0) . This is because when a network is heavily loaded with asynchronous traffic, e = Ɛ, as can be seen in Table 1 and Fig 4a. That means, all the spare bandwidth left by the synchronous traffic are reallocated to the asynchronous traffic, as shown in Table 1 and Fig 4a. Hence, as h drops from 80 to 0, Ɛ and hence **e** increases accordingly , maintaining the overall throughput at its maximum attainable level.

On the other hand, for the STLODGSTT protocol, the average cycle length drops as the load level of the synchronous traffic drops. Specifically, in Table 1, $\hat{C}_{STLODGSTT}$ drops from a maximum value of 100 to 20, as h drops from its maximum value of 80 (h = H = 80) to its minimum value of 0 (h = 0) . In other words, STLB protocol is more effective in utilizing the available bandwidth than the STLODGSTT protocol.

b. The Average Asynchronous Traffic Time Units Per Cycle

From Table 2, STLODGSTT protocol maintained a minimum value of 16 for the average asynchronous traffic per cycle, irrespective of the variations in the load level of the synchronous traffic from 80 (h = H = 80) to 0, (h = 0) . This is because, even if the network is heavily loaded with asynchronous traffic, the STLODGSTT protocol does not allow the asynchronous traffic to use the spare bandwidth, Ɛ left by the synchronous traffic. In essence, the STLODGSTT protocol restricts the asynchronous traffic to only the $\boldsymbol{\tau - T}$ time units (where $\boldsymbol{\tau - T = 100 - 84 = 16}$) , as can be seen in Table 2 and Fig 4b.

On the other hand, for the STLB protocol, the average asynchronous traffic transmitted in every cycle increases as the load level of the synchronous traffic drops. Specifically, the AV_{STLB} increases from a minimum value of 16 when h is at its maximum value of 80 (h = H = 80) ; to a maximum value of 96 when h is at its minimum value of 0 (h = 0). In other words, STLB protocol is more effective in utilizing the spare bandwidth for transmitting additional asynchronous traffic.

CONCLUSION

In this paper, a Static-Threshold-Limited BuST Protocol (STLB) protocol was developed to improve on the ability of the timed token protocol to utilize spare bandwidth in the occasion of fluctuating load levels of the synchronous traffic. The improvement that can be achieved with the new MAC protocol was demonstrated through analytical expressions and numerical example. Specifically, with the STLB protocol, the spare bandwidth left by the synchronous traffic is reallocated to the asynchronous traffic without violating the stringent time constraints required by the synchronous traffic. In essence, with STLB protocol, higher throughput can be realized than is obtainable in the existing Static-Threshold-Limited On-Demand Guaranteed Service Timed Token (STLODGSTT) protocol from which STLB protocol was developed.

RECOMMENDATIONS

The STLB protocol is developed and analyzed under heavy load of asynchronous traffic. However, with respect to the asynchronous traffic, heavy load situations are of different types. In the present analysis, all the nodes are assumed to be heavily loaded with asynchronous traffic. Such is the case of uniform heavy load. There is ongoing research by the authors to examine the performance of timed token protocols under non uniform heavy load of asynchronous traffic. In such situation, not all the nodes are heavily loaded with asynchronous traffic. Preliminary results from the study of other versions of the timed token protocols indicate that their performance differ significantly when examined under uniform and non uniform heavy load situations. It is therefore expedient that the STLB protocol should also be studied under non uniform heavy load of asynchronous traffic.

REFERENCES

Biao C. and Wei Z., (1992). Properties of the Timed Token Protocol. Department of Computer Science Texas A&M University College Station, TX 77843-3112 Oct., 1992 Technical Report 92-038.

Chan E., Chen D., Cao J. and Lee C. H. (1992). The time Properties of the FDDI-M Medium Access Control Protocol. The Computer Journal, Vol. 82 No. 1 , pp. 96-102, Jan. 1999.

Dept. of Defense US (1992). Survivable Adaptable Fibre-Optic Embedded Networks. MIL-STD-2004, US Dept. of Defense, Washington D.C., Sept, 1992.

Franchino G., Buttazzo G. C., and Facchinetti T. (2007). BuST: Budget Sharing TokenProtocol for Hard Real-Time Communication. In Proc. of the 12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2007), Sept. 2007.

Grow R. (1982). A timed token protocol for local area networks. Proc. Electro'82, Token Access Protocols, Paper 17/3, May 1982.

IEEE (1995). The Institute of Electrical and Electronic Engineers, Token-passing bus access method and physical layer specifications. America national Standard ANSI/IEEE std. 802.4 – 1985.

Indumathi G.and Murugesan K. (2010) . A bandwidth efficient scheduling framework for non real time applications in wireless networks .International Journal of Distributed and Parallel systems (IJDPS) Vol.1, No.1, September 2010

Kenneth C. S. and Marjory J. J.,(1987) .Cycle time properties of the FDDI token ring protocol. IEEE Trans. Software. Eng., vol. SE-13, pp. 376-385, Mar. 1987

Lee, D, Attias, R, Puri, A, Sengupta, R, Tripakis, S and Varaiya P (2001). A wireless token ring protocol for intelligent transportation systems. IEEE Intelligent Transportation Systems Conference Proceedings, Aug. 2001, pp. 1152-1157

Malpani N., Vaidya N., and Welch J (2001) . Distributed Token circulation on mobile Ad Hoc Networks. 21th International Conference on distributed computing systems (ICDCS 2001) PHOENIX, Arizona, USA, April 2001, pp.691-701.

Mcguffin L.J., Reid L.O, and Sparks S.R. (1998). MAP/TOP in CIM Distributed Computing. IEEE Network, vol.2, no. 3 , May 1988, pp. 23 – 31.

Nicholas M. and Wei Z.,(1994). The timed-token protocol for real-time communications. Computer, vol. 27, no. 1, pp. 35-41, January, 1994.

Ozuomba S., Chukwudebeb G.A., Obot A.B. (2011). Static-Threshold-Limited On-Demand Guaranteed Service For Asynchronous Traffic In Timely-Token Protocol. Nigerian Journal of Technology, Vol. 30, No. 2, pp124 - 142 , June 2011.

Ozuomba S. and Chukwudebe G.A, (2003). An Improved Algorithm For Channel Capacity Allocation In Timer Controlled Token Passing Protocols. A Journal Paper Published in An International Journal Of Nigerian Computer Society (NCS), Vol. 9 No 1 , pp 116 – 124, June 2003 .

Regnier, P. and G. Lima, (2006). Deterministic integration of hard and soft real-time communication over shared-ethernet. In Proc. of Workshop of Tempo Real (2006), Curitíba, Brazil

Ricardo M. (2010) .Survey of Real-Time Communication in CSMA-Based Networks. Network Protocols and Algorithms ISSN 1943-3581 2010, Vol. 2, No. 1

Shin K. G. and Zheng Q. (1995). FDDI-M: A scheme to double FDDI's ability of supporting synchronous traffic. IEEE Trans. on Parallel and Distributed Systems, Vol. 6, No. 11, pp. 1125 -1131,.Nov. 1995.

Tovar E., and Vasques F. (1991). Setting Target Rotation Time in Profibus Based Real-Time Distributed Applications. Proc. of the 15th IFAC Workshop on Distributed Computer Control Systems, 1998.

Uhlhorn R.W. (1991). The fibre-optic high-speed data bus for a new generation of a military aircraft. IEEE LCS, Vol.2, No.1, pp. 36 – 43, Feb. 1991.

White P. P., (1997).RSVP and Integrated Services in the Internet: A Tutorial . IEEE Communications Magazine • May 1997

Willig A. (2002). Analysis of the PROFIBUS Token passing protocol over wireless links. Proc. of IEEE Int. Symposium on Industrial Electronics (IEEE-ISIE 2002), L'Aquila, Italy, July 2002, pp. 56-61.

Zhang S. and Burns A., (1994 a). Timing Properties of the Timed Token Protocol. Tech. Rept. (YCS 243), Dept of Computer Science, Univ. of York (May 1994).

Zhang S. and Burns A., (1994 b). A Study of Timing Properties with the Timed Token Protocol. Technical Report (YCS 226), Dept. of Computer Sci., Univ. of York (March 1994).