EPOC

CL 102 ISSUES AND PROPOSED SOLUTIONS

Duane Remein &
Hesham ElBakoury
(Huawei)

tqSize

- tqSize is used to generate OctetsRemaining via the formula on pg 95 ln 14 :
  (((stopTime – localTime) x tqSize) – tqOffset)
  - Note if tqSize < 1 the grant will effectively be terminated
- In 64.2.2.1 “Constants” - tqSize is defined as: “This constant represents time_quantum in octet transmission times”
  with a value of 2 Octets/TQ
- In 77.2.2.1 & 102.2.2.1 “Constants” - tqSize is defined with a value of 20 Octets/TQ
- tqSize is defined as an integer

What about EPoC?

- Integers < 1 would need to be rounded down (=0)

EPoC tqSize							
		Bit Loading (b/Hz) / QAM					
RF CH (Mhz)	Num SC	7	8	9	10	11	12
		128	256	512	1024	2048	4096
24	480	0.34	0.38	0.43	0.48	0.53	0.58
30	600	0.42	0.48	0.54	0.60	0.66	0.72
36	720	0.50	0.58	0.65	0.72	0.79	0.86
42	840	0.59	0.67	0.76	0.84	0.92	1.01
48	960	0.67	0.77	0.86	0.96	1.06	1.15
54	1080	0.76	0.86	0.97	1.08	1.19	1.30
60	1200	0.84	0.96	1.08	1.20	1.32	1.44
66	1320	0.92	1.06	1.19	1.32	1.45	1.58
72	1440	1.01	1.15	1.30	1.44	1.58	1.73
78	1560	1.09	1.25	1.40	1.56	1.72	1.87
84	1680	1.18	1.34	1.51	1.68	1.85	2.02
90	1800	1.26	1.44	1.62	1.80	1.98	2.16
96	1920	1.34	1.54	1.73	1.92	2.11	2.30

OctetsRemaining (fyi)

- Defined in 102.2.2.3 “Variables” pg 95 ln 12
- “This variable is an alias for the expression (((stopTime – localTime) x tqSize) – tqOffset). It denotes the number of octets that can be transmitted between the current time and the end of the grant.

EPoC

PHY_DATA_SIZE & PHY_OVERHEAD_SIZE

- Defined in 102.2.2.3 “Variables” pg 96 ln 35
- PHY_DATA_SIZE defined as an integer with “The number of octets constituting the denominator in the EPoC de-rating Equation (102–1). To normalize the effective data rate, the MPCP control multiplexer waits PHY_OVERHEAD_SIZE octets per every PHY_DATA_SIZE octets transmitted”
- PHY_OVERHEAD_SIZE TYPE defined as an integer with “The number of octets constituting (together with PHY_DATA_SIZE) the numerator in the EPoC de-rating Equation (102–1). To normalize the effective data rate, the MPCP control multiplexer waits PHY_OVERHEAD_SIZE octets per every PHY_DATA_SIZE octets transmitted”
- Used to increment fecOffset (next slide)
- Per this definition for β = 12.5 (800 Mb/s PHY rate) any even number will work for PHY_DATA_SIZE. Thus fecOffset could be incremented by any number in the series {…, -600, -400, -200, 200, 400, 600, …}

Eq 102-1

$$\beta = \frac{XGMII\_Rate}{PCS\_Rate} = \frac{PHY\_DATA\_SIZE + PHY\_OVERHEAD}{PHY\_DATA\_SIZE}$$

Ref: garavaglia_3bn_01_0513.pdf, & garavaglia_3bn_02a_0513.pdf

fecOffset (fyi)

- Definition in 102.2.2.3 “Variables” pg 94 ln 22:
  - “A variable that advances by PHY_DATA_SIZE after every 8×(PHY_DATA_SIZE + PHY_OVERHEAD_SIZE) bit times (see EPoC de-rating equation 102-1). After reaching the value of FEC_CODEWORD_SIZE, this variable is reset to zero. In the CLT, this variable is initialized to 0 at system initialization. In the CNU, this variable is assigned in the GATE Processing CNU Activation state diagram (see Figure 102–14)”
  - In Figure 102–14 this is set to magic number16 (inherited from Fig 77-14 ONU Ctrl Mux SD represents 2 x 66b symbols in EPON for optical sync and RS alignment)
- Definition in 77.2.2.3 “Variables”:
  - “A variable that advances by 1 after every 8 bit times. After reaching the value of FEC_CODEWORD_SIZE, this variable is reset to zero. In the OLT, this variable is initialized to 0 at system initialization. In the ONU, this variable is assigned in the GATE Processing ONU Activation state diagram (see Figure 77–14)”
- fecOffset is a 32 bit unsigned variable use in numerous calculations & SDs (CheckGrantSize, P*MD_Overhead, Figure 102–13 and, Figure 102–14 CLT & CNU Ctrl Mux SDs resp))*

Eq 102-1

$$\beta = \frac{XGMII\_Rate}{PCS\_Rate} = \frac{PHY\_DATA\_SIZE + PHY\_OVERHEAD}{PHY\_DATA\_SIZE}$$

packet_initiate_delay

- Defined in 102.2.2.3 “Variables” pg 95 ln 32 is a 16 bit unsigned value defined as:
  “This variable is used to set the time?ut interval for packet_initiate_timer defined in 102.2.2.5. The packet_initiate_delay value is represented in units of octets”
  - 16 bits is too small as the function PMD_Overhead will yield a value of 117,179 at min RF channel (24 MHz) min Mod Order (256 QAM)
- packet_initiate_timer is defined in 102.2.2.5 “Timers” pg 98 ln 27. It is defined as “This timer is used to delay frame transmission from MAC Control to avoid variable MAC delay while MAC enforces IPG after a previous frame. In addition, this timer increases interframe spacing just enough to accommodate the extra parity data to be added by the FEC encoder”

CheckGrantSize

- Defined in 102.2.2.4 “Functions” pg 97 ln 12 as “This function calculates the future time at which the transmission of the current frame (including the FEC parity overhead) is completed“
- The function is called in *Figure 102–13 and, Figure 102–14 CLT & CNU Ctrl Mux SDs resp. in both cases it is used to set OctetsRequired.*
  - OctetsRequired <= CheckGrantSize(sizeof(data_tx) + tailGuard))
- Does not account for
  - multiple FEC sizes
  - FEC foreshortening
- Eq 102-2

$$eckGrantSize\langle length\rangle = \left\lceil \frac{fecOffset + length}{FECPAYLOADSIZE} \right\rceil \times FEC\_CODEWORD\_SIZE - fecOffse$$

CheckGrantSize

An equivalent function definition

```
CkGntSize = 0
length = length + fecOffset
Do While length > 0
    CkGntSize = CkGntSize + fecCWSz
    length = length - fecPldSz
Loop
CkGntSize = CkGntSize - fecOffset
```

PMD_Overhead

- Analogous to FEC_Overhead in 2012 STD (which also appears in *Figure 102–13 CLT Ctrl Mux SDs*)
- Defined in 102.2.2.4 “Functions” pg 97 ln 22 as “This function calculates the additional amount of time (in octet times) that the MPCP control multiplexer waits following transmission of a frame of size ‘length’ by the MAC. The additional time is added to allow the insertion of parity data into the frame by the PHY layer and to adjust the data rate to the effective data rate supported by the PCS and PMD. PMD_Overhead(). returns the number of octets that the PHY inserts during transmission of a particular packet and its subsequent IPG. Parameter ‘length’ represents the size of an entire frame including preamble, SFD, DA, SA, Length/Type, FCS, and IPG.” see formula below
- PMD_Overhead is currently only referenced in Figure 102–14 CNU Ctrl Mux SD where it is used to set packet_initiate_delay
  packet_initiate_delay <= PMD_Overhead(sizeof(data_tx) + tailGuard)
  note tailGuard also includes IPG as does the Eq 102-3 below)
- Probably should also be used in Figure 102-13 for the same calculation (instead undefined param FEC_Overhead is used here)

Eq 102-3

$$PMD\_Overhead((length), \beta) =$$

IPG → $$12 + \left\lceil (\beta - 1) \times length + \beta \times (FEC\_PARITY\_SIZE \times \left\lfloor \frac{fecOffset + length}{FEC\_PAYLOAD\_SIZE} \right\rfloor) \right\rceil$$

Ref. see
garavaglia_3bn_03_0313.pdf pg 23, garavaglia_3bn_02a_0313.pdf
garavaglia_3bn_01_0513.pdf, & garavaglia_3bn_02a_0513.pdf

Summary of calculations

tqSize => OctetsRemaining

OctetsRemaining represents the remaining grant size

PHY_DATA_SIZE & PHY_OVERHEAD_SIZE => β

β is the ratio of XGMII rate to PHY rate

β, fecOffset, FEC_PARITY_SIZE, FEC_PAYLOAD_SIZE

=> PMD_Overhead

PMD_Overhead => packet_initiate_delay

packet_initiate_delay determines wait time between frames to allow for PHY overhead insertion

fecOffset, FEC_PAYLOAD_SIZE, FEC_CODEWORD_SIZE,

=> CheckGrantSize

CheckGrantSize determines OctetsRequired (size of a frame + necessary overhead)

If OctetsRequired ≤ OctetsRemaining a frame is sent to the PHY(Figure 102-13/14)

Summary of Issues

- tqSize
  - Disallows PHY capacities less than 500 Mb/s
  - Excessive rounding error on capacities that are not integer multiples of 500 Mb/s
- PHY_DATA_SIZE & PHY_OVERHEAD_SIZE
  - Poorly defined, very circular and doesn't really state what these variable are
- packet_initiate_delay
  - Current definition to small for EPoC preventing sufficient delay between frames on low capacity channels
- packet_initiate_timer
  - The standard does not indicate how this timer is decremented.
- CheckGrantSize
  - Does not allow for multiple FEC with varying rates
  - Does not allow for shortened last code word
- PMD_Overhead
  - IPG over estimated

EPOC

PROPOSED SOLUTIONS

tqSize solution

EPoC tqSizeC

RF CH (Mhz)	Num SC	Bit Loading (b/Hz) / QAM 7	8	9	10	11	12
		128	256	512	1024	2048	4096
24	480	43.01	49.15	55.30	61.44	67.58	73.73
30	600	53.76	61.44	69.12	76.80	84.48	92.16
36	720	64.51	73.73	82.94	92.16	101.38	110.59
42	840	75.26	86.02	96.77	107.52	118.27	129.02
48	960	86.02	98.30	110.59	122.88	135.17	147.46
54	1080	96.77	110.59	124.42			
60	1200	107.52	122.88	138.24			
66	1320	118.27	135.17	152.06			
72	1440	129.02	147.46	165.89			
78	1560	139.78	159.74	179.71			
84	1680	150.53	172.03	193.54			
90	1800	161.28	184.32	207.36			
96	1920	172.03	196.61	221.18			

- Scale by some factor (2^7 suggested)
- Change name to avoid confusion

EPoC tqSizeC (percent Error)

RF CH (Mhz)	Num SC	Bit Loading (b/Hz) / QAM 7	8	9	10	11	12
		128	256	512	1024	2048	4096
24	480	0.02%	0.31%	0.54%	0.72%	0.86%	0.99%
30	600	1.41%	0.72%	0.17%	1.04%	0.57%	0.17%
36	720	0.79%	0.99%	1.14%	0.17%	0.37%	0.54%
42	840	0.35%	0.02%	0.79%	0.48%	0.23%	0.02%
48	960	0.02%	0.31%	0.54%	0.72%	0.12%	0.31%
54	1080	0.79%	0.54%	0.33%	0.17%	0.04%	0.54%
60	1200	0.48%	0.72%	0.17%	0.39%	0.57%	0.17%
66	1320	0.23%	0.12%	0.04%	0.57%	0.46%	0.37%
72	1440	0.02%	0.31%	0.54%	0.17%	0.37%	0.08%
78	1560	0.56%	0.47%	0.40%	0.34%	0.30%	0.26%
84	1680	0.35%	0.02%	0.28%	0.02%	0.23%	0.02%
90	1800	0.17%	0.17%	0.17%	0.17%	0.17%	0.17%
96	1920	0.02%	0.31%	0.08%	0.31%	0.12%	0.31%
min		0.02%		median	0.31%		
max		1.41%		avg	0.37%		

- Adjust formula for Octets Remaining

Solution (cont)

- OctetsRemaining with scaled tqSize:

(((stopTime – localTime) x tqSizeC)/128 – tqOffset)

EPoC

PHY_DATA_SIZE & PHY_OVERHEAD_SIZE

Redefine as follows:

- PhyOutDataSize
  - An integer representing the number of MAC data octets that can be sent at the PHY rate in the time it takes to transmit PhyInDataSize octets at the XGMII rate.
- PhyInDataSize
  - An integer number representing the number of octets that must be sent at the XGMII rate while PhyOutDataSize are transmitted at the PHY rate.

PHY_DATA_SIZE & PHY_OVERHEAD_SIZE

- Redefine fecOffset
  - “A variable that advances by PhyOutDataSize after every 8×(PhyInDataSize) bit times (see EPoC de-rating equation 102-1). After reaching the value of FEC_CODEWORD_SIZE, this variable is reset to zero. In the CLT, this variable is initialized to 0 at system initialization. In the CNU, this variable is assigned in the GATE Processing CNU Activation state diagram (see Figure 102–14)”
- Redefine Eq 102-1

$$\beta = \frac{XGMII}{Phy\ Rate} = \frac{PhyInDataSize}{PhyOutDataSize}$$

packet_initiate_delay & packet_initiate_timer

- packet_initiate_delay
  - extend size from 16b to at least17b (18b would be safer)
- packet_initiate_timer
  - Explicitly state that this timer is decremented once ever octet transmission time at the PHY rate

EPoC

CheckGrantSize for Multi FEC

```
CkGntSizeMulti = 0
  length = length + fecOffset
  Do While length > 0
    i = 0
    For i = 0 To fecNum - 1
      If length >= fecPldSz(i) Then
        CkGntSizeMulti = CkGntSizeMulti + fecCwSz(i)
        length = length - fecPldSz(i)
        i = i + 1
        Exit For
      End If
    Next i
    i = i - 1
    If length > 0 Then
      CkGntSizeMulti = CkGntSizeMulti + fecCwSz(i)
      length = length - fecPldSz(i)
    End If
  Loop
  CkGntSizeMulti = CkGntSizeMulti – fecOffset
```

Not optimized for FEC foreshortening, burst size, etc.

CheckGrantSize for Multi FEC with shortened last codeword

```
GntSize = 0
length = length + fecOffset
Do While length > 0
    ' Calculate size of  full CWs except smallest fec
    i = 0
    For i = 0 To fecNum – 2              ‘ for each FEC
        Do While length > fecPldSz(i)
        length = length - fecPldSz(i)
        GntSize = GntSize + fecCwSz(i)
    Loop
    ‘ Calculate size for remaining data
    i = 0
    For i = 0 To fecNum – 1              ‘ for each FEC
        tmpCw = Fix(length / fecPldSz(i))
        ' Fix is a floor function (rounds towards 0)
        If length - tmpCw * fecPldSz(i) > 0 Then
            ‘ shortened last CW
            cwFit(i) = length + (tmpCw + 1) *
                  fecPrtySz(i)
        Else
            ' no shortened last CW
            cwFit(i) = length + tmpCw * fecPrtySz(i)
        End If
    Next i
    ' select optimal Size for remaining length
    i = 0
    For i = 0 To fecNum – 2              ‘ for each FEC
        If cwFit(i) < cwFit(i + 1) Then
            GntSize = GntSize + cwFit(i)
            length = length - cwFit(i)
        End If
    Next i
    ' pick up smallest shortened cw if len >  0
    If length > 0 Then
        GntSize = GntSize + cwFit(2)
        length = length - cwFit(2)
    End If
Loop
' return calculated length
CheckGrantSize = GntSize - fecOffset
```

Note :this will need to be converted from visual basic into “quasi C”

IPG over estimation

- PMD_Overhead(length) change definition as shown:
  - This function calculates the additional amount of time (in octet times) that the MPCP control multiplexer waits following transmission of a frame of size 'length' by the MAC. The additional time is added to allow the insertion of parity data into the frame by the PHY layer and to adjust the data rate to the effective data rate supported by the PCS and PMD. PMD_Overhead(). returns the number of octets that the PHY inserts during transmission of a particular packet ~~and its subsequent IPG~~. Parameter 'length' represents the size of an entire frame ~~including~~ excluding preamble, SFD, DA, SA, Length/Type, FCS, and IPG (which are accounted for in tailGuard). The following formula is used to calculate the overhead:
  - formula changed (remove "12 +" for IPG):

$$PMD\_Overhead(length, \beta) = \left\lceil (\beta - 1) \times length + \beta \times \left(FEC\_PARITY\_SIZE \times \left\lfloor \frac{fecOffset + length}{FEC\_PAYLOAD\_SIZE} \right\rfloor \right) \right\rceil$$

THANK YOU