Technical Report

Tools for Simulation Output Analysis

Paulo Baltarejo Sousa

Luís Lino Ferreira

HURRAY-TR-070604

Version: 0

Date: 06-26-2007

Tools for Simulation Output Analysis

Paulo Baltarejo SOUSA, Luís Lino FERREIRA

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: {pbsousa,llf }@dei.isep.ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

This technical report presents a description of the output data files and the tools used to validate and to extract information from the output data files generated by the Repeater-Based Hybrid Wired/Wireless Network Simulator and the Bridge-Based Hybrid Wired/Wireless Network Simulator.

Index

List of Figures

Tools for Simulation Output Analysis

1.Introduction

Output data analysis is the examination of the data generated by a simulator and this examination has two purposes. Firstly, it is used to verify and validate the simulator and its simulation model. Secondly, it is used for testing, evaluating the performance of different scenarios and different systems configurations. Additionally, when the input variables are random values, the output data exhibits random variability. Therefore, the output data is used to estimate the confidence level, or to determine the number of observation required to achieve a desired precision.

The objective of this technical report is to present and describe the information produced by the Repeater-Based Hybrid Wired/Wireless PROFIBUS Network Simulator (RHW2PNetSim) [1] and by the Bridge-Based Hybrid Wired/Wireless PROFIBUS Network Simulator (BHW2PNetSim)[2]. It describes the output data files generated by both simulators from which it is possible to extract results.

The structure of this document is as follows. Section 2 presents the Timeline Visualization Tool and Section 3 describes Microsoft Excel-based tool to output data analyse.

2.Timeline Visualisation Tool

Figure 1 shows a screenshot of the Timeline Visualisation Tool which provides a way to show the network events using Gant Diagrams. This tool was developed using Microsoft Foundation Classes (MFC) [3] and C++ programming language. This figure depicts a diagram drew using the data files generated by BHW2PNetSim. In this figure it is possible to see the events accomplished by each module instance. When a `Master` module instance operates also as BM the events related to `BM` module instance are separately shown. On the other hand, the events accomplished by a bridge are also separated by each `BM` module instance that composes it. For example, Figure 1.shows that the bridge B3 is composed by BMs M7 and M10, thus the events of each `BM` module instance (BM_M7 and BM_M10) that composes a bridge are individually shown.

To illustrate the importance of this tool, in Figure 1 three transactions are highlighted using arrows: one IADT (between master M1 and slave S1) and two IDTs (between master M2 and slave S6 and master M1 and slave S5).

This tool was also of paramount importance for debugging and validating of the both simulation models, since it provides a temporal overview of the network events. Further, it is possible to check the characteristics of all events by a double click on the event object. Figure 2shows a screenshot of this feature using output data files generated by RHW2PNetSim. In this figure a message box shows the information related to the indicated event.

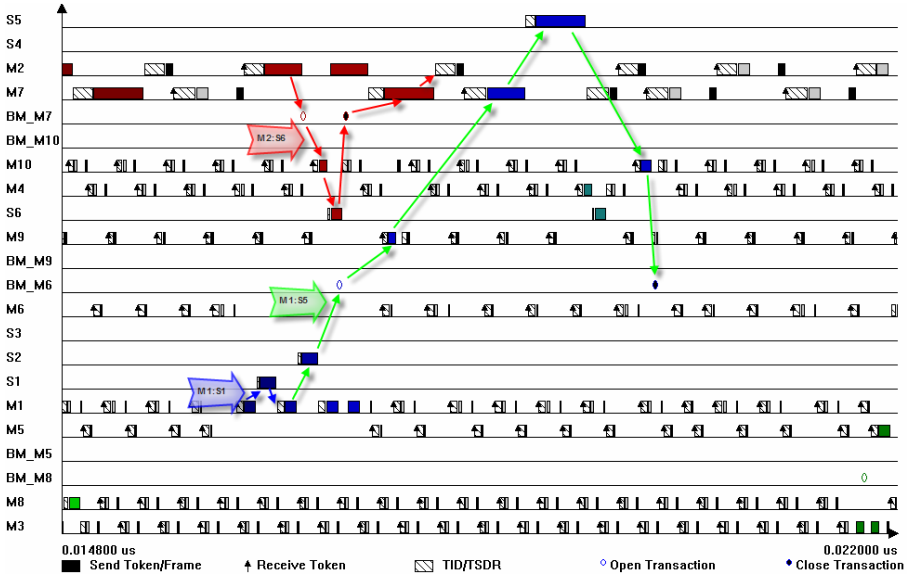


Figure 1– Screenshot of Timeline Visualisation Tool (BHW2PNetSim)

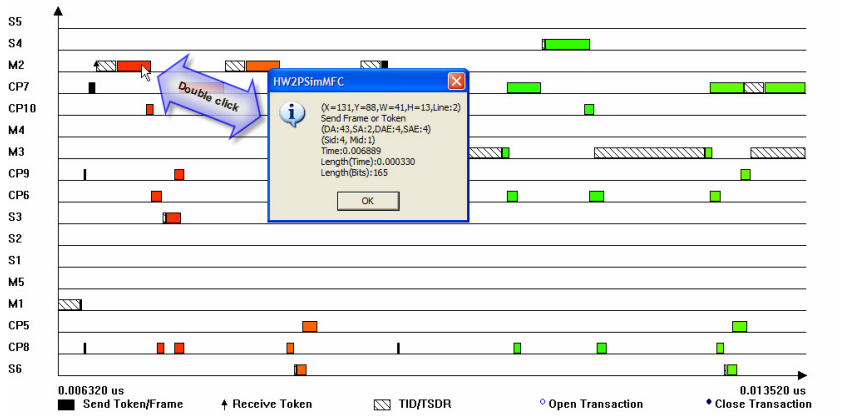


Figure 2 – Screenshot of Timeline Visualisation Tool (RHW2PNetSim)

In order to the RHW2PNetSim and BHW2PNetSim gather this information the `Controller` module `_output_gant_diagram` parameter must be set equal to 1. This diagram is built using two kinds of files. One kind contains the network configuration (with extension ".cfg") and is generated by the `Controller` module instance. The other kind contains the module instance events (with extension ".evt") which are generated by the other modules instances.

3.Output Data Analysis Tool

In order to extract information from the output data files and especially due to amount of information generated by he RHW2PNetSim and BHW2PNetSim a tool was developed which provides a fast way to decode text files containing the simulation results and present simulation statistical results in a convenient format. Output Data Analysis Tool was developed using Microsoft Excel and Visual Basic for Applications (VBA) [4].

Figure 3 depicts a screenshot of this tool. This tool permits the analysis of the message stream response time, stations state machine evolution over time, probability distribution functions and data related to bit error models.

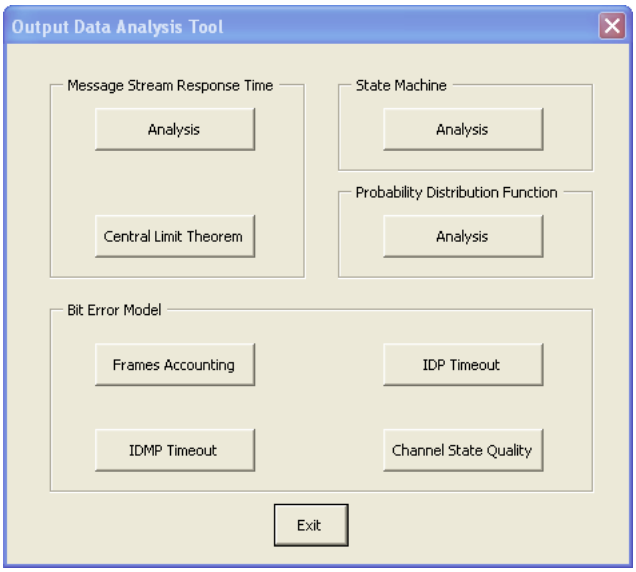


Figure 3 – Screenshot of the Output Data Analysis Tool

3.1. Message Stream Response Time

In order to compute the message stream response time `Master` and `Slave` module instance are able to gather information about transactions. This information is stored in text files which use the ".srt" extension and contain information depicted in Figure 4. In the first column (with the header "ID") is the identifier of the stream. The follows column contains the `Destination Address` (DA), the `Source Address` (SA), the `Destination Address Extension` (DAE) and the `Source Address Extension` (SAE).

The sixth column contains the time when the stream is queued on the `DLL` module instance output queue. The first transmission of the request frame appears in the seventh column (named FTxReq), the first transmission of the response frame appear in the eighth column (named FTxResp) and the last reception, i.e., when the transaction is finished appears in the ninth column (named LRecep). The last column displays path related information, the first item is the initiator's domain, when message is queued; the second item is the domain name to which the initiator belongs when the first request is transmitted; on the third item appears the domain name to which the responder belongs when it replies; on the fourth item contains the domain name to which the station belongs when the transaction finishes.

ID	DA	SA	DAE	SAE	Queued	FTxReq	FTxResp	LRecep	Domains
7	46	3	7	7	0.000000	0.000560	0.001120	0.005619	D1:D1:D3:D1
7	46	3	7	7	0.010000	0.010588	0.011265	0.015621	D1:D1:D3:D1
7	46	3	7	7	0.020000	0.020505	0.021207	0.025593	D1:D1:D3:D1
7	46	3	7	7	0.030000	0.030675	0.032323	0.035648	D1:D1:D3:D1
7	46	3	7	7	0.040000	0.040580	0.042337	0.045570	D1:D1:D3:D1
7	46	3	7	7	0.050000	0.050675	0.051481	0.055802	D1:D1:D3:D1
7	46	3	7	7	0.060000	0.060522	0.061403	0.065519	D1:D1:D3:D1
...									

Figure 4 – Output response time file (excerpt)

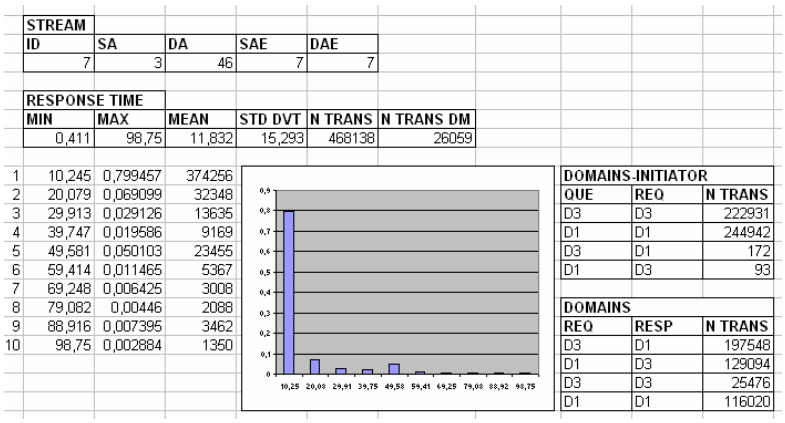
Note that, if the transaction is a SDR the transaction, then it only finishes when the initiator receives the response frame. But if it is a SDN then the transaction finishes when the "responder" receives the request frame. In the last case no contain any information.

The number of transaction that missed its deadline is also stored in a file. This information is recorded in text files (using the ".sdm" extension) by each `Master` module instance.

3.1.1. Message Stream Response Time Statistical Analysis

The tool is capable of decode the text files described in the last Section and retrieve statistical results. The response time is computed as a difference between the timestamp of the last reception (ninth column of the text file) and of the timestamp when the message was queued (sixth column of the text file).

Figure 5 shows a screenshot of a spreadsheet created by this option. It provides information about message streams characteristics, like: minimum (MIN); maximum (MAX); mean (MEAN); standard deviation (STD DVT); number of transaction (N TRANS) and number of transaction that missed the deadline (N TRANS DM). Further, this option builds a histogram of the message stream response time values.



STREAM					
ID	SA	DA	SAE	DAE	
7	3	46	7	7	

RESPONSE TIME					
MIN	MAX	MEAN	STD DVT	N TRANS	N TRANS DM
0,411	98,75	11,832	15,293	468138	26059

1	10,245	0,799457	374256
2	20,079	0,069099	32348
3	29,913	0,029126	13635
4	39,747	0,019586	9169
5	49,581	0,050103	23455
6	59,414	0,011465	5367
7	69,248	0,006425	3008
8	79,082	0,00446	2088
9	88,916	0,007395	3462
10	98,75	0,002884	1350

DOMAINS-INITIATOR		
QUE	REQ	N TRANS
D3	D3	222931
D1	D1	244942
D3	D1	172
D1	D3	93

DOMAINS		
REQ	RESP	N TRANS
D3	D1	197548
D1	D3	129094
D3	D3	25476
D1	D1	116020

Figure 5 – Screenshot of spread sheet created by Message Stream Response Time Analysis option

Additionally, it also provides information about the domain location of the initiator and responder during a transaction. This information is particularly important for IDTs involving wireless mobile stations. Since this option shows to which domain the initiator was belonging when a message was queued (QUE) and to which domain it belongs when the message was sent (REQ). Another kind of information provide by this option is related to the domain location when the initiator sends the request (REQ) the domain location of the responder when it replies (RESP).

3.1.2. Central Limit Theorem

The Central Limit Theorem is an option (Figure 6) that provides a way to compute the confidence interval of the message stream response time values according to the central limit theorem [5]. The lower bound and the upper bound of this interval is computed as mean value (MEAN) less error (ERROR) value and mean value more error value, respectively.

STREAM				
ID	SA	DA	SAE	DAE
1	1	41	1	1

CENTRAL LIMIT THEOREM			
MIN	MAX	MEAN	ERROR
0,275	99,994	13,91383	0,139599

RUN	MIN	MAX	N REG	MEAN
1	0,332	98,394	2583	14,04966
2	0,39	97,846	2575	14,23609
3	0,396	96,519	2588	13,8597
4	0,408	97,94	2626	13,99751
5	0,425	97,45	2566	14,69568
6	0,364	98,94	2590	13,95077
7	0,323	99,839	2580	14,32199

Figure 6– Screenshot of spreadsheet created by Message Stream Response Time Central Limit Theorem option

3.2. State Machine

A `Master` module models a PROFIBUS master and additionally can model the BM, DMM and GMM functionalities separately or simultaneously. Each of these elements has its own state machine. The information about the state machine transitions of these elements are recorded in text files (with ".stt" extension). Each line of this kind of file represents a transition. The transition instant (Time), the state name and a brief explanation about reason that causes the transition appear in first, second and third column, respectively. Figure 7 illustrates an example of this kind of files related to a `Master` module instance.

Time	State name	Description
…		
0.019468	USE_TOKEN	Received token from [6]
0.019534	AWAIT_STATUS_RESPONSE	Waiting for a FDL response from [3]
0.019648	USE_TOKEN	Slot time expired
0.019648	PASS_TOKEN	Trying to pass the token to [5]
0.019671	CHECK_TOKEN_PASS	Waiting for activity from [5]
0.019759	ACTIVE_IDLE	Activity detected from [5]
0.019848	USE_TOKEN	Received token from [6]
0.019914	AWAIT_STATUS_RESPONSE	Waiting for a FDL response from [4]
0.020028	USE_TOKEN	Slot time expired
0.020028	PASS_TOKEN	Trying to pass the token to [5]
0.020051	CHECK_TOKEN_PASS	Waiting for activity from [5]
0.020139	ACTIVE_IDLE	Activity detected from [5]
0.020228	USE_TOKEN	Received token from [6]
0.020294	PASS_TOKEN	Trying to pass the token to [5]
0.020317	CHECK_TOKEN_PASS	Waiting for activity from [5]
…	.	.

Figure 7 – Output state machine file (excerpt)

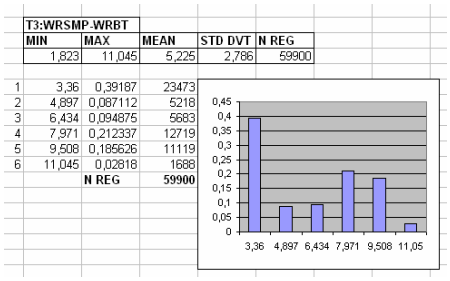
3.2.1. State Machine Statistical Analysis

The State Machine Analysis option (Figure 8) provides a fast way to summarise the information. This option builds histogram related to each transition computing the number of times (N REG) that a `Master` module instance was in each state. Additionally, it computes the minimum (MIN) and maximum (MAX) time spending in each state as well as the mean (MEAN) and the standard deviation (STD DVT).

3.3. Probability Distribution Function

The information about the random values generated by the probability distribution function (PDF) are recorded into several text files (different extension are used, for example, the files related with the T_{ID} and with the T_{SDR} has ".tid" and ".tsdr" extensions, respectively). Figure 9 presents an example of this

kind of files. The first line is used to identify the PDF and its parameters. In this case, the PDF is a triangular distribution function with apex at 50 bit times and extremes at 11 and 70 bit times.



T3:WRSMP-WRBT				
MIN	MAX	MEAN	STD DVT	N REG
1,823	11,045	5,225	2,786	59900

1	3,36	0,39187	23473
2	4,897	0,087112	5218
3	6,434	0,094875	5683
4	7,971	0,212337	12719
5	9,508	0,185626	11119
6	11,045	0,02818	1688
	N REG		59900

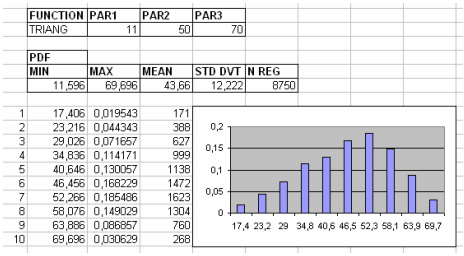
Figure 8 – Screenshot of spreadsheet created by State machine Analysis option

```
TRIANG#11.000000#50.000000#70.000000
18.103736
54.119785
62.908148
51.248531
26.079176
43.094876
66.415956
...
```

Figure 9 – Output PDF file (excerpt)

3.3.1. Probability Distribution Function Statistical Analysis

This output of the Probability Distribution Function Analysis option is similar to the previous. It computes some statistical elements like mimimum (MIN), maximum (MAX), mean (MEAN), standard deviation (STD DVT) values as well as a histogram. Figure 10 shows a screenshot of a spreadsheet created by this option.



FUNCTION	PAR1	PAR2	PAR3
TRIANG	11	50	70

PDF				
MIN	MAX	MEAN	STD DVT	N REG
11,596	69,696	43,66	12,222	8750

1	17,406	0,019543	171
2	23,216	0,044343	388
3	29,026	0,071657	627
4	34,836	0,114171	999
5	40,646	0,130057	1138
6	46,456	0,168229	1472
7	52,266	0,185486	1623
8	58,076	0,149029	1304
9	63,886	0,086857	760
10	69,696	0,030629	268

Figure 10 – Screenshot of spreadsheet created by the Probability Distribution Functions Analysis option

3.4. Bit Error Model

The information about the Bit Error Model (BEM) used in simulation runs are recorded in several output data files. First, includes information about the number of correct and corrupted transmitted frames. Second includes detailed information about corrupted frames transmitted. Third, includes information about IDT deleted and fourth includes information about IDMP aborted. The last one includes information about channel state quality and the frame transmitted by each `Domain` module instance. The third and fourth are only generated by the BHW2PNetSim and fifth is generated only if the BEM used is either Gilbert-Elliot Model (simplified or not) or Burst-Error Periodic Model.

3.4.1. Frame Accounting

The number of valid and invalid frames is also recorded to files (with ".cfr" and ".efr" extensions), as well as the information about the invalid frames relayed by each `Domain` module instance. The information is grouped into four groups: PROFIBUS, BEACON, IDP and IDMP-related frames, where the first is related to standard PROFIBUS frames, the second are the beacon frames, the third are the IDF used by the IDP protocol and the last group represents the frames related to the IDMP. For each group is presented the number of valid and invalid frames relayed by a `Domain` module instance. Figure 11 presents the information generated by a `Domain` module instance of the BHW2PNetSim.

```
PROFIBUS:8954:57
BEACON::
IDP: 246: 6
IDMP: 131:2
```

Figure 11 – Output frame accounting file (excerpt)

Detailed information about corrupted frames is also recorded to file. In such kind of file each line is composed by several fields separated by colons (see Figure 12). The first field is the timestamp at which an invalid frame was detected, the second and the third fields are the DA and SA contained in the frame, respectively. The frame's type appears in the fourth field. The remaining fields contain the remaining frame parameters: `Start Delimiter` (SD), `Frame Control` (FC) and the `Mobility Code` (MC).

```
0.209914:4:6:IDMP:::IQ_REQ:REQUEST_OR_SEND_REQUEST_FRAME:SEND_DATA_WITH_NO_ACKNOWLEDGE_HIGH
0.214466:44:3:IDF:::REQUEST_OR_SEND_REQUEST_FRAME:SEND_AND_REQUEST_DATA_HIGH
0.220326:5:4:PROFIBUS:FDL_REQUEST_STATUS::REQUEST_OR_SEND_REQUEST_FRAME:REQUEST_FDL_STATUS_WITH_REPLY
0.222138:42:1:PROFIBUS:::REQUEST_OR_SEND_REQUEST_FRAME:SEND_AND_REQUEST_DATA_HIGH
0.226439:5:4:PROFIBUS:FDL_REQUEST_STATUS::REQUEST_OR_SEND_REQUEST_FRAME:REQUEST_FDL_STATUS_WITH_REPLY
0.226918:45:1:PROFIBUS:::REQUEST_OR_SEND_REQUEST_FRAME:SEND_AND_REQUEST_DATA_HIGH
0.227865:4:1:PROFIBUS:TOKEN:::
...
```

Figure 12 – Information about invalid frames relayed by a `Domain` module instance

Figure 13 depicts a screenshot of the spreadsheet generated by Bit Error Model Frame Accounting option, where the information contained on the referred kind of files is summarized.

FRAME TRANSMISSION

	PROFIBUS	BEACON	IDP	IDMP
NO ERROR	123767334		1870962	1515670
ERROR	8407600		342954	221932

PROFIBUS

	FDL		FRAME	
TOKEN	REQUEST	RESPONSE	REQUEST	RESPONSE
4382639	3226415	20741	423497	354308

IDP

REQUEST	RESPONSE
178863	164091

IDMP

SMP	RSMP	PBT	RBT	SBT	IQ_REQ	BCN	D_REQ	D_RESP	RU	VOID
15627	24213	2753	2292	1628	41277	15524	47641	18916	52061	0

Figure 13 – Screenshot of spreadsheet created by the Bit Error Model Frame Accounting option

3.4.2. IDP Timeout

The IDP has a error recovery mechanism which deletes an entry from a BM_{ini} LOT if the timeout timer associated with that transaction expires. This behaviour allows the BM_{ini} to initialise a new LOT entry related to the same message stream.

The information about deleted IDTs in a BM_{ini} LOT is recorded in a file with the ".tidt" extension. Figure 14 shows an example of this kind of file. The information gathered in this file is the DA, SA, message ID and the timestamp when the deletion occurred.

Based on the information contained in this file a spreadsheet tool allows the analysis of the results.

Figure 15 depicts a screenshot of the spreadsheet where the information concerning IDTs deleted by BM M8 is shown. The information is organized by message stream.

DA	SA	DAE	SAE	Timestamp
…				
43	2	4	4	0.080469
46	2	5	5	0.156737
46	2	5	5	0.237376
43	2	4	4	0.250335
46	2	5	5	0.260427
46	2	5	5	0.293211
46	2	5	5	0.332400
46	2	5	5	0.445243
46	2	5	5	0.518283
46	2	5	5	0.565210
46	2	5	5	0.668706
46	2	5	5	0.748602
46	2	5	5	0.773215
43	2	4	4	0.080469
46	2	5	5	0.156737
…				

Figure 14 – Output deleted IDTs file (excerpt)

BM_M8 - IDP TIMEOUT				
DA	SA	DAE	SAE	N REG
46	3	7	7	136294
44	3	6	6	207629
43	3	8	8	85867
46	4	9	9	36270

Figure 15 – Screenshot of spreadsheet created by the Bit Error Model IDP Timeout option

3.4.3. IDMP Timeout Timers

Concerning IDMP, four timers are assigned to the GMM and one to each BM (($T_{BM\text{-}IDMPAbort}$)) and another to each DMM (`DMM_IDMP_Abort_Timer` ($T_{DMM\text{-}IDMPAbort}$)) presents in the network. Two of the timers associated to the GMM are used to detect and handle the errors during the Phase 1 (`GMM_Phase_1_Alert_Timer` ($T_{GMM\text{-}P1Alert}$) and ($T_{GMM\text{-}P1Abort}$)), while the others two are related to the Phase 2 (`GMM_Phase_2_Alert_Timer` ($T_{GMM\text{-}P2Alert}$) and `GMM_Phase_2_Abort_Timer` ($T_{GMM\text{-}P2Abort}$)).

In (BHW2PNetSim)[2] a mechanism was proposed to provide the IDMP with capabilities to operate in error-prone environments. This proposed mechanism is based on the timers: `BM_IDMP_Abort_Timer`, `DMM_IDMP_Abort_Timer`, `GMM_Phase_1_Alert_Timer`, `GMM_Phase_1_Abort_Timer`, `GMM_Phase_2_Alert_Timer` and `GMM_Phase_2_Abort_Timer`. Whenever a timer expires the simulator records information about it.

Figure 16 depicts part of this file (".tidmp" extension) which contains information about the expiration of the IDMP timers concerning the GMM. The first column contains the time when the timer expired and in second contains the identification of the expired timer.

timestamp	Timer
…	
108.211156	Phase 1 alert
109.211156	Phase 1 alert
109.222311	Phase 1 abort
109.412431	Phase 2 alert
109.611156	Phase 1 alert
109.622311	Phase 1 abort
111.211156	Phase 1 alert
111.216033	Phase 2 alert
111.413714	Phase 2 alert
113.011156	Phase 1 alert
114.211156	Phase 1 alert
114.222311	Phase 1 abort
108.211156	Phase 1 alert
109.211156	Phase 1 alert
109.222311	Phase 1 abort
…	

Figure 16 – Output IDMP alerts and aborts file (excerpt)

To analyse this results a spreadsheet-based tool has also been developed. Figure 17 depicts a screenshot of these results which contains the number of timer that the IDMP was triggered and the IDMP timers that expired.

GMM_M6 - IDMP TIMEOUT	
IDMP	59900
PHASE 1 ALERT	287
PHASE 1 ABORT	5
PHASE 2 ALERT	206
PHASE 2 ABORT	1

Figure 17 – Screenshot of spreadsheet created by the Bit Error Model IDMP Timeout option

3.4.4. Channel State Quality

In order to model burst sensitive models like the Gilbert-Elliot bit error model, there is the need to compute the state of the channel during time. This information can also be recorded to output data files by each `Domain` module instance. Figure 18 shows an example of this kind of file (which uses the ".cst" extension). The identification of the BEM used and its parameters are written in the first line. The following lines show, in the first column, the timestamp when state change occurred and the second column show when the new state.

The main objective of this feature is to validate the error model in use, by displaying statistical data regarding its operation.

```
#GILBERT_ELLIOT#0.327037#0.672963#0.000082#0.002889
0.0000000000        GOOD
0.0000005000        BAD
0.0000010000        GOOD
0.0000020000        BAD
0.0000025000        GOOD
0.0000040000        BAD
0.0000050000        GOOD
0.0000060000        BAD
0.0000065000        GOOD
0.0000085000        BAD
0.0000095000        GOOD
...
```

Figure 18 – Output channel state quality file (excerpt)

Figure 19 shows a screenshot of the spreadsheet related to channel state quality of one domain. This tool summarizes information regarding the periods in time during which the channel in one domain has been in the good or in the bad state of the Gilbert-Elliot bit error model. The tool provides some statistical parameters, like minimum (MIM), maximum (MAX), mean (MEAN), standard deviation (STD DVT) and the number of times that this domain was in this state (N REG). Additionally it also constructs a histogram of these timings.

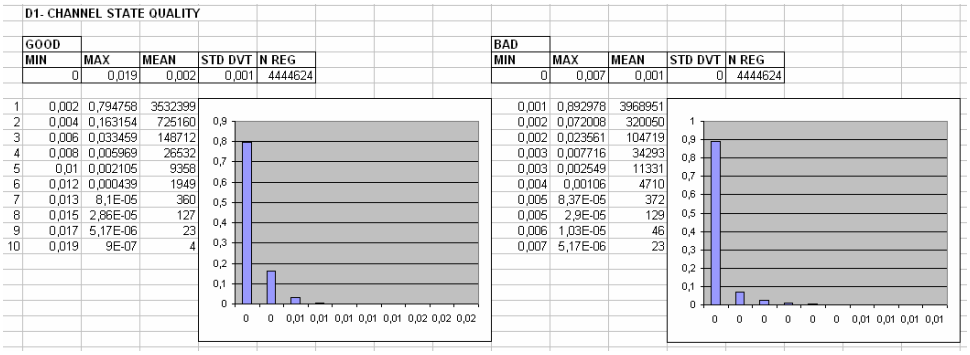


Figure 19 – Screenshot of spreadsheet created by the Bit Error Model Channel State Quality option

References

[1] P. Sousa and L. Ferreira, "Repeater-Based Hybrid Wired/Wireless PROFIBUS Network Simulator," Polytechnic Institute of Porto., Porto, Technical-Report Hurray-tr-060402, April 2006.

[2] P. Sousa and L. Ferreira, "Bridge-Based Hybrid Wired/Wireless PROFIBUS Network Simulator," Polytechnic Institute of Porto, Porto, Technical-Report Hurray-tr-050403, April 2005.

[3] J. Prosise, "Programming Windows With MFC, Second Edition ": Microsoft Press, 1999.

[4] J. Simon, "Excel Programing: Your visual blueprint for creating interactive spreadsheets". New York: Hungry Minds, Inc, 2002.

[5] A. M. Law and W. D. Kelton, "Simulation Modeling and Analysis", 3rd ed. New York: McGraw-Hill, 2000.