DESIGN NOISY DIGITAL COMMUNICATION CHANNEL EMULATOR BASED-ON WEIBULL DISTRIBUTION

Ibrahim Aref
Computer Engineering Department,
University of Tripoli,
Tripoli - Libya
iaref@tripoliuniv.edu.ly

Tarek El-Mihoub
Computer Engineering Department,
University of Tripoli,
Tripoli - Libya
tmihoub@tripoliuniv.edu.ly

Mohamed Aburkhiss
Faculty of Computer Technologies,
Tripoli - Libya
aburkhiss@yahoo.co.uk

ABSTRACT

This paper presents a new approach to model and simulate a Stochastic digital wireless channel with a controlled value of Bit-Error-Rate (BER) for Point-to-Point (P2P) and Point-to-Multipoint (P2M) platforms based on SystemC which is a modelling environment and can be used to express system functionality at various levels of abstraction. The data packet structure that has been used with this system is based on a High-Level Data Link Control (HDLC) standard. The challenge was how to efficiently simulate this channel by using the hardware modelling features available in SystemC, and then how we have developed this channel. Also discussed how the modelling features can be used to simulate different noise sources that can be used to introduce noise events depends on Weibull distribution.

KEYWORDS

SystemC, Modelling, Simulation, Channel, Wireless.

1 INTRODUCTION

With the growing demand of digital systems facilities, designing a digital channel under different environment becomes an important topic to achieve a digital communication system successful with high performance, however becomes more difficult due to the complexity of interactions that occur between hardware and software and the need to model each component at multiple levels of detail. At the moment, channel modeling is an important research topic in interconnection networking area, so this paper presents a new approach that can be used to design, model and simulate a point to point (P2P) and point to multipoint (P2M) digital channel platforms and investigated their performance under different conditions. The simulation tool is a SystemC[1, 2] that is provided various features to perform system level modeling model ling and simulation, which are missing in the generic HDL's such as VHDL and Verilog[3]. It is intended to enable system level design and IP exchange. The challenge was how to efficiently simulate this channel by using the hardware modeling features available in SystemC, and then how we have developed this channel. Also discussed how the modeling features can be used to simulate different noise sources that can be used to introduce noise events depends on specific distribution.

The complexity of digital systems increases. This has resulted in a higher degree of design automation and the need to increase the level of abstraction of a system description, so the modeling process of these systems is becoming more complicated. SystemC[4, 5] which is a hardware description language is more powerful and efficient in this side. It has been used in this paper as simulation tool to design and model a digital channel due to its versatility, high modeling fidelity, and ability to express system functionality at various levels of abstraction. This work addresses a digital communication channel estimation and implementation for point to point, point to multipoint and multipoint to multipoint platforms with very high speed data rate. The first topology, P2P, is addressed before on [6], but it was modeled based-on exponential distribution. Therefore this work is extended and improved by focusing on P2M topology. The last topology, Multipoint-to-Multipoint, is communication scenarios used to support contention and noncontention based shared wireless channel access and will be covered in another work.

The paper starts with briefly description of SystemC features and how we can model and simulate a digital system using SystemC, then structural description of the system as a hierarchical set of behavioral modules and finally presents simulation results with discussion and conclusion.

ISBN: 978-0-9853483-3-5 ©2013 SDIWC

2 WIRELESS DIGITAL CHANNEL MODEL

2.1 Transmitter and Receiver Modeling

The wireless communication system that is modeled and designed in this paper, shown in Figure (1), consists of one or more transmitter model and one or more receivers and channels models, which can be used to create P2P, P2M and/or M2M channels. The structural designs for these platforms are implemented in SystemC using modules, ports, processes and signals which represent the fundamental constructs of SystemC libraries. The transmitter, receiver(s) and channel(s) models have been implemented by using module and connected together through the ports which are created from the base class `sc_port<interface[,N]>port_name` and are bound to an interface type as illustrated in the next program segment[4].

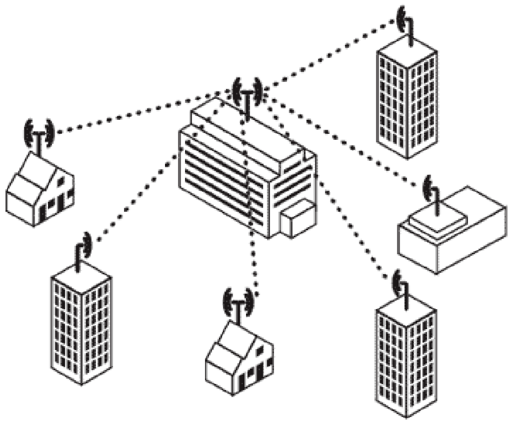

Figure 1. Point-to-Multipoint Channel Platform

```
SC_MODULE(transmitter) {
sc_port<sc_fifo_out_if<sc_bv<N> >,0 >
tpackout;
sc_port<sc_fifo_out_if<sc_bv<N>
>,0>transmitter_data;
...
...
};
SC_MODULE(receiver) {
sc_port<sc_fifo_in_if<sc_bv<N> > >
rpackin;
sc_port<sc_fifo_out_if<sc_bv<N> > >
receiver_data;
...
...
};
```

2.2 Digital Channel Modeling

The digital channel being designed here employs FIFOs as interface type and elasticity buffer. FIFO by simplicity, it is a first-in first-out buffer[4]. Each FIFO has a number of slots for storing values. The number of slots is by default fixed to sixteen during elaboration time. In this system, FIFOs have been used to buffer information packets as they traverse from point to point in the chain, because we have to guarantee that all the packets will remain in the right order during transmitting process. For the simulation purpose, data packets are represented by binary vector data type and are designed based on HDLC[7, 8]. Figure(2) shows the implementation of the channel model by using FIFOs and interface[9].

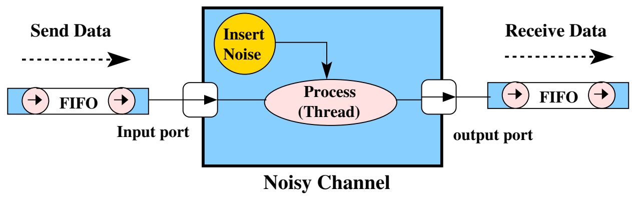

Figure 2. Wireless Digital Channel Module and Interfaces

2.3 Mechanism of Noise Generation

The performance of any communication system is effected by noise and interference from other resources (Figure(3)). From this point of view, our idea to introduce noise in the communication channel is done by using sophisticated process that depends on the accuracy of modeling noise as a random process, however this technique require more sophisticated calculations [10]. The noise has been introduced by determining the time or position of occurrence of next error. This event is a positive random number counts the number of errors occurring in a given interval[11].

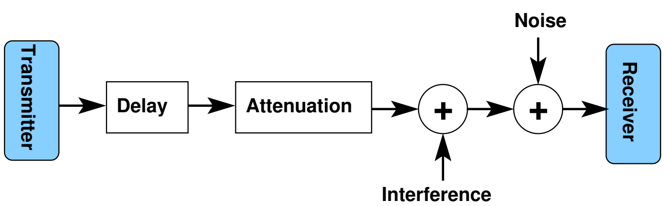

Figure 3. Block Diagram of Wireless Communication System

From this assumption, let T denote the waiting time until the first error occurs when observing this process in which the mean number of errors in

ISBN: 978-0-9853483-3-5 ©2013 SDIWC

the unit interval is k. Then T is a continuous type random variable, and let proceed to find its distribution function. This has been done depending on purely stochastic process of the continues Weibull distribution for $k > 0$ (Figure (4)). The probability density function of a Weibull random variable x is:

$$f(x;\lambda,k) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k} & x \geq 0, \\ 0 & x < 0, \end{cases}$$

where $k > 0$ is the shape parameter and $\lambda > 0$ is the scale parameter of the distribution. The process of noise generation is illustrated in the following steps:

1) We have to choose S/N value that is suitable for the system that we are going to design it.
2) Getting the corresponding BER from the waterfall curves (Figure(5)) relate BER and $\frac{S}{N}$. In this experiment, QPSK is selected as modulation technique.
3) Calculate $k = \frac{1}{BER}$and also calculate random value according to P.D.F.
4) Finally, insert it in simulation program.

Moreover a description of each platform will now be presented in the following sections.

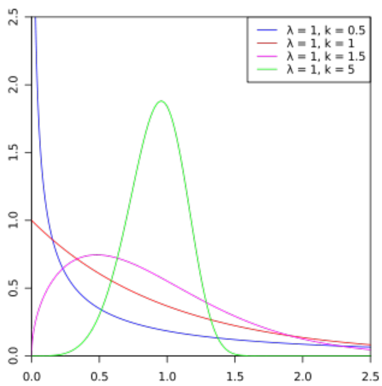

Figure 4. The probability density function (pdf) of Chi-squared distribution

2.4 Channel Topologies

In P2P platform, the channel has been designed to implement the communication scheme just between the transmitter and receiver models as shown in Figure(6-A). The basic design consists of a transmitter model, a receiver model, noise model, compare model and a model representing the data transfer medium which called digital channel and

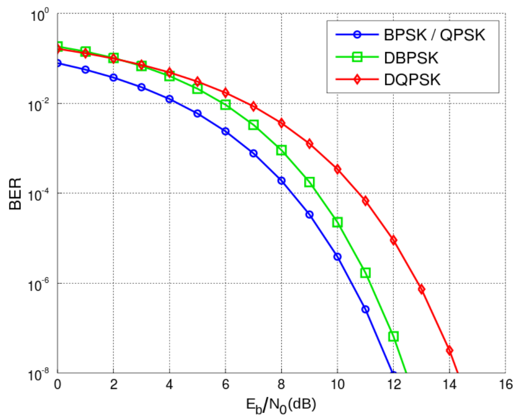

Figure 5. Bit error curve

can be modeled wired and/or wireless. In this work, we have focused on wireless networks.

The transmitter model sends data packets that are designed and created based on HDLC format to the channel which is received those packets, and sends them on to the receiver. The digital channel can be introduced different types of errors from noise model to represent the actual error rate of the system. The noise model has introduced errors randomly by applying Stochastic technique which is used to simulate the error event [10]. The occurrence of the error event is dependent on the selected distribution and then the error rate will be related to it. In the other side, the receiver model receives data packets with some error from the channel model and then send them to the compare model. Finally, the compare model is used to analysis the packets in order to detect the errors and then evaluate error rate by comparing the packet that is sent from the transmitting model with the packet received by receiving model. The process continues until all data packets are sent.

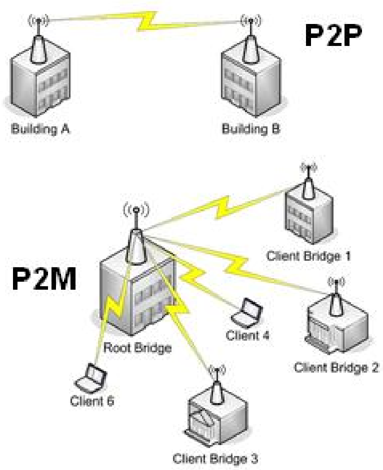

Figure 6. Wireless Communication Channel Topologies

In the other side, P2M platform (Figure(6-B)) is created by using the multiport and/or port_array feature of SystemC and provide the second parameter of `sc_port<interface[,N]>port_name` as illustrated in the previous program segment, the array size parameter can be set. At this point the idea is to provide of a number of like-defined ports. Here we assumed that our communications system have a number of interfaces all with the same connectivity and a number of *N* devices connected to it. Lines(10-14) in the next program segment shows how we can create N transmitter models by using dynamic instantiation feature. Moreover by using this feature of the modules; we can generate as much as we wanted from these devices (models). After that, all the modules that have been instanted (created) have to bind together. Lines(19-22) illustrate how transmitter module has been binded to the channels, also by the same manner, binding process is applied between receivers modules and channels modules as shown in the lines (24-27). Finally, we have to note that dynamic instantiation and binding must be done during elaboration of the simulation.

2.5 Data Packatization

In data transmission process that achieved in this system, there is no mechanism to mark the beginning or end of a packet, so by using HDLC technique, the beginning and end of each packet has to be identified. This is done by using a packet delimiter, or flag, which is a unique sequence of bits that is guaranteed not to be seen inside a packet[12]. In this system, we have used Asynchronous Balanced Mode (ABM) which either combined stations may initiate a transfer[12, 8]. Therefore, to test and evaluate the noisy digital channel described in this work, a test platform consisting of the P2P and P2M platforms described earlier using Go-Back-N ARQ and a packet format based on HDLC (Figure(7)) has been constructed, as described in [8].

Flag	Address	Control	Information	Packet ID	Flag
8 bits	8 bits	8 bits	16 bits	16 bits	8 bits

Figure 7. HDLC Format

2.6 ARQ Communication Protocol

```
SC_MODULE(top) {
SC_CTOR(top){
...
...
transmitter *Transmitter[NTransmitters];
receiver *Receiver[MReceivers];
channel *Channel[MChannels];
compare *Compare[MChannels];
...
...
// Generate (N = NTransmitters)
Transmitters
for (unsigned i=0; i < NTransmitters;
i++) {
std::ostringstream trans_name;
trans_name << "Transmitter" << i <<
std::ends;
std::string s = trans_name.str();
Transmitter[i] = new
transmitter(s.c_str());}
...
...
//this for loop to bind the Transmitter
to Channels trough TCh_fifo
Transmitter[0]->clock(Tclock);
for (unsigned i=0; i< MChannels; i++){
//bind fifos to module ports
Transmitter[0]->tpackout(*TCh_fifo[i]);
Transmitter[0]->t_data(*TComp_fifo[i]);}
//this for loop to bind the Receivers to
ChR_fifos and Receivers to RComp_fifos
for (unsigned i=0; i< MReceivers; i++){
Receiver[i]->clock(Rclock);
Receiver[i]->rpackin(*ChR_fifo[i]);
Receiver[i]->r_data(*RComp_fifo[i]);}
...
...
}//end of constractor
};
```

Automatic Repeat reQuest (ARQ) is an error-control approach that can be used for data transmission. It uses acknowledgements and timeout signals to perform reliable data transmission over an unreliable service [8, 13]. For example, if the transmitter does not receive an acknowledgment before the timeout, it usually retransmits the packet until the transmitter receives an acknowledgment or exceeds a predefined number of re-transmissions. This means that, at any time, if the transmitter detects an error when exchanging data with the receiver, specified packets have to be retransmitted [14]. There are three types of ARQ protocols, as follows:

- Stop-and-wait ARQ
- Go-Back-N ARQ

- Selective Repeat ARQ

These protocols exist in the Data Link or Transport Layers of the OSI model [14].

2.6.1 Stop-and-wait ARQ

In a backward error control system, a packet is transmitted by a transmitter which then waits for the receiver to indicate that it has received this packet before transmitting the next packet. If the packet is corrupted or lost, no acknowledgement will be sent by the receiver. The transmitter will wait for certain period of time. If the acknowledgement does not arrive within this time, the transmitter then re-transmits the previous (lost) packet and waits again for an indication of its receipt. This type of data link protocol is known as a "stop and wait" ARQ, or idle RQ [13, 14, 15]. As above is referred to as an Automatic ReQuest protocol, where there is not always a requirement to wait for the receipt of the ACK associated with a packet before transmitting the next packet. Idle RQ protocol is just one specific example of an ARQ protocol - one which is quite inefficient as considerable time is spent waiting for data rather than transmitting.

2.6.2 Window Flow Control

To overcome the inefficiency associated with idle RQ, it is possible to transmit new packets up until a given limit without waiting for any received ACKs [8, 16]. The number of outstanding unacknowledged packets is known as the window and is fixed to some maximum value, and leads to the idea of a sliding window protocol. Idle RQ protocol is often referred to as a sliding window protocol with a window size of 1. Use of windows in this manner provides a degree of flow control and prevents a transmitting node overwhelming a receiver (quite likely in the case of re-transmissions), as well as ensuring an upper limit on the buffer space required [13, 15].

Therefore, to deal with the lost packets, for every packet transmitted, the transmitting node maintains a timer. If a particular timer lapses and an acknowledgement has not been received, the transmitter assumes that the packet is lost and that another has to be re-transmitted [14]. Alternatively, breaks in sequences can be used to indicate that a packet has been lost. There are then two re-transmission options available. Either the node transmits only the packet that was in error, this approach is called selective re-transmission; or it also re-transmits all the packets transmitted after the lost packet, this method is called Go-Back-N transmission [8, 14, 15]. This work simulates the latter approach.

i- Selective Repeat ARQ

In a selective scheme, the node transmits only the packet that has errors. This has the advantage of being less wasteful of link capacity, but relies on the receiver being able to ensure that packets are delivered to the network layer in the correct order. In sliding window protocols, there is a number called a sequence number; its role is critical [8, 15]. As a packet is transmitted it is given a number identifying when it was output in relation to other packets. These numbers are used by the receiver to ensure that the packets are passed onto the network layer in the correct order [13, 14].

The range of sequence numbers allowed is a function of both window size and retransmission method. In the case of a sliding window protocol with a window size of 1, only a one bit sequence number is required [14]. In general, with a window of K packets, the range of sequencing numbers required is 0 to $2K + 1$ for a system utilising selective re-transmission and 0 to K+1 for Go-Back-N systems. In practice, sequence number fields within packets are preset, usually to 3 bits or 8 bits.

ii- Go-Back-N ARQ

In a Go-Back-N scheme, which is modelled in this work, the transmitting node also retransmits all the packets that were transmitted after the lost packet. Go-back-N retransmission would be used when no re-sequencing could be done, and the retransmission order would be preserved at the cost of lost capacity [13, 15, 16]. In a Go-Back-N system the receiver uses sequence numbers to identify when an error has occurred or when it has received duplicate packets. If a packet has been received correctly it will have a sequence number N, and the next packet received should have sequence number $N + 1$. If the number received/detected is $N + 2$, a packet has been lost [8, 14].

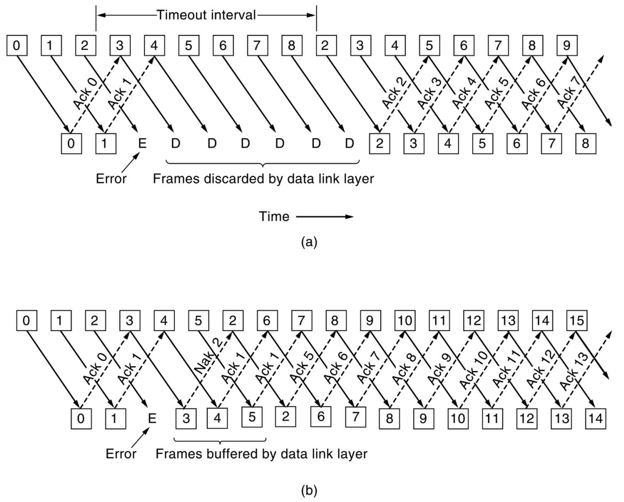

Figure 8. (A) Go-Back N retransmission - (B) Selective re-transmission

3 RESULTS

The simulation program is designed to investigate *BER* for P2P and P2M channel models, the transmitter model is implemented to generate N packets that have been processed by the system and then noise model has introduced errors into these packets randomly which called N_e. The *BER* is determined using Monte Carlo simulation (MCS). The packet error probability cannot be determined but rather is estimated by using MCS ($P_e = \frac{N_e}{N}$) and evaluated after the generation of each packet.

We have performed MCS with different value of λ that represented Packet Error Ratio $\frac{1}{BER}$. In the simulation of P2P, we have set k which is actually $\frac{1}{BER}$ value (probability of bit error) is 0.015. Each experiment in this simulation is based on $N = 15,000$ transmitted packets. The result of replicating the random experiment of passing 15,000 packets through the random channel is shown in Figure (9). The randomness is evident in that the *BER* based on any number of transmissions $N \leq 15,000$ gives a spread of results. This spread is related to the variance of the estimate and in general, in order for simulation results to be useful, the spread should be small. Note that, for the results shown, the variance grows smaller as N grows larger. This is typical behavior for a correctly developed estimator.

Also note that for large N , the results cluster about the true probability or error, and we tend to believe that, for a correctly developed simulation, the estimator P_e, will converge to the probability of error P_r, consistent with the relative frequency definition of probability. This is also typical of correctly developed estimators.

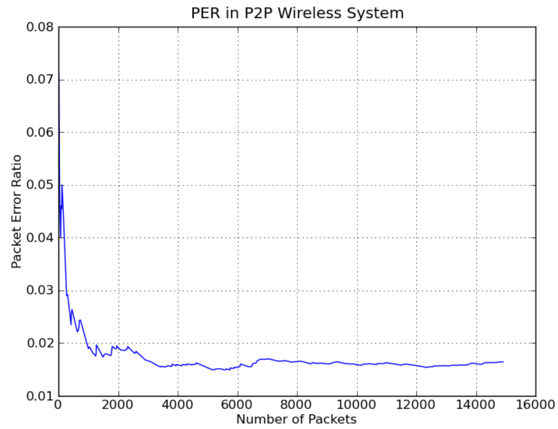

Figure 9. *BER* of the P2P Channel Platform using Monte Carlo technique .

For the P2M platform we have set $0.001 \leq k \leq 0.03$ randomly to ensure that we will get different values of k, also each channel to process each data packet with different noise. In this simulation, we have got the values of k as shown in Table(I) with number of packets $N = 15,000$. Figure(10) shows the P2M *BER* result with six channels and six receivers.

Table I
PER Values for P2M Wireless System

Channel ID	k (BER)
Ch-0	0.006
Ch-1	0.008
Ch-2	0.009
Ch-3	0.017
Ch-4	0.018
Ch-5	0.019

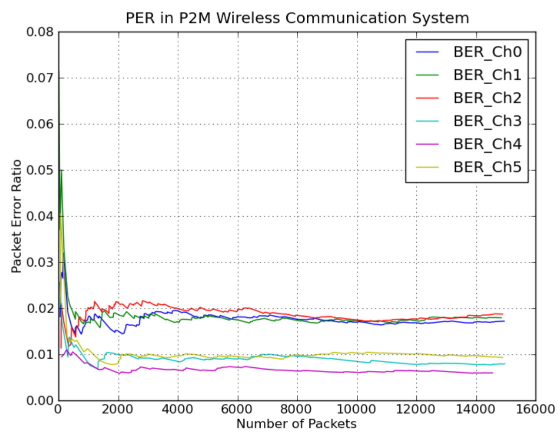

Figure 10. *BER* of the P2M Channel Platform for communication system with three nodes

We have seen that the noisy channel has been simulated correctly and our platform has verified the correct simulation of the system and the correct injection of errors in the bit stream. A sample of the expected throughput for P2P (shown in Figure (11)) and for P2M (shown in Figure (12)) has been determined by the monitor with an ARQ window size set to 7. The transmitted packet for ARQ Go Back-N (N=7) is illustrated in Figure(13) and the discarded packets at the received side is shown in Figure(14).

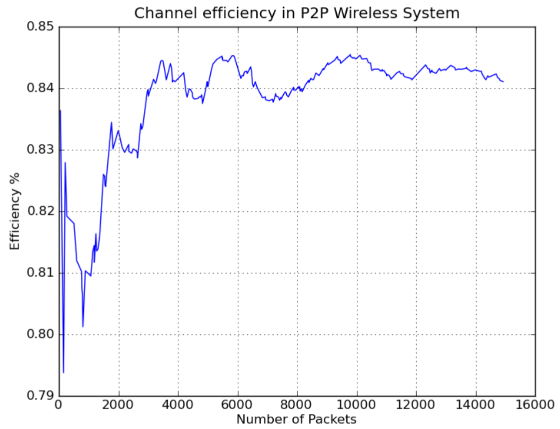

Figure 11. System throughput for P2P with ARQ window size 7

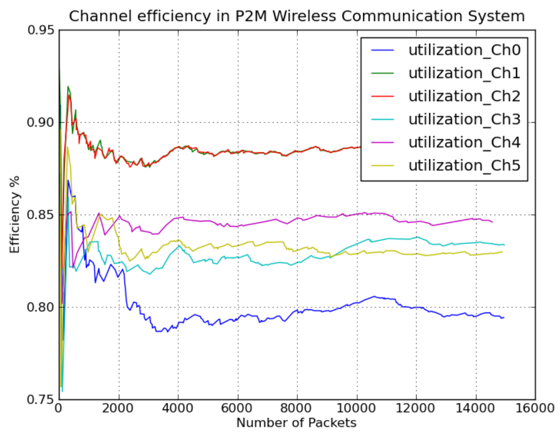

Figure 12. System throughput for P2M with ARQ window size 7

4 CONCLUSION

This paper has explored a new approach to model and develop a digital channel using SystemC at a high level of abstraction and then as future work we can refine the design down to a level that allows hardware implementation (RTL level). Although, much works have found in literature on modeling with SystemC, but to our knowledge, this is a new technique that has been done in digital channel modeling using SystemC, and much work with it, is still as future potential. Also, we can say that SystemC is provided various features to perform system level modeling and simulation, which are missing in the generic HDL's such as VHDL and Verilog and it is intended to enable system level design and IP exchange.Moreover, we need to say that SystemC is not the perfect solution for all tasks, but it combines many design characteristics that are missing in other language.

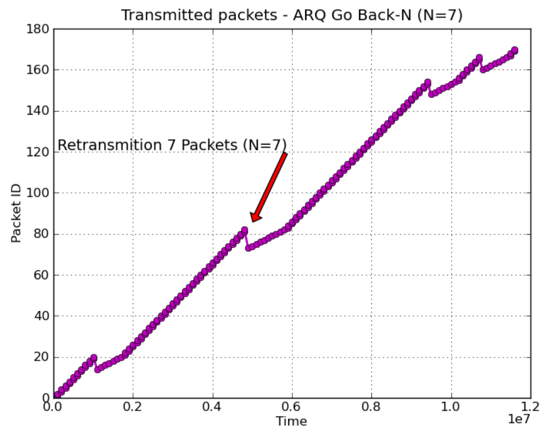

Figure 13. Transmitted packets for ARQ Go Back-N (N=7) system

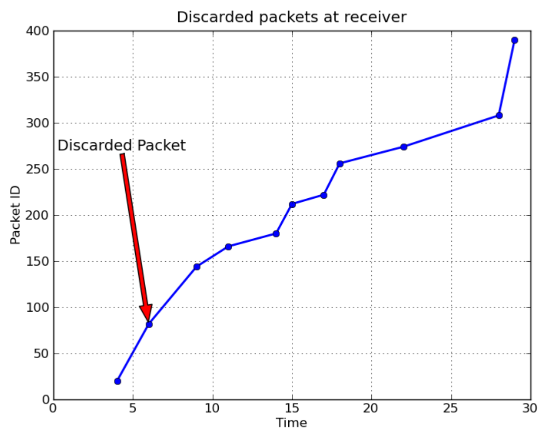

Figure 14. The discarded packets at receiver

REFERENCES

[1] T. Groetker, S. Liao, G. Martin, and S. Swan, *System Design with SystemC*. 2002.

[2] D. Black and J. Donovan, *SystemC: From the Ground-up*. Kluwer Academic Publishers, first ed., 2004.

[3] J. Bjornsen and T. Ytterdal, "Behavioral modeling and simulation of high-speed analog-to-digital converters using systemc," *IEEE Cir-*

cuits and Systems, ISCAS '03, vol. 3, pp. III–906–III–909, May 2003.
[4] OSCI, *SystemC User's Guide*. Open SystemC Initiative, 2002.
[5] L. Franzens, "Systemc for embedded system design," *University of Innsbruck, Institue for Informatics*, 2006.
[6] N. Ahmed, I. Aref, F. Rodriguez, and K. Elgaid, "Wireless channel model based on soc design methodology," *Fourth International Conference on Systems and Networks Communications (ICSNC)*, September 2009.
[7] A. Ahmad, *Data Communication Principles for Fixed and Wireless Networks*. 2003.
[8] B. A. Forouzan, *Data Communications and Networking*. McGraw-Hill Science/Engineering/Math, forth ed., 2006.
[9] W. Muller, W. Rosenstiel, and J. Ruf, *SystemC Methodologies and Applications*. Mar. 2003.
[10] W. H. Tranter, K. S. Shanmugan, T. S. Rappaport, and K. L. Kosbar, *Principles of Communication Systems Simulation with Wireless Applications*. first ed., 2004.
[11] B. Sklar, *Digital Communications Fundamentals and Applications*. Prentice Hall, 2003.
[12] HP, *HDLC Frame Protocol Users Guide*. HP invent, 1 ed., Feb. 2004.
[13] V. Bagad and I. Dhotre, *Data Communication and Networking*. Technical Publications Pune, Jan. 2006.
[14] A. Leon-Garcia and I. Widjaja, *Communication networks: fundamental concepts and key architectures*. McGraw Hill, 2004.
[15] B. Walke, S. Mongold, and L. Berlemann, *IEEE-802 Wireless Systems*. John Wiley and Sons, 2006.
[16] S. Iyengar, N. Parameshwaran, V. Phona, N. Balakrishnan, and C. Okoye, *Fundamentals of Sensor Network Programming - Applications and Technology*. Wiley, 2010.