Design, Implementation, and Applications of a B-ISDN Simulation Testbed*

Jin-Fu Chang
Shi-Chung Chang Zsehong Tsai† Jung-Shyr Wu‡ Hsu-Chun Yen

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC

‡Department of Electrical Engineering
National Central University
Chung-Li, Taiwan, ROC

Abstract

In this paper we describe the need and desired features for broadband ISDN (B-ISDN) simulation environments and present the design and implementation details of a simulation testbed prototype. The simulation testbed is designed to be with high flexibility in the network topology/traffic descriptions and provides a friendly graphical user interface. We also show that interesting simulation results can be easily observed via the use of this testbed.

1 Introduction

Broadband networks based on optical transmission have been a main stream of communication networks research and development (R&D). Future deployments of broadband networks are expected to have revolutionary impacts on the telecommunications industry. According to [3], B-ISDN is intended to "support switched, semi-permanent and permanent, point-to-point and point-to-multipoint connections and provides on demand, reserved and permanent services. Connections in B-ISDN support both circuit mode and packet mode services of a mono- and/or multi-media type and of a connectionless or connection-oriented nature and in a bidirectional or unidirectional configuration. A B-ISDN will contain intelligent capabilities for the purpose of providing advanced service characteristics, supporting powerful operation and maintenance tools, network control and management."

*This was supported in part by National Science Council of ROC under grant NSC 82-0416-E-008-083, NSC 82-0416-E-008-087, NSC 82-0416-E-008-089, NSC 82-0416-E-002-246, NSC 82-0416-E-002-248, NSC 82-0416-E-002-249.

†All correspondence should be sent to Zsehong Tsai, E-mail address: stsai@cc.ee.ntu.edu.tw

As broadband networks support a wide spectrum of services with different quality of service(QOS) requirements, many new challenges arise in network architecture design, high-speed protocol design, analysis of network configuration, paradigms for network resource control and allocation, design of critical network components, strategies for network operation, management and maintenance, traffic modeling, measurement and control, etc.

Various B-ISDN testbeds have also been built in the USA, Europe [13, 14] and in Japan [17]. Without going into testbeds of a large scale and a high level of fidelity, testbeds based on discrete-event computer simulation have long been developed as flexible platforms for network R&D, especially at the planning stage [9, 10]. In recent years, a number of network simulators (many of which are commercially available) have been developed in an attempt to ease the efforts involved in designing complicated networks. See, e.g., [1, 2, 4, 6, 7, 11, 12, 21]. To the best of our knowledge, with the exception of [4] and a sepcial version of [6] none of the above was specifically designed for simulating B-ISDNs. As a consequence, the existing network simulators, strictly speaking, lack the abilities to cope with various transmission, switching, and routing techniques which are unique to B-ISDN.

Challenges for computer simulation of B-ISDN

Commercially available network simulation software include BONeS[6], OPNET [11], Q+[12] and NETWORK II.5. There are also University developed packages such as Nest[7], MaRS[2], Ptolemy[21] and a distributed simulator developed by Choi and Gosh [4]. Many advantages and features of these existing network simulators are also desirable in simulating broadband networks. Distinct challenges for

1080-241X/95 $4.00 © 1995 IEEE

developing an effective broadband network simulator are summarised as follows.

(1) Powerful Network Description

To capture the variety and complexity involved in modeling broadband networks, a network description method should

- provide a language that is general enough and allows a user to describe all the aspects of broadband networks intuitively;
- allow a user to define model building blocks at various levels of details and facilitate repetitive use of existing building blocks;
- assist a user in describing complex networks systematically.

(2) Effective Simulation Methodology

In a broadband network, concurrent events occur in a much higher rate than a narrowband network. Many of the interested events such as buffer overflows and transmission errors take place with a probability at the order of 10^{-9} to 10^{-12}. Both the very large amount of events and the collection of rare event statistics demand fundamental advancement of simulation techniques [16, 8, 15] in random number generation, event generation, simulation control, and statistics collection, etc., for

- fast and accurate simulation of rare events
- full exploitation of parallel/distributed processing technology for speeding up large-scale simulations.

(3) Computational Efficiency

In addition to the simulation methodology itself, computer software techniques such as data structure, data base and its management, and algorithms and programming should be considered carefully from an integrated perspective to reduce the memory space requirement and to increase simulation speed.

(4) Freindly User Interfaces

As a broadband network simulator will be very complicated by the nature of simulating a B-ISDN, advanced techniques for developing user friendly interfaces are needed to make the simulator transparent to users, allow easy access to and manipulation of various simulator functions, facilitate flexible experimentations and measurements, and support tools for analysis and presentation of results.

(5) Flexibility

Flexibility is essential for adapting to the evolution of both the B-ISDN and the simulator itself. Challenges are how to realize design concepts such as modularised simulator development and the construction of a simulator development environment (or environment-based simulator).

In simulating broadband networks, more stringent requirements than those for the narrowband networks need to be met over all the above six categories. For example, the complexity of a broadband network demands a more powerful and flexible modeling language of the simulator without sacrificing its simplicity and friendliness to the users; as transactions or events occur in a much higher frequency in broadband networks, new simulation methodology and software techniques are needed to achieve acceptable computational efficiency. None of the existing network simulators satisfy all the broadband network requirements according to our assessment. There are therefore pressing needs for developing new platforms for simulating and prototyping broadband networks.

The remainder of this paper is organized as follows. In Section 2, we present the design methodology and software architecture of the simulation testbed. Simulation examples are provided in Section 3. Conclusions are provided in Section 4.

2 Design methodology and software architecture

The primary goal of our testbed is to provide an environment within which the interaction of various B-ISDN network elements, LAN, and MAN modules can be simulated, allowing the network designer to have a better understanding of what the actual network would behave even in the absence of such a network physically. To this end, the testbed provides cell level simulation and statistics measurement so as to allow the end-to-end performance of two network entities to be measured. The simulation testbed is versatile in that existing B-ISDN networks elements, such as ATM adaptation layer entities, ATM switches, ATM-MAN internetworking units, various traffic policing mechanisms, as well as new network elements

can be pieced together and then be simulated in a single simulation environment.

Figure 1 depicts the overall software architecture of the simulation testbed. To allow the reader to have a better feel for how the testbed functions, the remainder of this section is devoted to a brief discussion of each of the building blocks as well as their interconnections. For more details, the reader is referred to [18, 19].

Graphical user interface

An X-window-based *graphical user interface* (GUI) is provided to serve as a bridge between the user and the simulation testbed. The main purpose of the graphical user interface is to allow the user to create, delete, and alter a network as well as to start, pause, and stop a simulation with the help of a mouse on a point-and-click basis. Figure 2 displays typical windows of our B-ISDN testbed.

The windows shown in Figure 2 are called the *network editing window* on which the graphical representation of the network to be simulated is displayed. Our user interface features a hierarchical way of viewing a network, which allows the user to organize the network to be simulated at the desired level of detail. The concept of a *subnet component* is central to such a hierarchical fashion of design. For example, clicking the icon of a subnet object will result in the appearance of a subwindow, displying the details of the clicked subnet object. Furthermore, subnets can be nested. Using such a hierarchical user interface, the user can design a network in either a top-down or a bottom-up fashion, and at each level, irrelevant portion of the network can be abstracted out by using subnet objects.

Graph language

The network created by the user interface is stored in the form of a *graph language*, which describes network objects together with their attributes. (See [18] for the detailed syntax of the graph language.) A graph language is in existence in the form of a plain text; as a result, it can be edited by invoking the user interface (which will then redisplay the network graph associated with the language) or by using standard text editors such as 'vi', 'emacs', among others. With the help of an object-based description scheme and a well-defined graph language, a *consistency checker* is implemented in the testbed which facilitates the check of inconsistency in the description of the underlying network.

Software Architecture

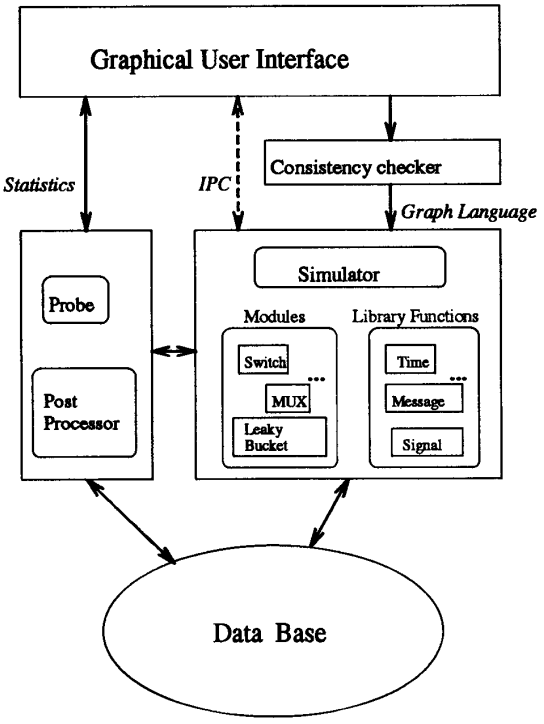

Figure 1: Software architecture of the B-ISDN simulation testbed.

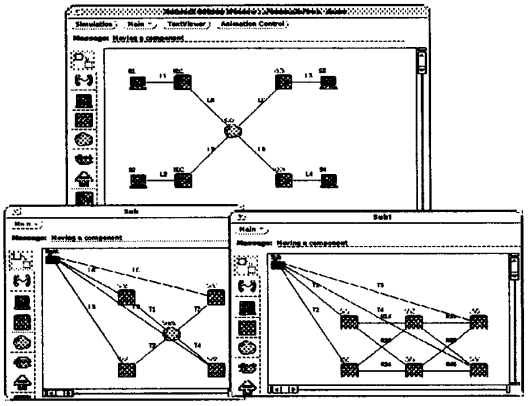

Figure 2: Windows of the graphical user interface.

Modular approach

The concept of an *object* plays an important role in the design of our simulation testbed. All important resources in the testbed are modeled as objects. For example, *network objects*, characterized by their *attributes*, define the abstract representations of network elements being simulated in the environment. Aside from network objects, a number of supplementary objects, such as *probes* which serve the purpose of collecting simulation statistics, are also in existence in the testbed.

An *object class* consists of a group of objects with similar attributes. An object class is said to be the subclass of another object class, called *superclass*, if all the attributes of the superclass are inherited. The interested reader is referred to [18] for the detailed containment relationships among objects. As the concept of object-orientation is gaining popularity in the computer and communication communities, our object-based approach thereby provides a good starting point for future extension in that direction.

From the implementation point of view, an important source of flexibility of our B-ISDN simulation testbed comes from the *modular* programming approach. In our setting, a network component comprises one or more *modules*, each of which is characterized by an *initial function*, a *node function*, and a *signal function*. See Figure 1. The initial function provides the necessary initialization of the associated module before the simulation starts; the node function carries out the actual simulation; the signal function allows the module to communicate with internal modules (i.e., modules within the same component) or with external network components.

The modular approach of our design facilitates the easy creation of a network. By selecting the appropriate component modules and connecting them accordingly via the graphical user interface, one can easily piece various components of a complicated network together. By clicking the corresponding icon of a network component, one can then provide necessary parameters which are necessary for the simulation to proceed. Aside from the above advantage, our modular environment allows a new network component to be integrated into the testbed relatively easily.

Simulation module

Our underlying simulation technique is based upon the so-called *discrete-event simulation* mechanism, which, in comparison with other simulation techniques, offers the greatest flexibility. As its name suggests, in discrete-event simulation each network component is abstracted as a sequence of events, each of which updates the state variables associated with the component as well as the simulated time. As a result, it is relatively easy to modify and expand the simulation environment. For example, adding a new component is as simple as defining a set of state variables together with a set of events describing the behavior of the component. Another advantage of using discrete-event simulation is that *parallelized* discrete-event simulation has been gaining its momentum at a rapid pace during the past decade. Hence, the choice of discrete-event simulation (over other simulation techniques) offers another dimension of flexibility as far as transplanting our B-ISDN testbed to a parallel environment is concerned. More will be said about the scheduler later in this paper.

To control the progress of a simulation, the user can utilize a window called *simulation window*. The file in which the network is stored (in the form of a graph language) as well as the file recording the simulation results are specified in such a window. Information concerning the simulation itself (such as the total simulation time, the random seed used in the simulation, and the number of sample points) must be provided by the user before the simulation begins. During the course of a simulation, the user can pause, change the original simulation setting, examine intermediate statistical results, and then continue the simulation by clicking the respective buttons on the window. Upon the completion of a simulation, the user can invoke a *post processing* facility by clicking the STATISTICS button. At the current phase of our testbed design, statistical data can be tailored to the format of of MATLAB (which is widely available commercially) or GNUPLOT, which can be invoked via the postprocessor of the testbed.

Scheduler mechanism

To support the necessary timing requirements, the testbed provides a variety of library functions using which individual modules can coordinate their executions with the event scheduler. Basically, during the course of the simulation control is passed between the event scheduler and the simulated modules when the simulation library functions are used. The typical condition is that when a library function is called, the order of the event list need to be rearranged and thus the execution of a particular module is suspended until the module is back to the first position in the event list.

The simulation library functions are divided into categories, namely, *time*-related functions,

communication-related functions, and statistical functions. The former include *current-time()*, *advance()*, *wait-for-message()*, and *slot-sync()*, whereas the second include *recvmsg()*, *sendmsg()*, and *signal-to()*. The third category is for statistical measurement and will be discussed later. (For the interested reader, the detailed description of each of the above functions can be found in [18].)

Among these functions, the slot-sync() function is especially important since it is essential for converting a continuous-time cell arrival process into discrete time processes, as it would actually happen in the time-slotted ATM network. For example, the generation of AAL-PDUs by an ATM-LAN gateway might be modeled as certain continuous-time processes, such as Poisson process, etc. Therefore, it is natural to generate of ATM layer PDUs at the gateway on a non-slotted fashion. As shown in Figure 3 a conversion module which provides slot synchronization is thus desired for modeling ATM transmission and is simple to implement.

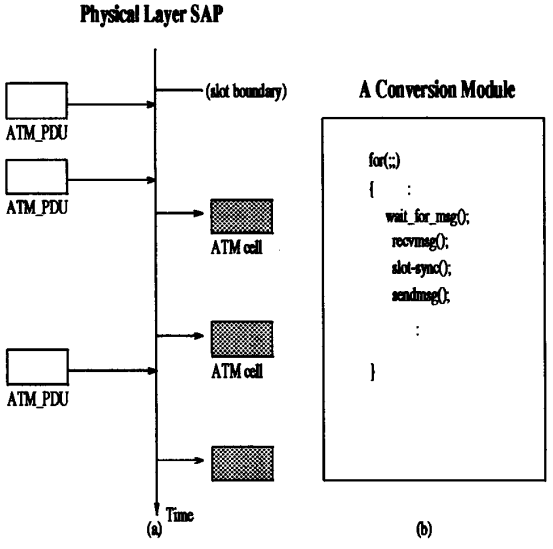

Figure 3: Timing diagram and a conversion module for slot synchronization.

Statistical measurement

To measure the traffic characteristics of network components in the simulation environment, two approaches are provided in the testbed: using the *probes* or via statistical functions. The so-called *probes* is allowed to be attached to network objects by the user so that standard performance measures, such as the number of cell arrivals, departures, and losses, of the probed network objects can be recorded and then analyzed. Based on their functionalities, three types of probes, namely, *component-wise*, *connection-wise*, and *link-wise* probes, are defined and built in the testbed scheduler. Component-wise probes are used to measure the incoming and outgoing traffic data of the probed components, such as multiplexers and switches. Connection-wise probes, on the other hand, allow the end-to-end traffic flow between two peer sources to be measured. Finally, link-wise probes are used for measuring the utilizations of probed links. All statistics are collected by the scheduler while the corresponding simulation functions are executed. The second approach requires the statistical functions to be directly coded within the module so that special performance measures can be collected. This feature provide the users with the maximal flexibility in the performance measurement.

3 Examples of applications

Hybrid simulations

Three types of techniques are allowed for generating cell traffic patterns in this simulation testbed: 1)using a simple mathematical model, 2)employing empirical data/trace files, or 3)simulating a detailed simulation model within a source module. Although these techniques actually suggest different applications of the testbed, as explained below, our testbed allows the simultaneous use of all.

In the first approach, mathematic models that are popular in ATM network performance analysis, such Bernoulli process, Switched Bernoulli process, Markov-Modulated Poisson process, and ON-OFF sources etc. can be used as source modules. While using this approach, since the upper layer applications represented by the cell pattern is irrelevant to the testbed, the cell data structure is in the form of *dummy cell*. That is, a cell without payload is passed among network components during the simulation.

When empirical data/trace files are employed, the 48-octect ATM cell information field can be loaded with actual data, after appropriate ATM Adaptation Layer segmentation procedure is executed in the source at the pre-determined time, which could be recorded in a trace file. For example, if AAL Type 1 is chosen, there will be one octect reserved in the ATM cell data structure for the Sequence Number and Sequence Number Protect and the remaining 47 octect

in the information field will be used to carry empirical data, such as PCM-coded voice samples. If the virtual channel connection indeed represents a point-to-point voice call, the module at the receiving end could reassemble the received information, which is subject to simulated cell losses within the network, and save it as a file for further study. In other words, our testbed provides users an environment to directly observe the impact of network congestions on upper layer applications, including voice, video, etc. One might notice that the cell data structure used in this approach should require more memory than the dummy cell. When the modeled virtual channel connections have large delay-bandwidth product, memory requirement of the simulation testbed might become significant, relative to the physical memory available on the computer. Based on such considerations, we employ the concept of ***Hybrid Simulation*** by allowing the mixed use of dummy cells and cells with empirical data simultaneously within the testbed, and suggest the user only to use minimal amount of empirical data.

In certain cases, the ATM cells are generated by an end system which itself requires detailed modeling and cannot be replaced by a simple mathematical model. For example, an LAN-ATM gateway or an MAN-ATM gateway can be included as one module in the testbed and its gateway function or the MAN/ATM behaviour may be of interests. Then the aforementioned detailed modeling approach is suggested. Here, the code for the simulation of the end system is directly included within a module so that the cells are generated as observed on a simulator dedicated to that LAN-ATM or LAN-MAN gateway. Certainly the cell generating pattern should be much closer to reality than the simple mathematical model. Since the use of this approach requires much more CPU time than the first two techniques, we still suggest to keep the number of such modules at the minimal level and mix it with other traffic sources.

Subjective evaluation of QOS

Video and voice services are important real time applications that B-ISDN should support. Although a quantitative analysis is essential for evaluating a network design, a qualitative analysis based on subjective measures of users is as important for evaluating the quality of real-time services provided by the network.

Figure 4 shows the details of the source/sink modules for simulating the H.261 video service, which include, at the source end, an empirical RGB image file, RGB to YUV conversion, a H.261 encoder, a simple type-3/4 AAL for video data segmentation and sequence number assignment and a simple ATM layer module, and, at the receiving end, a simple type-3/4 AAL for cell reassembly, loss detection and error concealment, a H.261 decoder, and a video player. In segmenting the H.261 data, a macroblock of a picture frame from the H.261 encoder serves as the basic unit. Temporal replacement is chosen as the error concealment scheme for its simplicity, where a lost macroblock (due to cell losses) in one received frame is replaced by the macroblock at the same corresponding position of the preceding frame.

When the traffic intensity of the interfering cell stream is set to zero, all the 2171 cells of H.261 video data (30 frames) are received in simulating one-second of the network. One of the completely received frame is given in Figure 5. Figure 6 demonstrates the same frame with cell loss but no error concealment after the traffic intensity of the Bernoulli source is increased to 0.2. A total of 319 cell losses occurs in this case, which amounts to a very high cell loss probability of 0.14 . Note that cell losses only affect macroblocks locally instead of the whole frame as can be clearly seen in Figure 6. This demonstrates one aspect the effects of the segmentation and reassembly scheme at the AAL. When the error concealment function is added, the received frame quality (Figure 4) is much improved, although still visually unsatisfactory due to heavy cell losses.

The above example illustrates the capacity of the testbed for conducting hybrid simulation involving both empirical and mathematical sources and for providing an environment for subjective evaluation of QOS.

4 Conclusions and future directions

We have designed and implemented a Broadband ISDN simulation testbed which features user-friendliness and flexibility. This simulation testbed prototype should fit the unique characteristics of Broadband ISDN and is able to expedite the description process of the network topology/traffic conditions prior to the simulation. Testing results show that current version of simulation testbed is reliable and can be readily used for network design purposes.

We believe our network simulation testbed can be even more flexible and easy to use when more network modules such as various LANs, MANs, and ATM/AAL modules are developed. However, this work depends on the need of simulation and can be separated from the testbed development.

Although efficient simulation technique such as important sampling is not yet included in this simulation

testbed, we believe this should be the most promising future direction.

Acknowledgement

The authors are grateful to Tzi-Dar Chiueh for his valuable comments during the definition and implementation of this system. The authors also would like to thank the B-ISDN Simulation Testbed team members, especially C.T. Chang, B.L. Chen, W.S. Chen, W.W. Chuang, L.T. Hsu, Y.S. Hsu, F.S. Jou, M.C. Shu, R. Tsai, T.Y. Tung, C.M. Wang, and L.H. Wang.

References

[1] C. Alaettinoglu, A. Shankar, K. Dussa-Zieger, and I. Matta, *Design and Implementation of MaRS: A Routing Testbed,* Tech. Rep., CS-TR-2964, Univ. of Maryland, 1992.

[2] C. Alaettinoglu, K. Dussa-Zieger, I. Matta, O. Gudmundsson, A. U. Shankar, MaRS (Maryland Routing Simulator) - Version 1.0 Programmer's Manual, UMIACS-TR-91-107 and CS-TR-2723, Univ. of Maryland, July 1, 1991.

[3] CCITT:Recommendation I.121, "Broadband Aspect of ISDN," 1991.

[4] A. Chai, S. Ghosh, "Modeling and Distributed Simulation of a Broadband-ISDN Network, " *IEEE Computer*, Vol. 26, No. 9, Sep. 1993, pp. 37-51.

[5] S.-C. Chang, "Needs and Desired Features of a Broadband Network Simulation Systems," *1993 Workshop on Computer Communication Networks*, Taipei, Feb. 1993, pp.168-175.

[6] COMDISCO System Inc., *Block Oriented Network Simulator (BONes)*, 1991.

[7] A. Dupuy and J. Schwartz, *Nest System Overview*, Tech. Report, Computer Science Dept., Columbia University, New York, March 1988.

[8] R. Fujimoto, "Parallel Discrete Event Simulation," *Comm. of ACM*, 33(10), 1990, pp. 31-53.

[9] *IEEE Journal on Selected Areas in Communications*, Special Issue on Computer-Aided Modeling and Analysis of Communication Systems, January, 1984.

[10] *IEEE Journal on Selected Areas in Communications*, Special Issue on Computer-Aided Modeling and Analysis of Communication Systems, January, 1988.

[11] MIL 3, Inc., "OPNET M Version Technical Overview," Washington, DC, 1989.

[12] B. Melamed, "Q+ User Guide & Reference Manual," AT&T Bell Laboratories, NJ, 1989.

[13] C. Mossoto, "Pathways for Telecommunications: A European Outlook," *IEEE Comm. Magazine*, Vol. 31, No. 8, Aug. 1993, pp. 52-58.

[14] H. Ricke, J. Kanzow, (ed.) "BERKOM - Broadband Communication within the Optical Fibre Network," R. V. Decker, Heidelberg, 1992.

[15] R. Righter, J. C. Walrand, "Distributed Simulation of Discrete Event Systems," *IEEE Proceedings*, Vol. 77, No. 1, 1989, pp.99-113.

[16] R. Rubinstein, *Monte Carlo Optimization, Simulation and Sensitivity of Queueing Networks*, John Wiley & Sons, 1986.

[17] S. Sakata, "B-ISDN Multimedia Workstation Architecture," *IEEE Comm. Magazine*, Vol. 31, No. 8, Aug. 1993, pp. 64-67.

[18] *Broadband ISDN Simulation Testbed - Reference Manual* (Version 1.0), Tech. Report, Dept. of Electrical Engineering, National Taiwan University, April 1994.

[19] *Broadband ISDN Simulation Testbed - User's Guide* (Version 1.0), Tech. Report, Dept. of Electrical Engineering, National Taiwan University, April 1994.

[20] "Gigabit Network Testbeds," *IEEE Computer Magazine*, September 1990, pp.77-80.

[21] *The Almagest: Manual for Ptolemy (Version 0.3)*, Dept. of Electr. Engrg. and Comp. Sci., University of California at Berkeley, Berkeley, 1991.

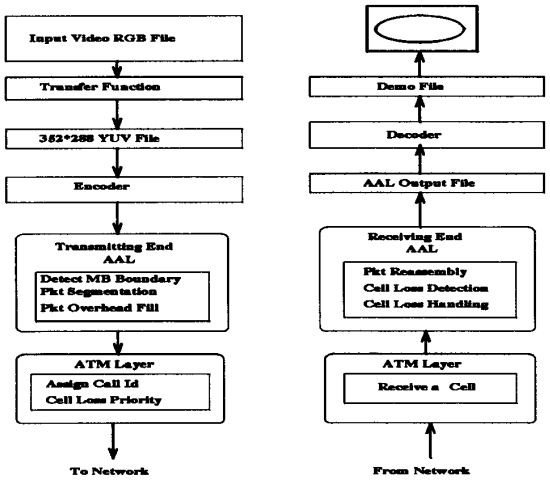

Figure 4: H.261 source/sink simulation modules.

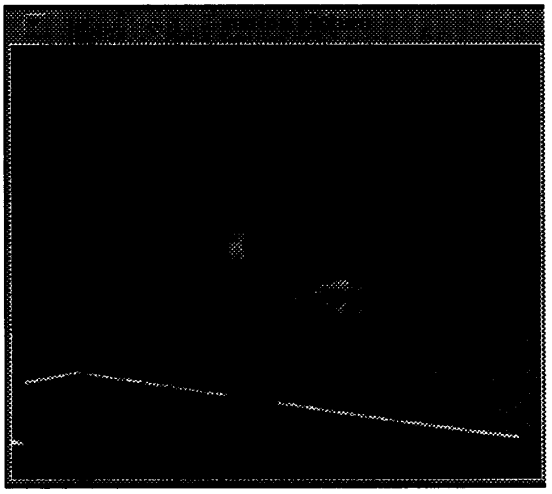

Figure 6: A frame with cell losses but no error concealment.

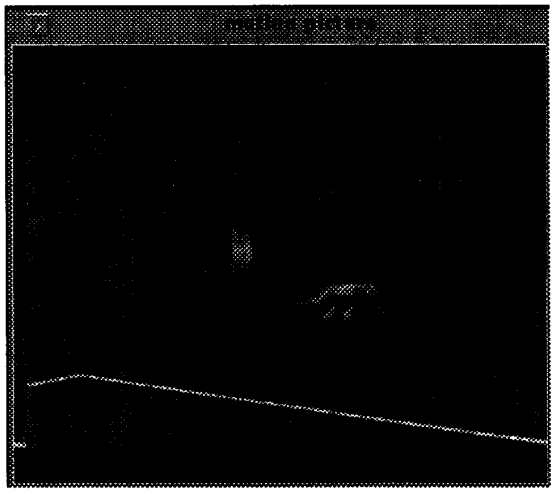

Figure 5: A completely received frame.

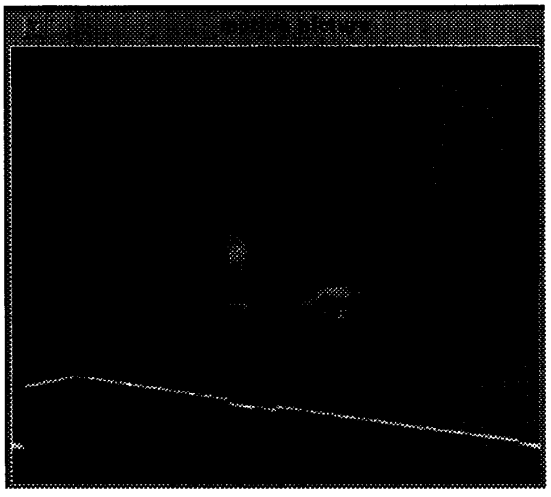

Figure 7: A frame with cell loss and error concealment.