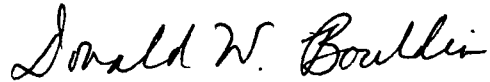


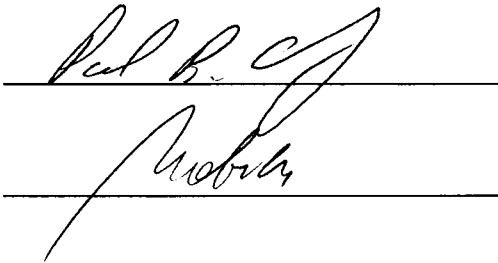
To the Graduate Council:

I am submitting herewith a thesis written by Badhri Narayanan Uppiliappan entitled "Behavioral Modeling for System level Simulation and Verification of a Microelectronic System Controller." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.



Donald W. Bouldin, Major Professor

We have read this thesis
and recommend its acceptance:



Accepted for the Council:



Associate Vice Chancellor
and Dean of The Graduate School

Behavioral Modeling for System Level Simulation and Verification of a Microelectronic System Controller

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Badhri Uppiliappan
December 1998

DEDICATION

This thesis is dedicated to my father

Late Thiru. K. Uppiliappan,
(திரு. உப்பிலியப்பன்)

my mother Thirumathi Vasundhara Uppiliappan,

(திருமதி. வசுந்தரா உப்பிலியப்பன்)

For their inspiration, guidance and understanding,

my brother Bharath Uppiliappan

for his support and encouragement, and

to my pet dog Late Dennie.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank the large number of people who have helped me throughout this program towards the completion of this degree.

First, I am greatly indebted to the U.T. - O.R.N.L. Joint Program in Mixed Signal VLSI and Monolithic Sensors for providing me the opportunity to pursue graduate studies at UTK. Special thanks are due to Drs. James Rochelle, Michael Simpson and Alan Wintenberg. My gratitude to Bill Bryan and Mark Musrock of the ORNL Instrumentation and Controls division at the Oak Ridge National Laboratory are also due.

I would also like to thank my committee members, Drs. Donald Bouldin, Paul Crilly and Miljko Bobrek for their supportive nature and rallying with me during the last days of the preparation of the thesis and defense. Without them, this would not have completed.

Special thanks are due to Lloyd Clonts for his help with CAD tools and advice on methodology. I am greatly obliged to Scott Puckett for his friendship and advice on a wide variety of topics.

I am grateful to the numerous friends and acquaintances who have made my experience at UTK a special one and whom I shall cherish forever. They have truly helped me in a million ways for which I shall remain indebted to them forever.

Last, but not the least, my gratitude to my family for all the support, encouragement and patience.

ABSTRACT

System level simulation and verification help in obtaining information necessary to evaluate and verify new designs and implementations. The use of behavioral modeling helps in obtaining these results with minimal simulation time and resources. Use of the simulation data can provide guidance during prototype production and debugging. The creation and capabilities of a behavioral model for a GHz transmitter and receiver chips are presented. The working of the model under different link configurations was also verified. The models were tested with a mode bit set for both the 16-bit and 20-bit parallel words. The methodology for importing FPGA designs from the Foundation Series to ViewLogic environment for board level simulation is also presented. The Xilinx FPGA designs, ALM FPGA and Heap Manager FPGA, were successfully imported into the ViewLogic Powerview (v6.0) CAD tool with post-layout timing information included in the netlist. The Heap manager FPGA was tested in the Foundation Series simulator. The design achieved its performance requirements of a minimum of 40MHz working frequency. The steps towards creating a mixed signal simulation model are also explored by simulating an analog behavior model.

TABLE OF CONTENTS

Chapter	Title	Page
1	Introduction And Background	1
1.1	Modeling of Systems	1
1.2	Concept Abstraction	1
1.3	Concept Evaluation	2
1.4	Concept Objectives	3
1.5	Targets of Concept Development	3
2	Software, Design Flow And System Selection	4
2.1	Multiple Modeling Notations	4
2.2	Selection and Evaluation of the System to be Modeled	5
2.3	Objectives of the System Modeling Effort	7
2.4	Functionality of the Heap Manager Board	7
2.5	FPGAS	8
2.6	Description of XC4000E and XC4000XL Architecture differences	14
2.7	Possible Solutions: Implementable and Unimplementable	15
2.8	Design and Software Flow	15
2.9	Expected Results	22
3	FPGA Designs	24
3.1	Functional Description of Address list Manager FPGA	24
3.2	Functional Description of the Heap Manager FPGA	24
4	GLINK Chipset Model: Requirements and Implementation	29
4.1	Description of GLINK Receiver chip	29
4.2	Model for the Receiver chip	34
4.3	Simulation results for the Receiver Model	37
4.4	Modeling the GLINK Transmitter	40
4.5	Transmitter Model Implementation	44
4.6	Simulation results for the Transmitter Model	47
4.7	Model for the Phase Locked Loop	49

5	Simulation Integration, Results and Comparison	51
5.1	Integration Flow for FPGA Models and GLINK models	51
5.2	Simulation of GLINK Chips: Configurations	52
5.3	Analog modules of the Chips	58
5.4	Board Level Simulation	58
6	Summary, Conclusions and Future Work	62
6.1	FPGA Modeling	62
6.2	GLINK Models	62
6.3	Scope for Future Work	64
	References	67
	Appendix	69
	Vita	80

LIST OF FIGURES

FIGURE 1 - HEAP MANAGER BOARD DIAGRAM	9
FIGURE 2 - XC4000 CLB DIAGRAM.....	11
FIGURE 3 - DOMAINS OF DESCRIPTION – GAJSKI KUHN DIAGRAM.....	17
FIGURE 4 - DESIGN PROCESS FLOW.....	18
FIGURE 5 - XILINX FOUNDATION SERIES DESIGN SUITE.....	20
FIGURE 6 - VIEWLOGIC MIXED SIGNAL DESIGN FLOW	21
FIGURE 7 - BLOCK DIAGRAM OF ADDRESS LIST MANAGER FPGA DESIGN.....	25
FIGURE 8 - FLOW DIAGRAM FOR THE CHANNEL READOUT SEQUENCE OF THE HEAP MANAGER FPGA DESIGN	27
FIGURE 9 - SIMULATION OVERVIEW: HEAP MANAGER STATE MACHINE	28
FIGURE 10 - GLINK RECEIVER BLOCK DIAGRAM.....	30
FIGURE 11 - RECEIVER BLOCK TIMING DIAGRAM.....	33
FIGURE 12 - IMPLEMENTATION OF THE RECEIVER MODULE	38
FIGURE 13 - OVERALL VIEW OF A SIMULATION OF THE IMPLEMENTED GLINK RECEIVER.	39
FIGURE 14 - BLOCK DIAGRAM OF GLINK TRANSMITTER CHIP.....	41
FIGURE 15 - TIMING CHARACTERISTICS OF TRANSMITTER BLOCK	43
FIGURE 16 - TRANSMITTER CHIP IMPLEMENTATION.....	46
FIGURE 17 - OVERALL VIEW OF A SIMULATION OF THE IMPLEMENTED GLINK TRANSMITTER CHIP.....	48
FIGURE 18 - BLOCK DIAGRAM OF A BASIC PHASE LOCKED LOOP CIRCUIT.....	50
FIGURE 19 - OVERALL SIMULATION OF SIMPLEX 1 LINK CONFIGURATION WITH LOW-SPEED RETURN PATH.	53
FIGURE 20 - SAMPLE ZOOM VIEW OF THE SIMPLEX 1 LINK CONFIGURATION WITH LOW-SPEED RETURN PATH.....	54
FIGURE 21 - OVERALL VIEW OF SIMULATION OF SIMPLEX CONNECTION WITH PERIODIC SYNCH PULSES.....	56
FIGURE 22 - SAMPLE ZOOMED VIEW OF SIMULATION OF SIMPLEX LINK CONFIGURATION WITH PERIODIC SYNCH PULSES.	57
FIGURE 23 - PLL SIMULATION DATA - BEFORE LOCKING TO INPUT FREQUENCY.....	59
FIGURE 24 - PLL SIMULATION - JUST BEFORE LOCKING BEGINS FOR THE DYNAMIC RANGE	60
FIGURE 25 - PLL SIMULATION - DURING LOCKED CONDITION.....	61

Chapter 1

Introduction and Background

1.1 Modeling of Systems

Modeling of systems is generally undertaken to evaluate possible architectures for a new system, evaluation of implementation styles and similar needs. The modeling results can be of very many types - ranging, from simple functional information, to even data on power consumption, area, speed and so forth. These data can be collected depending upon the extent of modeling for the concerned system and the time and resources available to evaluate these features.

System level modeling is achieved through a combination of individual component behavior modeling and some higher abstraction of specifications using abstract data types, records and other constructs.

Typically, system-modeling limits endeavors to functional verification of a possible system protocol and its implementation. However, it is not uncommon, to include some timing information, if known or assumed, to give a more realistic system analysis.

1.2 Concept Abstraction

System level simulation provides data that can be used to study the general functionality of a proposed system. Simulations at this level depend upon high level models, either developed or pre-existing, to be put together and simulated. The development of models for custom static design takes the most time and resources, since detailed specifications are available.

This relatively high level of simulation provides for faster design refinements and debugging. Also, the model development effort provides an opportunity to study various issues like CAD flow for mixed-signal models and integration of designs from multiple CAD tools. The ease of inclusion of models for designs that changed very often can also be investigated.

1.3 Concept Evaluation

Modeling a system at board level with different types of components is a challenge. The primary reason being the need for making decisions on the depth of modeling and investment of resources and available tools to achieve it with minimal time. These decisions are based upon a number of criteria or inputs like available CAD tools, supported libraries, dynamics of system specifications and human and machine resources available. The ability to modify or incorporate CAD flow branches during the process of modeling a system alleviates resolution of multiple design format conflicts. These issues are to be explored during the process of modeling the system and its components. Specific implementation or modeling issues or styles were to be explored at the component level. Methods to convert designs from one platform to another were to be explored. Timing issues were to be considered to effect good models but were not to be modeled in great detail.

CAD tools in the Microsoft Windows™, SUN OpenWindows™ and Solaris™ environments were to be evaluated for suitability to implement the system design modeling effort.

1.4 Concept Objectives

The system modeling effort aims at **development of validated model(s)** for the components. The level of validation necessary is proportional to the accuracy required in estimating system performance. The user incorporates these models in the system design to evaluate functionality.

The purpose of the modeling effort is to

- ◆ Develop models with minimal usage constraints
- ◆ Allow the user to modify some model parameters easily
- ◆ Automate and / or document the necessary CAD tools and utilities to integrate the model in the simulation network.

The system to be modeled had three major components (i.e. two FPGAs and a GHz receiver). These were supported by peripheral and glue logic. The development of business oriented industry standard models was not the objective.

1.5 Targets of concept development

Models of reasonable validity for the components would be developed and made available for the system simulation network. Integration flow of multi-format designs into the selected system-modeling environment would be made available. Also, it was desirable to identify possible improvements to the model capabilities to deliver consistent performance and additional data that might help future use of the models in a different system design. The steps to develop a mixed signal model to visualize the analog components on the board were to be researched. Some of the targets were to be explored and dealt with only if resources permitted to develop them into deliverable solutions.

Chapter 2

Background: Software, Design Flow and System Selection

2.1 Multiple Modeling Notations

Behavioral level modeling is the highest level of abstraction at which a system can be modeled for functional verification of a particular system or component. Behavioral modeling refers to the substitution of more abstract, less computationally intensive circuit models for lower level descriptions of analog / digital functions. These simpler models emulate the transfer characteristics of the circuit elements that they replace, but with increased efficiency, leading to substantial reduction in the actual simulation time per circuit. This reduction in elapsed time per simulation, considering the whole design and simulation cycle, can lead to a tremendous increase in design efficiency, as well as, possible reduction in the time necessary to take a design from concept to a marketable product.

Behavioral modeling of systems is typically done in high-level description languages such as VHDL and Verilog. Sometimes, the programming language, C, is also used but more for algorithm or protocol verification. This level is characterized by processes, packages, sequential and concurrent statements.

Structural modeling (using VHDL) can be appropriate at any level of design. For example, test benches of completed components are often described using structural VHDL. Furthermore, structural VHDL supports the use of libraries and component reuse. This method enables choosing entities and appropriate architectures for the component from the libraries and configuring them appropriately with parameters for the component.

Gate Level modeling is more extensive, time and resource consuming and gives results that are more detailed and incorporate broad timing issues for consideration. In this form, the inputs and outputs of each gate are calculated and propagated through subsequent gates. The output levels usually are a minimum of four states which include 1, 0, X and Z. The number of nodes considered increases as a composite of the input and output nodes excluding common output and input nodes.

Transistor level modeling and simulation is the most time and resource consuming verification of the functionality and timing for a system model. This is a direct result of the high accuracy levels, large number of parameters considered and number of nodes to be evaluated. Solving the embedded or derived equations for modeling and producing the results costs computer time and resources that can hardly be tolerated for system level studies.

Thus, for quick estimates on the functionality and timing requirements, the use of behavioral level modeling and simulation results are best. It requires the least amount of resources, yet, produces sufficient data to compare architectures, system size, speed estimates and functionality checks.

2.2 Selection and Evaluation of the System to be Modeled

The heap manager card contained two FPGA designs and a static design in the form of a GHz Receiver chip. This provided an opportunity to explore the concept of system modeling to evaluate the functionality of the board design. This test system was of reasonable size to model and develop a methodology to integrate successfully the models into a system level simulation effort.

However, since the receiver chip was actually a part of a chip set which included a transmitter chip too, a model for that too was considered a befitting and logical supplement to the model development effort.

The other components were not modeled since they were in no significant way going to affect the functionality of the board and the performance in terms of delays could be modeled as line delays in the inter-component connections, if needed.

For the FPGAs, it was decided to work with the implemented design that was available at the beginning of the modeling effort. This was so because FPGA design simulation flow would be similar even if the design changed. The FPGA design work, needed to be done on a new set of design tools (Xilinx Foundation Series) since the old tools (ProSeries) could not support some of the new families of devices.

Also, amongst other considerations were the methodology, CAD tools and libraries needed to bring together the simulation netlists of the FPGAs and the GLINK receiver chip model. Since, the two design workspaces were different in the different tools, the intermediate formats for netlist conversion, export-import and re-netlisting were to be figured out.

As these problems or needs became obvious, it was time to dig in a little more into the requirements of the transmitter and receiver models. It was found that there was a need for some analog subsystems in the form of a Phase Locked Loop (PLL) and clock recovery circuit.

2.3 Objectives of the System Modeling Effort

The main objective of the modeling effort was to be able to generate a good model for the GLINK Tx and Rx chips and to integrate successfully the models of the FPGAs. The simulation at the board level was to be performed and functionality verified and to some extent timing specifications and performance limits. There is, however, no attempt to build a simulation efficient model, i.e. to build a model that uses simulation resources optimally.

A secondary objective was to investigate the possible techniques of successfully integrating a mixed-signal simulation model for the chipset, after possibly modeling the analog subsystems.

2.4 Functionality of the Heap Manager Board

The Heap Manager daughter card is part of the Front End Electronics Module (FEM) being developed for the EMCAL sub project of the PHENIX project. The electronics for the PHENIX project is being developed by the Instrumentation and Controls Division of the Oak Ridge National Laboratory. The Heap Manager sits in the backplane of a FEM with nine other such daughter cards. The primary function of the heap manager is to control and coordinate the activities of the other nine daughter cards to enable sequenced and timed data readout operations.

There are three main components of the Heap Manager Board. These are as follows:

1. GLINK receiver chip.
2. Address List Manager FPGA – An 84 pin XC4005XL family device implementation.
3. Heap Manager FPGA – A 160 pin XC4010XL family device implementation.

Some of the other components in the Heap manager board are ECL to TTL and TTL to ECL logic compatible components and buffers. These help the received data to be properly handled in terms of logic levels.

The GLINK receiver chip is the point of entry of the data to be acted upon by the heap manager board. Data arrives in serial form to be converted to a 20-bit parallel word with error checking performed, as appropriate, to ensure data integrity. Once data bits are extracted properly, they are buffered and acted upon. The Address List Manager is, as the name suggests, is an address space manager of a FIFO buffer. The Heap Manager logic handles the incoming data appropriately depending upon the control signals extracted from the data and received through other inputs to the board. Shown in Figure 1 is a representative Heap Manager board block diagram. The GLINK transmitter chip is external to the board. It is included in the board representation since it is a logical complement to the GLINK Rx chip. The buffer and logic converter components are not modeled since these would only affect delays and not the board functionality.

2.5 FPGAS

The *Field Programmable gate array* (FPGA) has been in use for more than a decade. An FPGA consists of an array of functional blocks along with an interconnection network, and as the name implies, its configuration can be determined in the field. The specific function of each block and the connections between blocks are prescribed by the user.

In the following pages, a brief overview of the functional description of Xilinx FPGAs is presented [1].

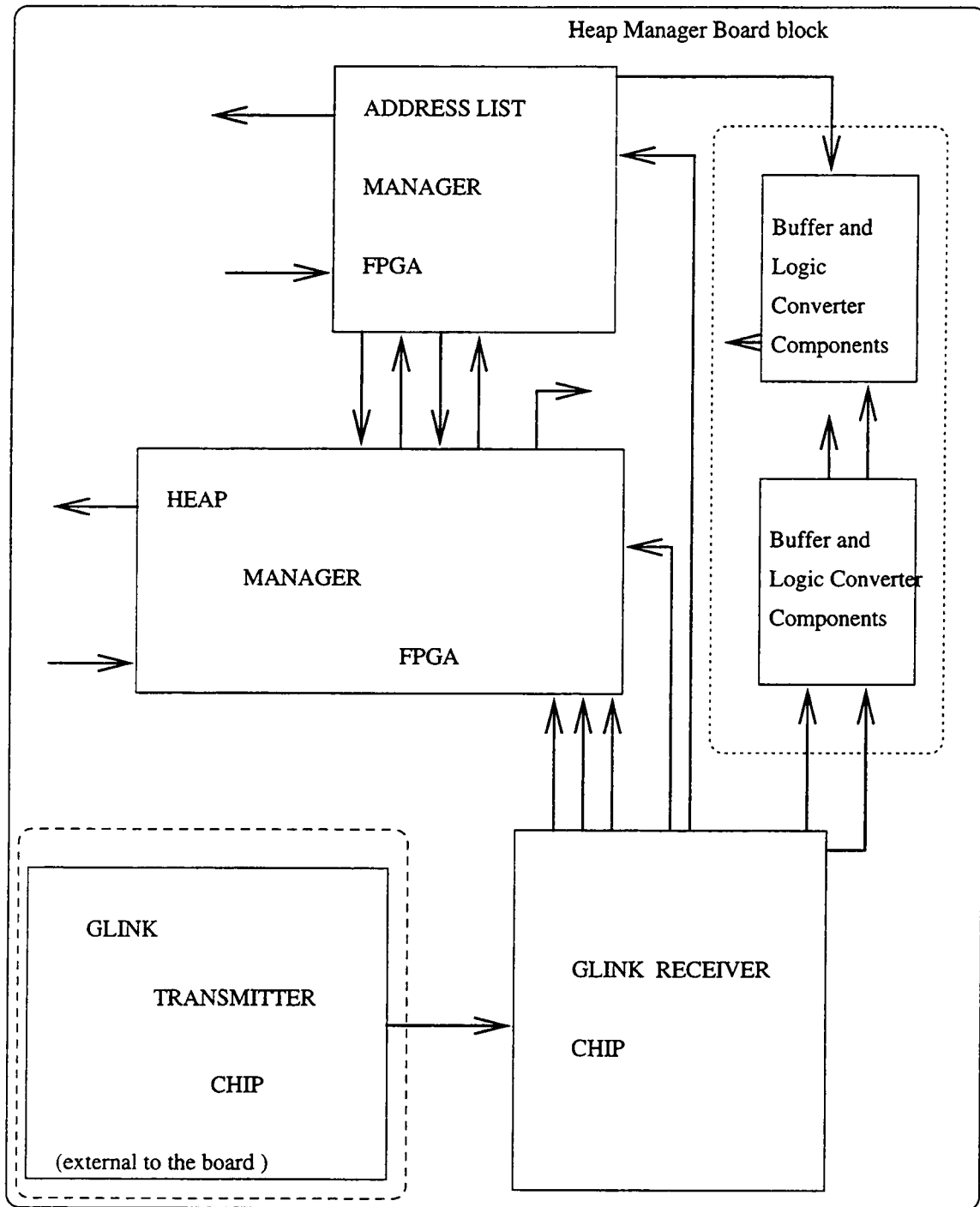


Figure 1 - Heap Manager Board Diagram

Basic building blocks: Xilinx user-programmable gate arrays include two major configurable elements; configurable logic blocks (CLBs) and input/output blocks (IOBs).

- ◆ CLBs provide the functional elements for constructing the users' logic
- ◆ IOBs provide the interface between the package pins and internal signal lines.

Three other types of circuits are also available:

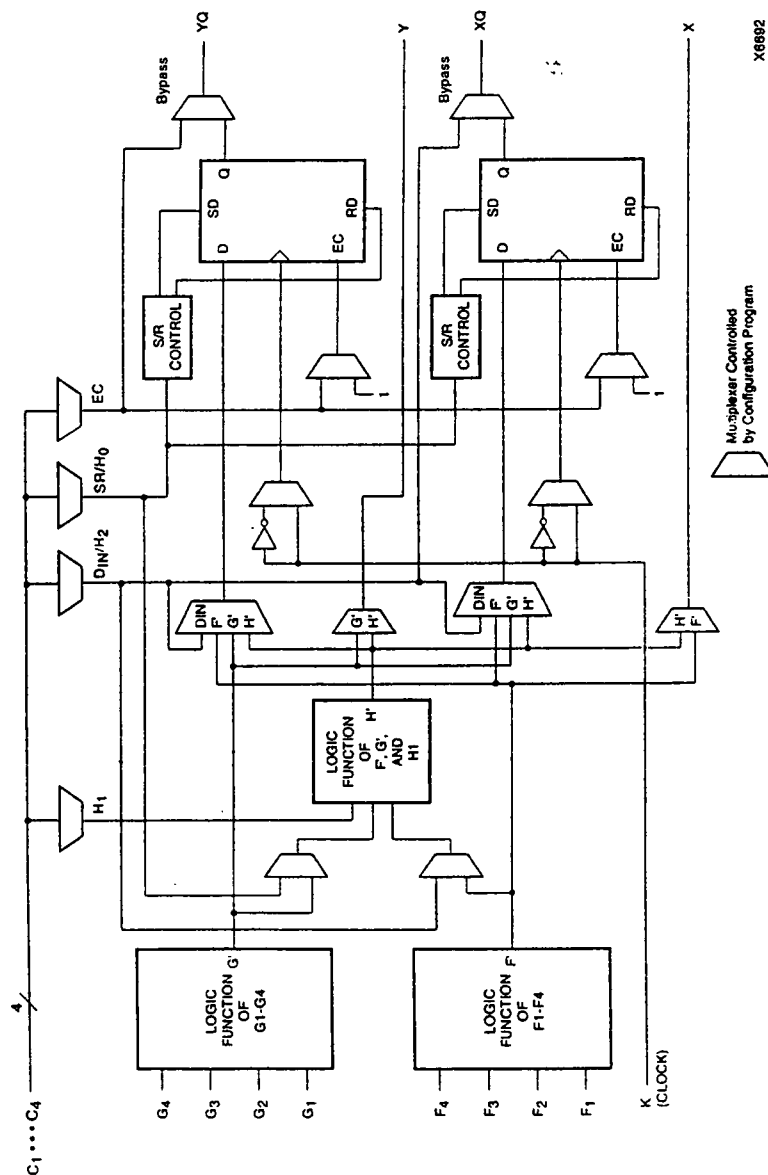
- ◆ 3-state buffers (T-BUFs) driving the horizontal long lines are associated with each CLB.
- ◆ Wide edge decoders are available around the periphery of each device.
- ◆ An on chip oscillator is provided.

Programmable interconnect resources provide routing paths to connect the inputs and outputs of these configurable elements to the appropriate networks.

The functionality of each circuit block is customized during the configuration by programming internal static memory cells. The values stored in these memory cells determine the logic functions and interconnections implemented in the FPGA. Each of these available circuits is described in this section.

Configurable logic blocks implement most of the logic in an FPGA. The principal CLB elements are shown in the Figure 2.

The 4-input function generators (F and G) offer unrestricted versatility. Most combinatorial logic functions need four or fewer inputs. However, a third function generator (H) is provided.



X6892

Figure 2 - XC4000 CLB diagram

Source: Xilinx, Inc.

The H function generator has three inputs. Zero, one or two of these inputs can be the outputs of F and G; the other input(s) are from outside the CLB. The CLB, therefore, can implement certain functions of up to nine variables, like parity check or is expandable. Each CLB contains two storage elements that can be used to store the function generator outputs. However, the storage elements and the function generators can also be used independently.

These storage elements can be configured as flip - flops in both XC4000E/X devices; in the XC4000X they can be optionally be configured as latches. DIN can be used as a direct input to either of the two storage elements. H1 can drive the other through the H function generator. Function generator outputs can also drive the two outputs independent of the storage element outputs. This flexibility increases logic utilization and simplifies routing. Thirteen CLB inputs and four CLB outputs provide access to the function generators and storage elements. These inputs and outputs connect to the programmable interconnect resources outside the block.

Function Generators: Four independent inputs are provided to each of two function generators (F1-F4 and G1-G4). These function generators, without outputs labeled F' and G' are each capable of implementing any arbitrarily defined boolean function of four inputs. The function generators are implemented as memory look-up tables. The propagation delay is therefore independent of the function implemented.

A third function generator, labeled H', can implement any boolean functions of its three inputs. Two of these inputs can optionally be the F' and G' functional generator outputs. Alternatively, one or both of these inputs can come from outside the CLB (H2, H0). The third input must come from outside the block (H1).

Signals from the function generators can exit the CLB on two outputs. F' or H' can be connected to the X output. G' or H' can be connected to the Y output.

A CLB can be used to implement any of the following functions:

- ◆ Any function of up to four variables, plus any second function of up to four variables, plus any third function of up to three unrelated variables.
- ◆ Any single function of five variables.
- ◆ Any function of four variables together with some functions of six variables
- ◆ Some functions of up to nine variables.

Implementing wide functions in a single block reduces both the number of blocks required and the delay in the signal path, achieving both increased capacity and speed.

The versatility of the CLB function generators significantly improves system speed. In addition, the design-software tools can deal with each function generator independently. This flexibility improves cell usage.

Flip - Flops: The CLB can pass the combinatorial output(s) to the interconnect network, but can also store the combinatorial results or other incoming data in one or two flip-flops, and connect their outputs to the interconnect network as well. The two edge-triggered D-type flip-flops have common clock (K) and clock enable (EC) inputs. Either or both clock inputs can also be permanently enabled.

2.6 Description of XC4000E and XC4000XL architecture differences

Essential differences in XC4000X only with respect to XC4000E are as follows:

Increased routing: New interconnect in the XC4000X includes twenty-two additional vertical lines in each column of CLBs and twelve new horizontal lines in each row of CLBs. The twelve "Quad lines" in each CLB row and column include optional re-powering buffers for maximum speed. Additional high-performance routing near the IOBs enhances pin flexibility.

Faster Input and output: A fast dedicated early clock sourced by global clock buffers is available for the IOBs. To ensure synchronization with the regular global clocks, a Fast Capture latch driven by the early clock is available. The Input data can be initially loaded into the Fast Capture latch with the early clock, then transferred to the input flip-flop or latch with the low-skew global clock. A programmable delay on the input can be used to avoid hold time requirements.

Latch capability in CLBs: Storage elements in the XC4000X CLB can be configured as either flip-flops or latches. This capability makes the FPGA highly synthesis-compatible.

IOB output MUX from output clock: A multiplexer in the IOB allows the output clock to select either output data or the IOB clock-enable as the output to the pad.

Additional address bits: Larger devices require more bits of configuration data. The XC4000X therefore extends the addressing in master-parallel configuration mode to 22 bits

2.7 Possible solutions: Implementable and Unimplementable

Modeling solutions: GLINK Transmitter and Receiver chips – The modeling of the chips could be implemented in any of the three modeling capable description languages, viz. VHDL, Verilog HDL or 'C'. Each one has its own nuances and advantages over each other. The CAD tool that was planned on being used for building the model was Viewlogic Powerview (v6.0) toolset. It also became clear, through talks with the CAD support, that VHDL methodologies were much better supported than Verilog or 'C' designs. This was a site-specific CAD tool limitation, not something that Powerview inherently had, and therefore, veered the decision to model the chips in VHDL. Also, aiding this decision, in a significant manner was the lack of exposure to Verilog for descriptive purposes on my part.

The FPGAs had to be designed in Xilinx's proprietary tool, namely, *Foundation Series*, since part of the design had already been captured in that tool. Also, to migrate the design to the Powerview environment was difficult due to the lack of proper libraries.

The analog subsystems were planned for implementation in a behavioral manner. Since, the behavioral model for a rudimentary Phase Locked Loop was already available, it was decided to work around with this code for the present and explore the simulation integration process to build a mixed signal model.

2.8 Design and Software Flow

Design Methodologies: There are many ways of describing a design. The plethora of design data can be understood more clearly by separating "design description and notation". To describe the designs unambiguously, data must be created, or generated, in three domains of description: structural domain, behavioral domain and physical domain [2] as shown in figure 3.

Once the design description and notational aspects have been resolved and distributed, the task of actual description and coding is done through a series of activities. These are in general terms defined as “Design Process flows”. The primary activities in designing may be termed design specification, design capture, design validation and physical design shown in figure 4.

2.8.1 Xilinx Foundation Series Software Toolbox elements

Design entry in the Xilinx Foundation series is possible in 3 different ways:

1. Using a VHDL description (Behavioral and / or structural).
2. A state machine description using a graphical editor (Behavioral domain).
3. Schematic editor for structural definitions (Structural Domain).

Functional simulation is performed after initial design capture to validate the functionality of the design. An iterative process is involved once the design is captured and simulated for functionality until desired performance is obtained. Thereafter, the design flow leads to the implementation based on some technology. The XACT M1 Design manager allows the functional design to be mapped and optimized for a particular Xilinx part. Implementation options and controls lets the designer guide the implementation as to achieve desired performance goals. The implementation produces data to analyze the mapped design, its timing data, the Place and Route issues and so forth.

These can be analyzed and then used as a guide data to further refine the design implementation to meet design criteria.

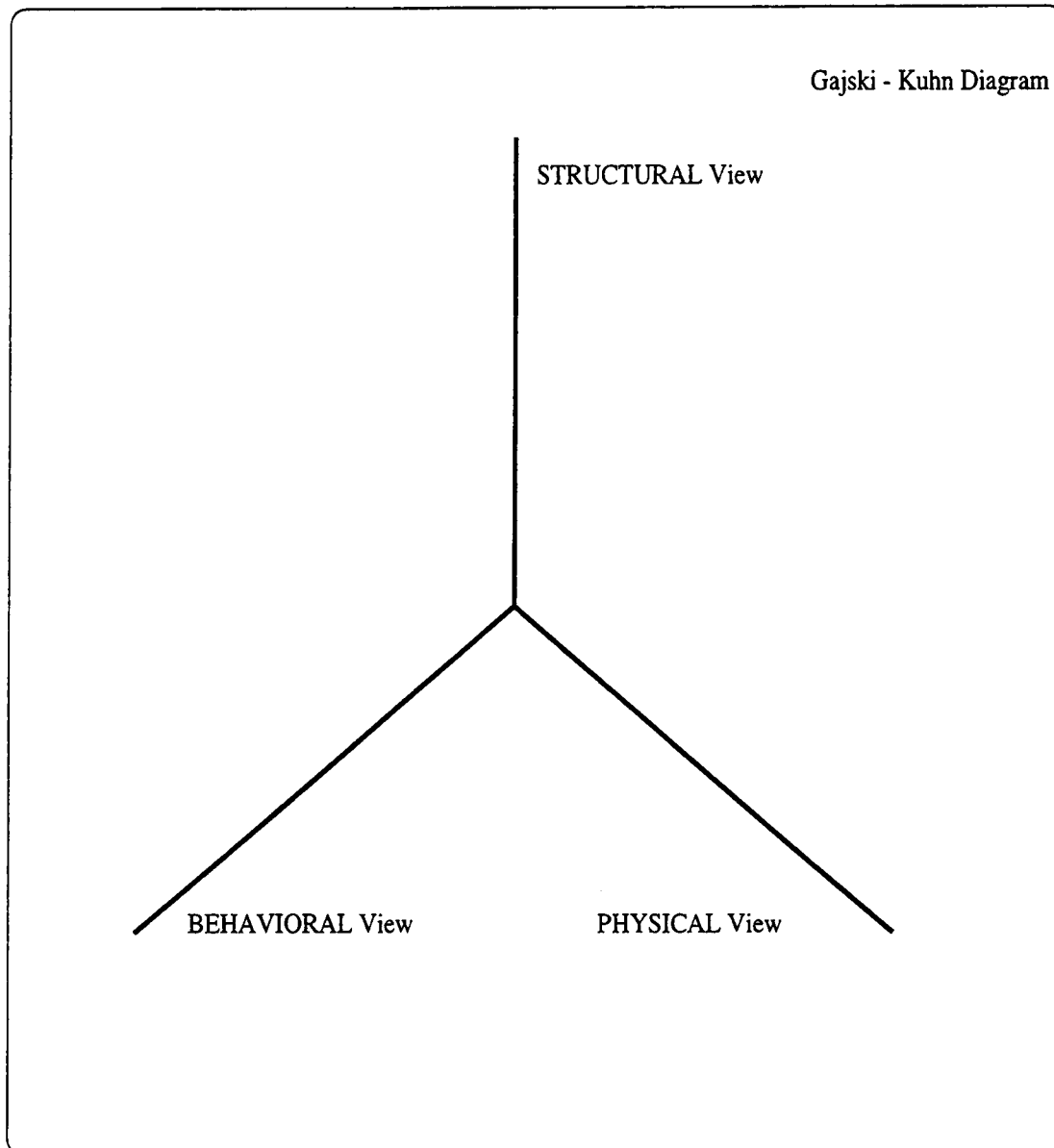


Figure 3 - Domains of description – Gajski Kuhn Diagram

Source: Oldfield, John V., Dorf, Richard C.: Field Programmable Gate Arrays - Figure 1-18

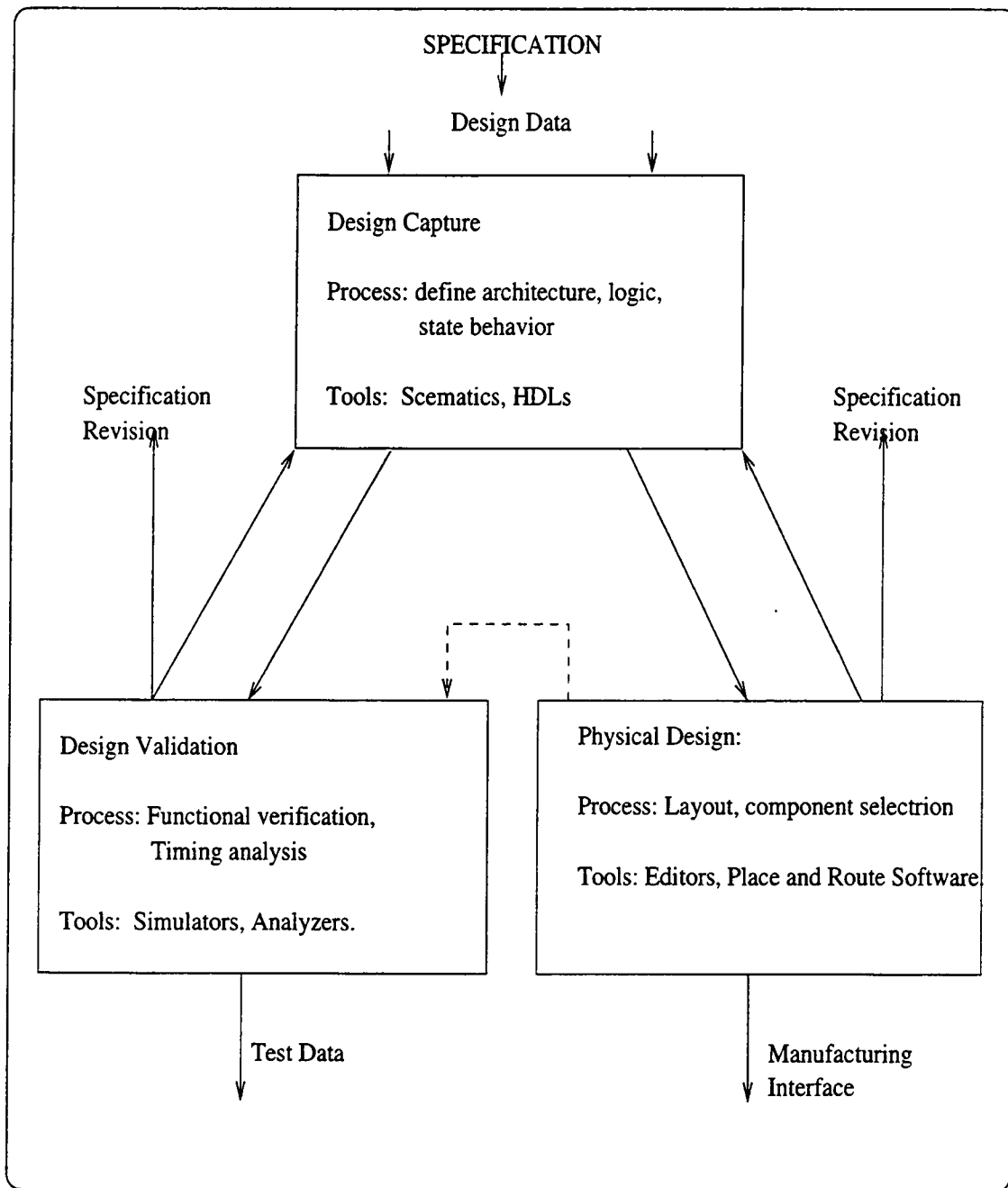


Figure 4 - Design Process flow

Source: Oldfield, John V., Dorf, Richard C.: Field Programmable Gate Arrays - Figure 4-1

The Design manager also allows the designs' configuration netlist to be made available in a format that is compatible with other EDA CAD vendors such as Mentor Graphics, ViewLogic etc. The Place and Routed design can be transformed into a VHDL / Verilog file with timing information in the form of a Standard Delay format file.

Figure 5 is representative of the Xilinx Foundation Series Design Suite.

2.8.2 ViewLogic Design Flow

The design flow for the ViewLogic tools also are similar to that mentioned previously. The process of design capture involves the integration of different digital, analog and delay description libraries. Once this phase is over, the netlisting process takes place, generating .vli, .vsm files for digital simulation, while producing .cir files for analog models. These files contain the delay annotations and are then pulled into the simulator and analyzed for performance.

A figure representing a composite design flow [4] with representative library inclusion is shown in Figure 6 to grasp the essence of the design capture through to simulation stage.

2.8.3 Comparison between XC4005XL (3.3V) and XC4005E (5V) simulation for ALM FPGA

The ALM FPGA design was mapped to both the 3.3V and 5V part to get preliminary design estimates in terms of estimated design operating frequency and so forth. In table 1 we look at the estimated maximum operating clock frequency for the two different technologies under different optimization settings.

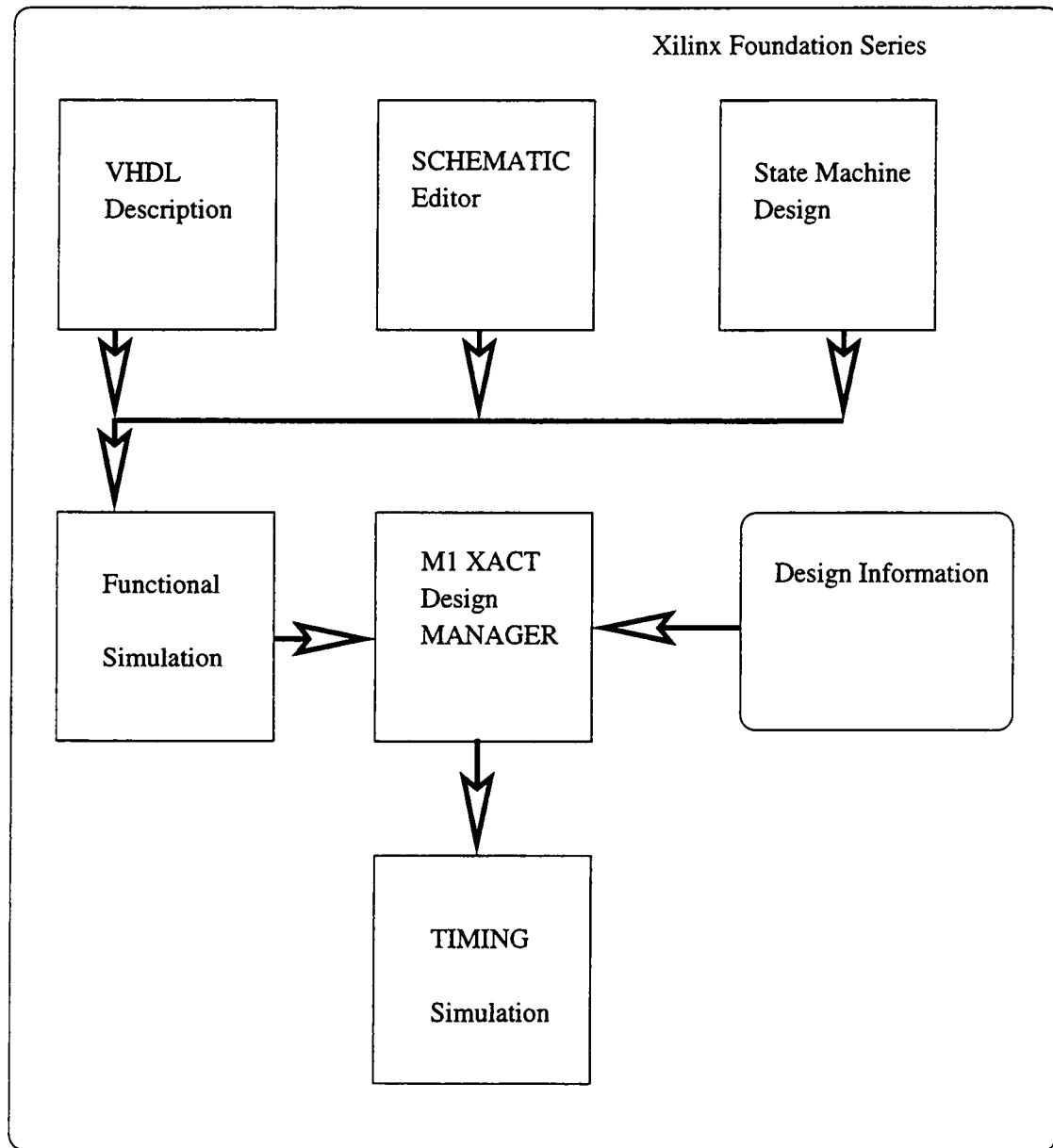


Figure 5 - Xilinx Foundation Series Design Suite

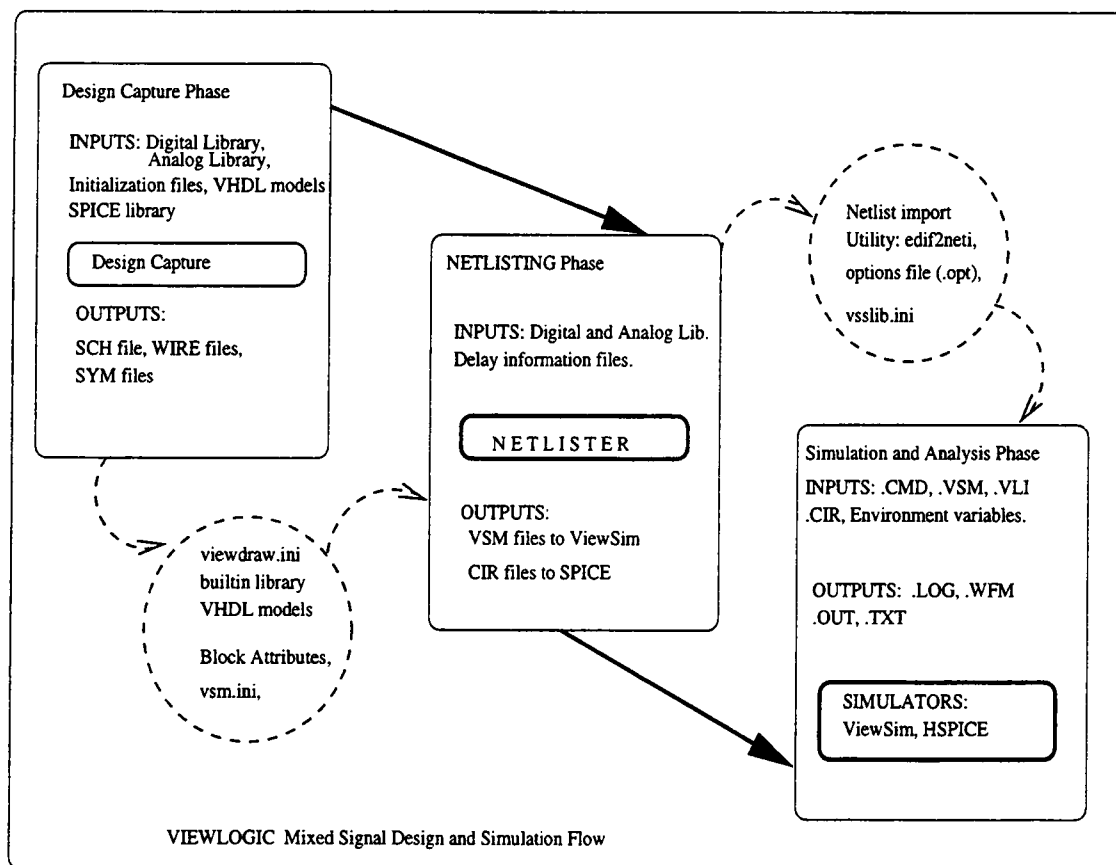


Figure 6 - ViewLogic Mixed Signal Design Flow

Adapted from ViewSpice User's Guide: Mixed Signal Design with ViewSpice – ViewLogic Inc.

Table 1 - Comparison of XC40005E and XC4005XL performance

Part Number (Voltage)	Optimization OFF	Optimization BALANCED	Optimization SPEED
XC4005E (5V)	11.157MHz.	11.564MHz	11.564MHz.
XC4005XL (3.3V)	11.183MHz	12.975MHz	12.975MHz

Clearly, the XC4005XL (3.3V) part performed better in the initial design estimates. Also, due to the fact that the XL technology part was a low power device helped lower the power requirements, though it is not known if there were any power specifications for the Heap Manager board.

However, one of the main driving forces in selecting the XL technology part, as in any business or project enterprise, was COST. At the time of evaluating options on the 4000E and 4000XL series, it was found that the XL part was about 40% the cost of the E (5V) technology part. This surely appealed to the board designers and the decision to switch to XL (3.3V) part was made.

2.9 Expected Results

It is expected that a working mixed mode simulation model would be available for the GLINK chips. Also, a functional verification of the board level system will be performed with the developed model for the GLINK Transmitter and Receiver chips. This will enable the verification of the model's working in the system for which it was primarily developed. Also, tools and libraries needed to move a design netlist for simulation between the two CAD tool environments would also have been figured out enabling future designs to be easily migrated across the platforms and CAD tools.

Also expected, is the gain of insights into the development of commercial models, their user friendliness and the mismatch between those levels of service and that provided by the models as part of this simulation model-development effort.

Chapter 3

FPGA Designs

3.1 Functional Description of the Address List Manager FPGA

The address list manager ALM FPGA's task is to manage an address space with a length of 64 addresses.

A 64 word deep FIFO implements the buffer task. Once the FIFO is initialized from the INIT block during startup, the FIFO queue waits for about 40 words to be passed through. At this instance AMU words are written to the FIFO locations and there exists a one to one correspondence on the FIFO input and outputs. By the time the data written as an AMU word arrives at the output end of the FIFO, the arrival of a trigger signal, namely LVL1 indicates the necessity of the contents and the address is routed to the FIFO ADC queue. Otherwise, it is passed through to the FIFO Available route. In the former route, the address is sorted and its contents transferred and used. The address word, then continues to proceed, to a FIFO ADC DONE block to indicate its contents have been utilized and gets into a queue for going back to the Available FIFO address list. The Re-sorting logic block takes care of arranging the address in a sequential manner from the addresses arriving through the top and bottom routes. This design's netlist is being imported after its implementation by another designer. Figure 7 shows a block diagram of the ALM FPGA design.

3.2 Function Description of the Heap Manager FPGA

The Heap Manager FPGA is involved in managing a data heap read out process from 6 boards each with about 26 channels in a specific readout manner. The boards are read sequentially 1 through 6. However, the channels that are read are sequential though not contiguous. The

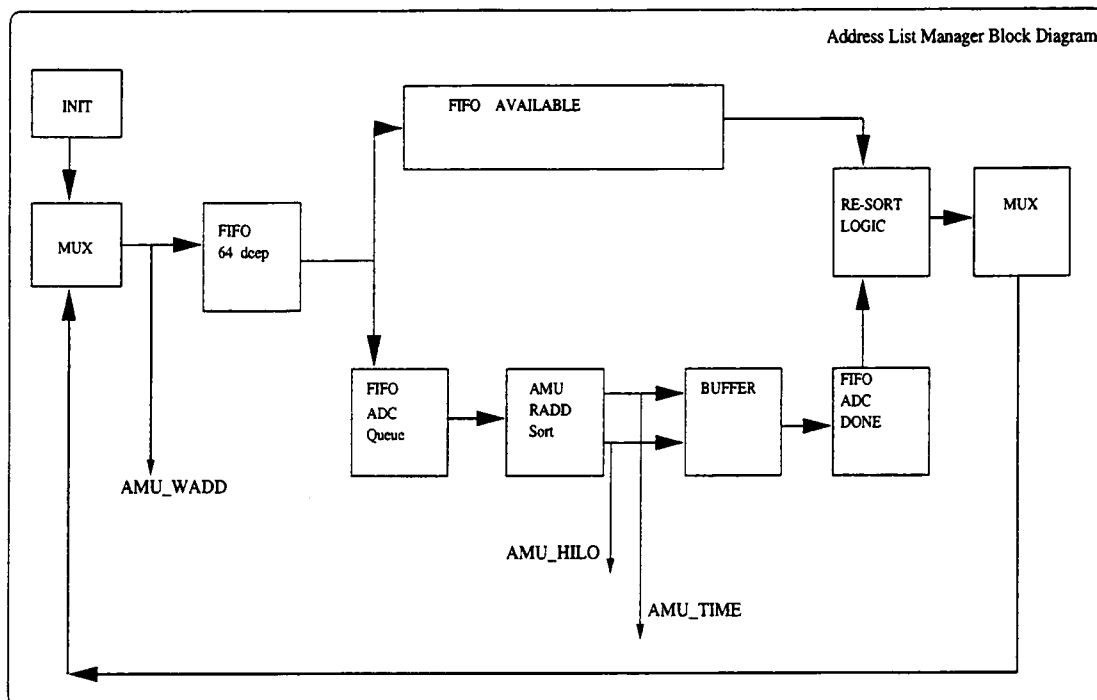


Figure 7 - Block Diagram of Address List Manager FPGA Design

channels are read 1 through 11, 16 through 27. Once all the boards have been read, two more channels are read now. These are channels 28 and 29 from each board 1 through 6.

Also, there is a concurrency requirement in the specifications for the heap manager. The board readouts are initiated upon being triggered by a TRIGGER FUNCTION signal.

Once board readout has begun, the arrival of another board readout trigger is buffered while the previously initiated readouts are still being handled by the readout facility. Once, the first readout is completed, the buffer is checked to see for the existence of another readout trigger and a new readout process initiated. Otherwise, a wait state is entered until a new trigger is available.

A host of other signals also needed to be coordinated to enable the Heap Manager FPGA to perform its function effectively. Some of these are shown in the example simulation output.

These include the Data formatter write enable signals, HILO_RD and TIM_RD signals and so forth. The Channel readout sequence and state machine loop structure is shown in Figure 8.

The EMCAL heap manager board needed to be verified for functionality and timing estimates before putting it into fabrication as is necessary for any board level design.

State machine redesign and new technology mapping was needed since the old design did not have concurrency features and had a sequential and contiguous channel reading process. The re-mapping of the entire Heap Manager FPGA design from the 5V technology XC4005E to the XC4005XL design was done.

The re-mapping process first involved importing the entire ProSeries schematics into the Xilinx Foundation Series tool. The importing process was automated by Xilinx tools that converted the ViewLogic formats into Xilinx compatible formats. This was then followed by a re-mapping process by selecting the necessary part and speed numbers. The re-mapped design was checked and corrected for some signal connectivity losses and a simulation net-list created. A functional verification was done to check for design equivalence in the ProSeries and Xilinx tool domains. The new state machine design was then incorporated and a new functional simulation done to verify operation for new specifications. The functionally verified design was mapped, translated, place and route performed and timing data was analyzed. An optimization of speed and area was done to help the design meet specifications with respect to timing needs. Simulation result for the state machine implementation with concurrency and new board readout order is shown in figure 9. The simulation data shows the completion of the first pass of board readouts and then the beginning of the second pass covering channels 28 and 29.

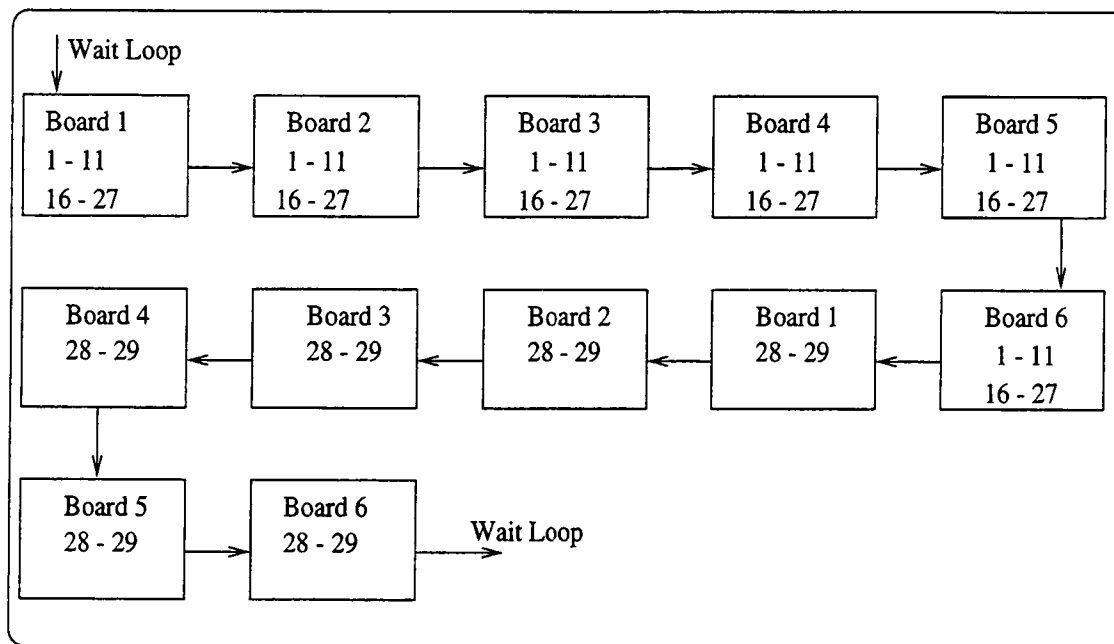


FIG : Channel Read out sequence for Heap Manager FPGA

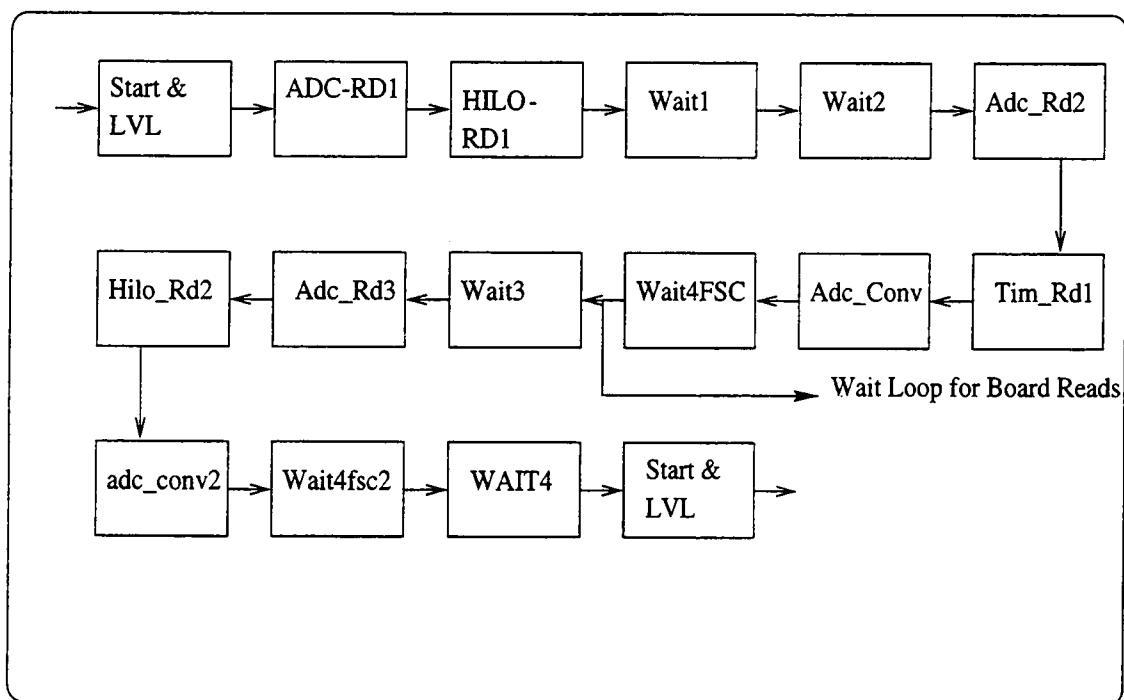


FIGURE : State Machine Loop structure.

Figure 8 - Flow Diagram for the Channel Readout Sequence of the Heap Manager FPGA Design

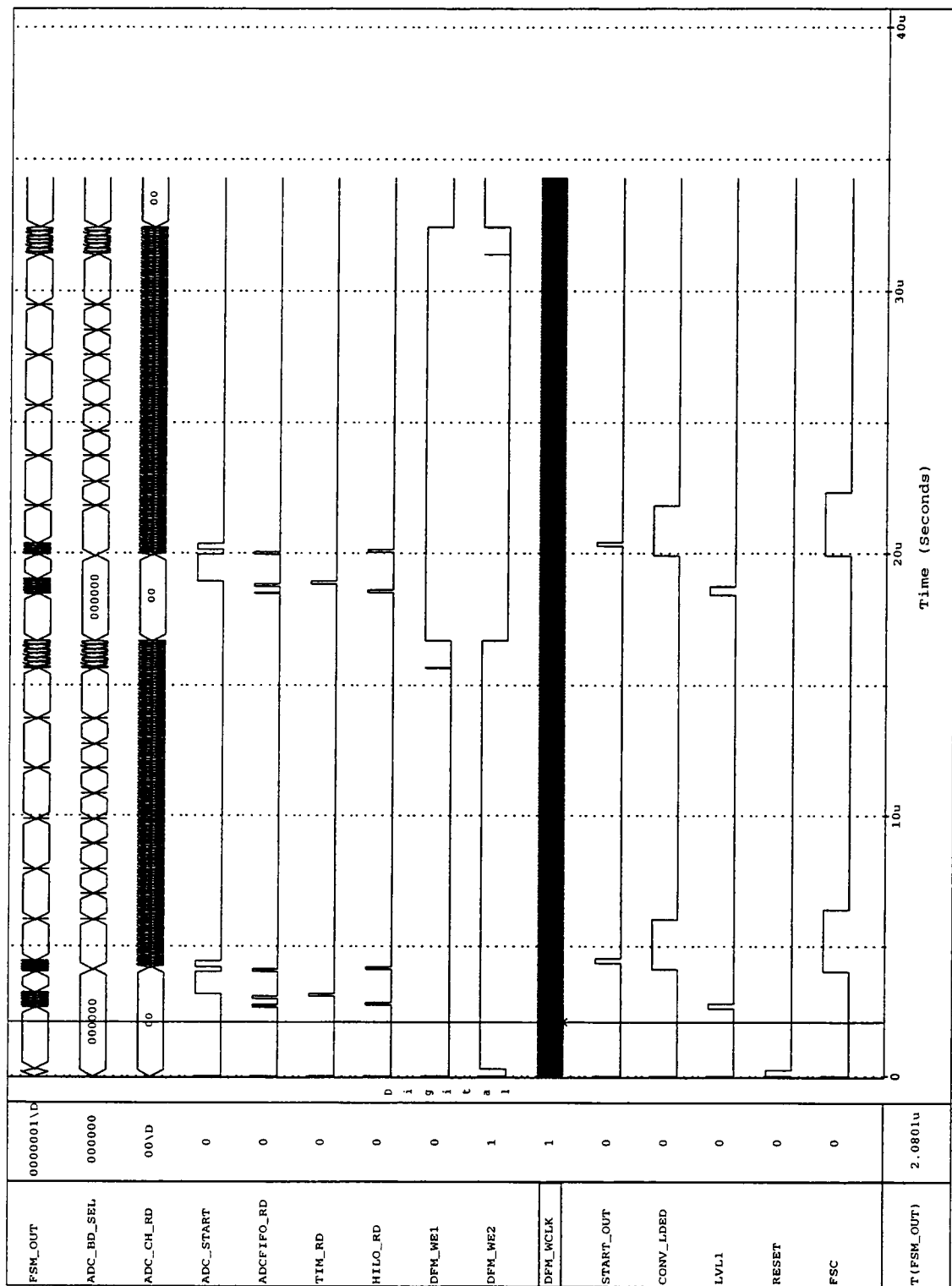


Figure 9 - Simulation Overview: Heap Manager State Machine

Chapter 4

GLINK Chipset Model: Requirements and Implementation

4.1 Description of the GLINK Receiver chip

The Transmitter and Receiver Chips modeling has been divided into two parts. One for the Receiver chip (shown in Figure 10) that is used in the Heap manager Board. The other is the Transmitter Chip that is not used in the system board design but is assumed to be delivering data as an external source.

The HDMP Rx is designed to convert a serial data signal sent from the HDMP Transmitter into either 16,17,20 or 21 bit wide parallel data. In doing these, it performs the functions of

- ◆ Clock recovery
- ◆ Data recovery
- ◆ De-multiplexing
- ◆ Frame decoding
- ◆ Frame synchronization
- ◆ Frame error detection
- ◆ Link State Control.

To this end, various blocks as shown in the block diagram implement the functions.

Input Select: The Input select block determines which input line is used. In normal operation (LOOPEN=0), DIN is selected for input processing. LIN is selected by asserting LOOPEN='1', a feature used for loop back testing exclusive of the transmission medium.

Phase/Frequency Detect: This block compares the phase or frequency of the incoming signal to the internally generated serial clock (from the CLOCK select block). The output signal, PH1, controls the VCO input (Analog subsystem).

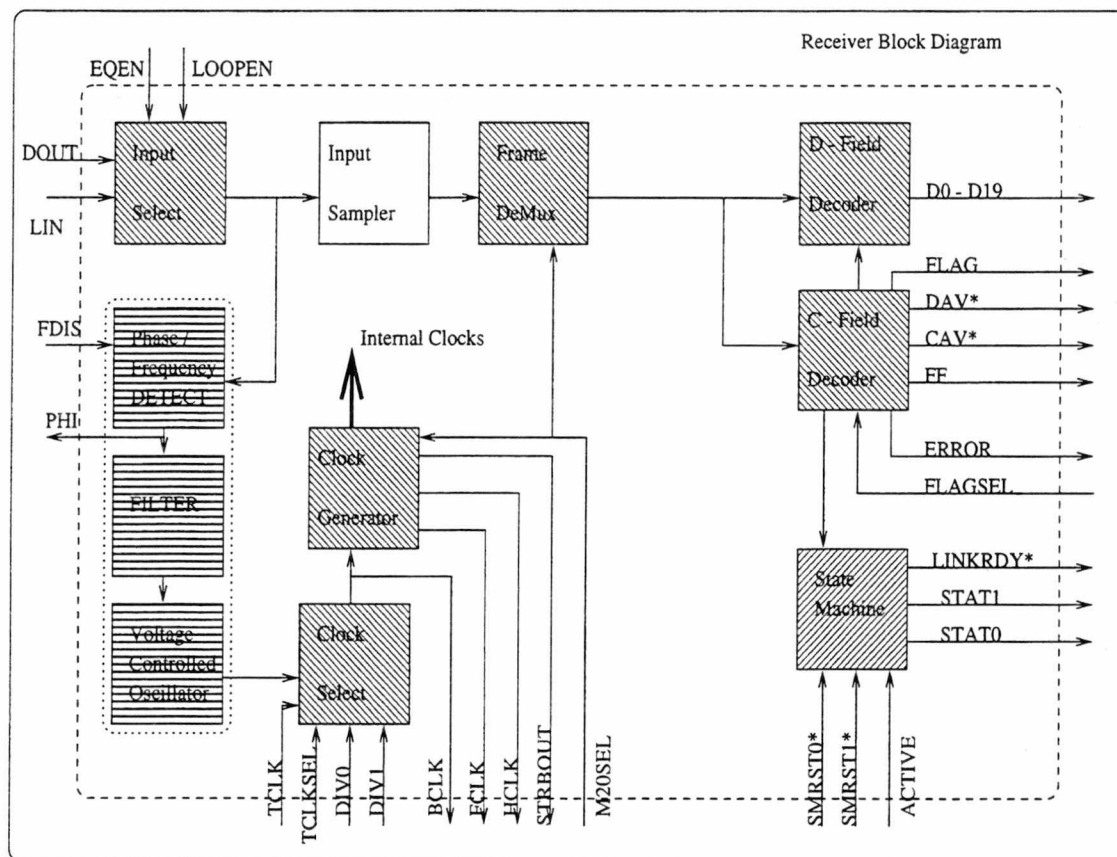


Figure 10 - GLINK Receiver Block Diagram

Filter: This is a loop filter that takes the PHI signal as input and converts it to a control signal for the VCO (Analog subsystem).

VCO: Its output is a high-speed digital signal to the clock select unit (Analog subsystem)

CLOCK select: Generates a high-speed clock internally and is available through the pin BCLK. TCLKSEL enables the use of TCLK input clock signal as the internal high-speed clock signal.

Clock Generator: Generates frame rate clock based on the M20SEL setting. This output is used to ensure proper frame synchronization of the output frame clock, STRBOUT.

Input sampler: The serial input signal is converted into a serial bit stream using the extracted clock from the clock select. This output is sent to the FRAME de-multiplexer (Analog subsystem not implemented).

Frame De-multiplexer: This block modifies the data stream into a 20 / 24 bit-wide parallel data word, based on the setting of M20SEL. The most significant 4-bits are sent to the C - Field Decoder while the remaining 16/20 bits are sent to the D-Field Decoder.

C-Field Decoder: The C decoder accepts the control information from the Frame De-multiplexer and determines the kind of frame being received and its polarity (true or inverted). The control bits are sent to the SMC for error checking. The decoded information is sent to the D decoder. CAV* is set low when received frame is Control word. DAV* is set low if the information is a data frame. If neither DAV* or CAV* is set low, then the frame is expected to

be a fill frame. If FLAGSEL is asserted, the FLAG bit is restored to its original form. Otherwise, FLAG is used to differentiate the even and odd frames in Double Frame Mode (Not modeled) or for error checking.

D-Field decoder: The D decoder accepts the data field of the incoming frame and based on the C-field outputs to it, determines the data type sent and restores the parallel data to its original form.

State Machine: The SMC is used in full duplex mode to perform functions of link startup, link maintenance, and error checking. By setting the SMRST0* and SMRST1* low, the user too, can reset the state machine and initiate link startup. SMRST1* is usually connected to the transmitters LOCKED output.

STAT1 and STAT0 denote the current state of the link during startup. ACTIVE is an input normally driven by the "STAT1" output. This ACTIVE input is re-timed by STRBOUT and presented to use as LINKRDY*.

HDMP-1014 (Rx) Timing: (Figure11) The synchronous outputs, D00-D19, LINKRDY*, DAV*, CAV*, FF, ERROR and FLAG are updated every data frame with a delay of "td1" after the falling edge of STRBOUT. There is a latency delay of two frames from the input of the serial data frame to the update of the synchronous outputs. The SMC outputs, STAT0/1, appear with the falling edge of the STRBOUT after a delay of "td2". These outputs are updated every 128 frames.

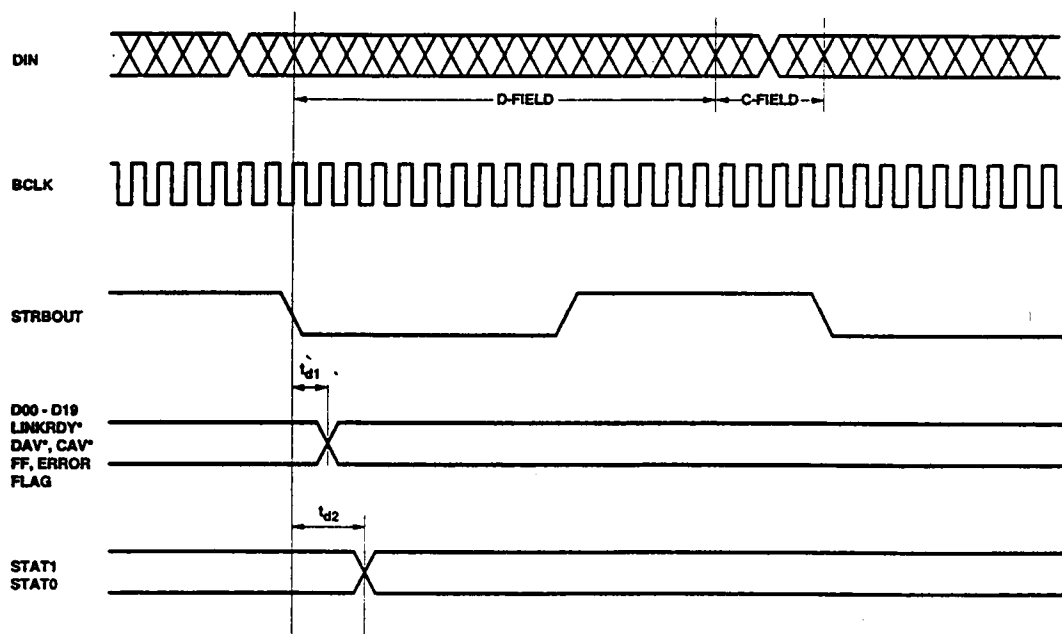


Figure 11: Receiver block timing diagram

4.2 Model for the Receiver Chip

The modeling of this chipset was done mostly in the ViewLogic Powerview CAD tool. A two level approach was used. One level consisted of VHDL models for different functional blocks that then were associated with a second level of schematics connecting the different functional blocks.

There are several requirements for the receiver chipset and some of them were embedded implicitly in the data sheet specifications. Following is a brief description of the requirements of the receiver chip and the status of the model in terms of its meeting these requirements. Some requirements have been partially modeled.

Some assumptions that form the basis of the model:

- ◆ Currently, it is assumed that the PLL output of the receiver chip has reached a steady state under the LOCKED status.
- ◆ Error Control is not extensively implemented. It is assumed that the error will be flagged. However, the model will not correct them though "single-bit" errors are corrected by the actual HDMP 1014 chip.
- ◆ The Double frame option has not been modeled explicitly.
- ◆ Use of extracted clock by the PLL for running the receiver chip.

The model assumes the PLL to be in LOCKED condition and therefore works at a single clock frequency as specified in the command file supplied for verification.

This imposes the restriction that the appropriate frequency be entered for the model to work properly.

The Model options modeled are as follows:

1. The model is able to operate in both the 16 bits and 20 bits mode. This operational mode is selected by an input pin named M20SEL. A "LOW" in this input specifies operation in 16 bits mode while a "HIGH" enforces 20 bits mode.
2. The model to works in a 17-bit and 21-bit mode if the input pin named FLAGSEL is asserted "HIGH". When asserted "LOW", the model should be able to output an error checking enhancing feature output through the FLAG output pin.

If a Fill Frame is detected, the FLAG pin remains LOW (0). A Control word detection involves the toggling of the FLAG output between "0" and "1". The Data frame uses this FLAG output bit as an error detection and control feature during the operation of the chipset by toggling between "0" and "1" having started with a "0" and ending with a "1".

The workings of the FLAG output pin is described below.

When FLAGSEL= "0": If the C-Field is "1011" then next C-Field HAS TO BE from "1101" or "0010" plus the DC balancing requirement confining the actual choice of the C-Field encoding.

Representative calculations are shown in table 2.

Table 2 - Data frame interpretation by Rx

Current C-Field	Current Flagbit	DC link Requirement	Next C-Field	Next Flagbit
1011	0	None	1101	1
1101	1	None	1011	0
1011	0	Yes	0010	1
0010	1	Yes	0100	0

When FLAGSEL= "1": Depending on the Flagbit only, True/Inverted signal C-Field is available to maintain DC balance that automatically determines the next C-Field. Representative calculations for this condition are shown in table 3.

Table 3 - Data frame interpretation by Receiver module

Current C-Field	Current Flagbit	DC link Requirement	Next C-Field	Next Flagbit
1011	0	None	1101	1
1011	0	Yes	0010	1
1011	0	None	1011	0
1011	0	Yes	0100	0
1101	1	None	1101	0
1101	1	Yes	0100	0
1101	1	None	1101	1
1101	1	Yes	0100	1

3. The use of Start-up Link control through the workings of the state machine controller.

The State Machine control works primarily to facilitate

- a) Link Start up and maintenance.
- b) Error checking ability and 1-bit error correction.
- c) Essential CONTROL and STATUS signal updates.

The State Machine block is implemented physically in the Rx chip though logically a separate entity. The SMC monitors the data frame status indicators (ERROR, DAV, CAV, FF) from the Rx, and is able to force (or control) various characteristics of the TX and the Rx chip.

During Link startup, the SMC sees to it that three essential states are passed through for LINK ready signals to be activated properly for transmission and receipt of data.

These are

- ◆ State 0: Frequency Acquisition mode shown by STAT = "00".
- ◆ State 1: Waiting for peer mode indicated as STAT = "10".
- ◆ State 2: Sending Data frames mode defined by STAT = "11".

The Link "restart" in HDMP chips is achieved over 128 frame periods as compared to the model which establishes LINK READY status in approximately 14 Frame time periods. This restart in the model, currently, is caused only due to a reset signal being applied to the Rx chip.

During link startup the SMC model checks for the receipt of FF0s or FF1s during State 1, while looking for FF1s specifically during the latter half of the State 1. The SMC, however, does not perform any kind of error checking function on the received data frames once link is established.

The error modification capability can be included by the user in the model as a report type rather than modify and send it to the parallel outputs. For this purpose, the SMC has the data available at its inputs.

SMC module in the Rx chip would flag two non-modifiable errors and reset the Link. However, the model does no such reset operation. The only error that it would detect would be alternation of the flag bit if the transmitted frame was a data frame. This is possible because of the strict alternating nature of the encoded control fields.

Implemented GLINK Receiver chip blocks are shown in Figure 12.

4.3 Simulation Results for the Receiver chip of the GLINK chip

Rx simulation (Figure 13) annotation: The simulation shows the Link startup of the Rx unit. The SMC status lines are "STAT" and goes through the prescribed sequence of states '0, '2' and '3'. The link startup in state 0 is achieved by the use of FF0s while state 1 (STAT = 2) is filled with FF1s. Moreover, it is seen that the FLAG bit is automatically toggled as soon as "LINKRDY" goes low. This occurs for both the control words and the data words.

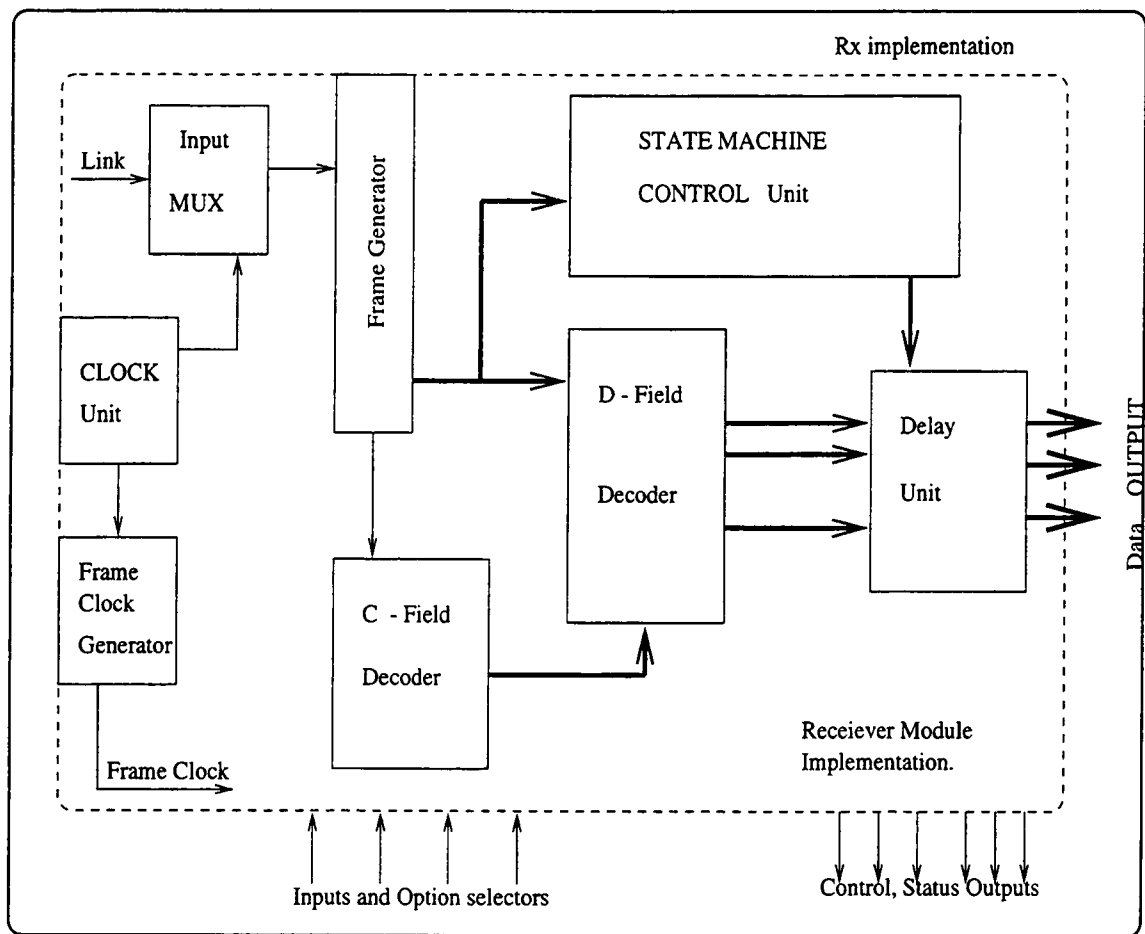


Figure 12 - Implementation of the Receiver Module

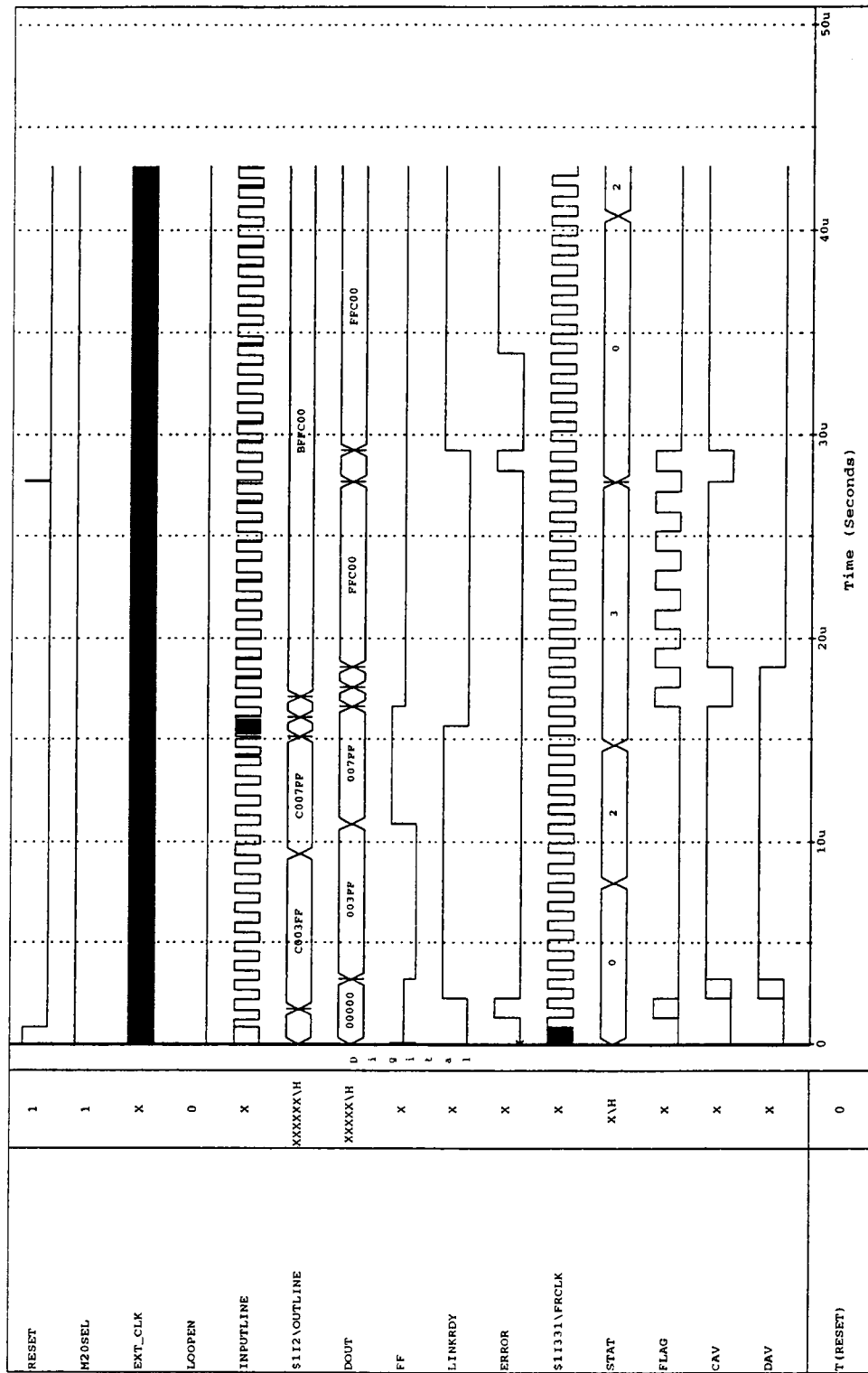


Figure 13 - Overall View of a Simulation of the Implemented GLINK Receiver

Around 28 μ s, the Rx is reset, and the SMC status lines show "0" indicating restart of link establishment procedure. However, note that the restart does not proceed error free. The error signal goes high indicating that the link procedure is not correct. This is due to the fact the input sequence is not being followed for the startup, i.e. the Fill frames for frequency and phase lock is not available.

4.4 Modeling the GLINK Transmitter

4.4.1 Transmitter Functionality

The HDMP 1012 was designed to accept 16 or 20 bit wide parallel data and transmit it over a high speed serial line while minimizing the users' necessary interface to the high speed circuitry. In order to accomplish this task, the HDMP 1012 (Figure 14) performs the following functions:

- Parallel Word Input
- High Speed Clock Multiplication
- Frame Encoding
- Parallel to serial Multiplexing.

PLL / Clock Generator:

The phase lock loop and clock generator are responsible for generating all internal clocks needed by the transmitter to perform its functions. These clocks are needed by the transmitter to perform its functions. The clocks are based on a supplied frame clock (STRBIN) and control signals (M20SEL, MDFSEL – not modeled, EHCLKSEL, DIV 1 and DIV 0). In normal operation (MDFSEL=0), STRBIN is expected to be the incoming frame clock. The PLL/CLOCK generator locks on to this rate and multiplies the clock up to desired speed based on M20SEL. HCLK pin allows access to the high speed serial clock.

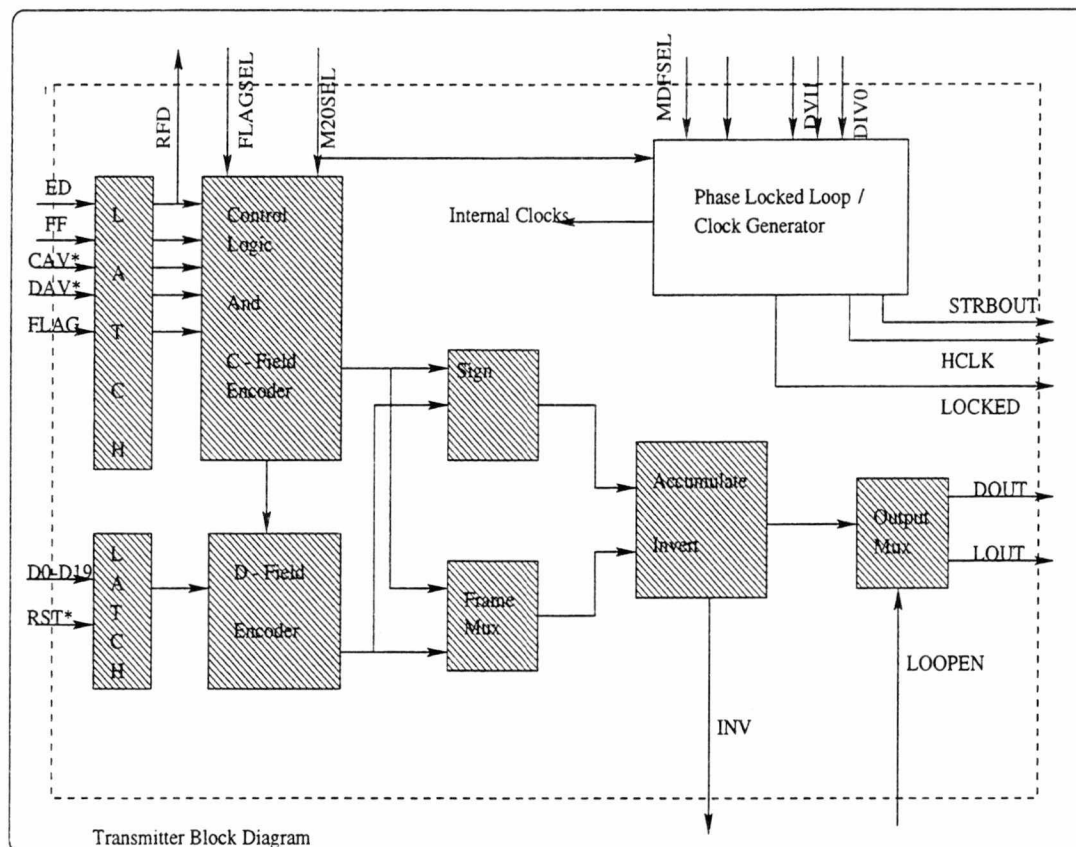


Figure 14 - Block Diagram of GLINK Transmitter Chip

EHCLKSEL allows for the use of an externally supplied high speed clock (which the model uses) at STRBIN. This clock is used to directly output the serial data. The PLL modeling is discussed under the section dealing with analog subsystems modeling.

Control Logic and C – Field Encoder: The control logic is responsible for determining what information is serially sent to the output. IF CAV* is low, it sends the data at D0...D8 and D9...D17 as control word information. IF CAV* is high and DAV* is low, it sends parallel word data at the data inputs. If neither CAV* nor DAV* is set low, then the transmitter unit assumes that the link is not in use. It is triggered by control logic to send Fill Frames to maintain link DC balance and allow receiver to maintain frequency and phase lock. The type of fill frames sent (FF0 or FF1a/b) is determined by the input at the FF pin. In a duplex connection, FF pin is connected to the STAT1 pin of the Rx. The C-Field encoder based on the inputs at DAV*, CAV*, FLAGSEL, and FLAG, supplies four encoded bits to the frame multiplexing unit. This encoded data contains the master transition as well as information regarding the data type: control, data, or fill frame. In order for the FLAG bit to be used as an additional data bit, FLAGSEL must be high for both the Tx and Rx unit.

D – Field Encoder: The D-Field encoder provides the remaining parallel word data to the frame multiplexing unit. Based on control signals from the control logic, the D-field encoder outputs either the parallel information at its data inputs (D0...D19) or the designated Fill Frame. RST*, when set low, resets the internal chip registers.

Frame Multiplexing unit: The Frame multiplexer accepts the output from the C-field and D-field encoders. The four control bits are attached to the data bits, either 16 or 20 bits based on

the M20SEL input. This parallel information is now multiplexed to a serial line based on the internal high speed serial clock (or EHCLKSEL based external clock).

Sign: The Sign circuitry determines the cumulative sign of the outgoing data frame, containing the data and control bits. This is used by the accumulator / inverter to maintain DC balance for the transmission line.

Accumulator / Invert: The Accumulator / Invert block is responsible for maintaining the DC balance of the serial link. It determines, based on history and the sign of the current data frame, whether or not the current frame should be inverted to bring the line closer to the desired 50% duty cycle. INV is set high when the data frame is inverted.

Output select: In normal operation, the serial data stream is placed at DOUT. By asserting LOOPEN, the user may also direct the serial data stream to LOUT, which may be used for loop back testing. Typical timing characteristics of the transmitter block are shown in figure 15.

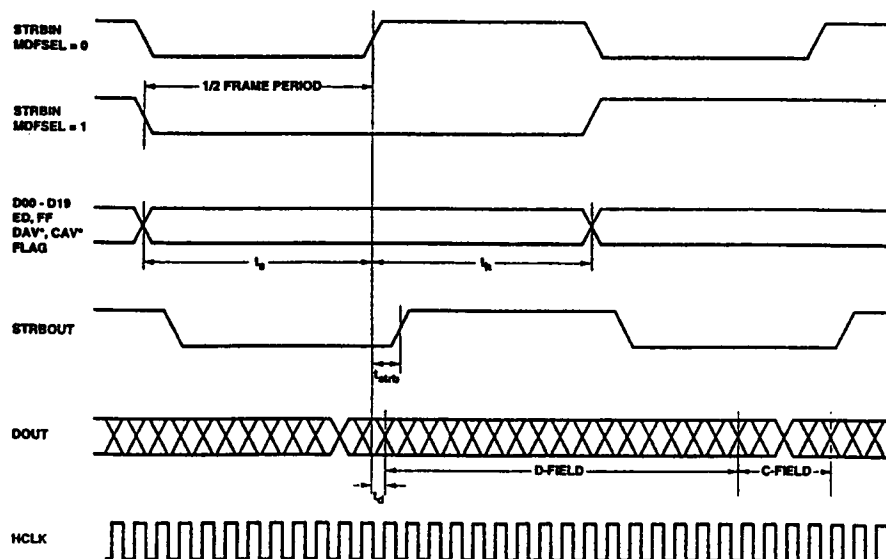


Figure 15: Timing Characteristics of Transmitter Block

4.5 Transmitter Model Implementation

4.5.1 Assumptions, details of the Transmitter implementation

The model for the transmitter was constructed using two levels of data representation as in the case of the receiver model. The top level is a schematic connecting different functional blocks that are each coded in VHDL. The VHDL code is simulation capable but has not been tested for synthesis capability. Some modifications would be necessary to make the model a synthesis compatible core.

Some of the assumptions for the model are as follows:

- ◆ The LOCKED output pin is asserted by the user as needed. It should be maintained low for a few clock cycles after Reset has been removed and then asserted HI to enable Rx to begin frequency tracking.
- ◆ CAV and DAV signals will need to be asserted LO one frame clock earlier than the intended control or data word to be transmitted.
- ◆ The models serial output begin transmitting the input data bus word after 2 frame clocks in the control and data word formats while Fill frames transmitted are available at the output after 1 frame clock.

Model Capabilities:

1. The model achieves the Word width capability of either a 16 bit or 20 bit input . The use of the FLAGSEL pin enables the transmission of a 20 bit word or a 24 bit word with the FLAG input encoded in the transmitted data. This encoding mechanism is the same as the decoding functionality discussed in the Rx block. This has been explained Section 4.2, Tables 2 and 3.

2. The model transmits FF0 frame during link startup while the FF pin is asserted Low and shifts to transmitting FF1a and FF1b once FF pin is asserted HI by the Rx STAT0 pin. However, during idle link maintain period, the model transmits only FF1a instead of a FF1b too as the original chips do.

Model implementation blocks:

The model is implemented as shown in figure 16. The major blocks are as follows:

Input latch: The input latch block latches on to the inputs at the rising edge of the frame clock. This is then presented one clock cycle later to the encoder block for encoding.

Encoder block: The encoder block takes its data from the latch output. Depending on the inputs presented to it, the encoder outputs a 20 or 24 bit pattern with encoded FLAG data if FLAGSEL is asserted. The output of this block is available at the rising edge of the next frame clock.

Parallel to Serial Multiplexer / SIGN: This unit, called tserializer, works on the high-speed clock which is the same as the transmit clock. This transmit clock is derived from the Frame clock and the M20SEL pin setting. This block also calculates the sign of the current data frame. It counts the number of bits that are "1" and depending on a threshold value assigns the output SIGNBIT a "1" or "0". The threshold value is 12 for a 24-bit frame while it is 10 for a 20-bit frame.

Signbit	Bits per frame: No. of bits "1"	Bits per frame: No. of bits "1"
1	24 : 13-24	20 : 11-20
0	24 : 1 - 12	20 : 1- 10

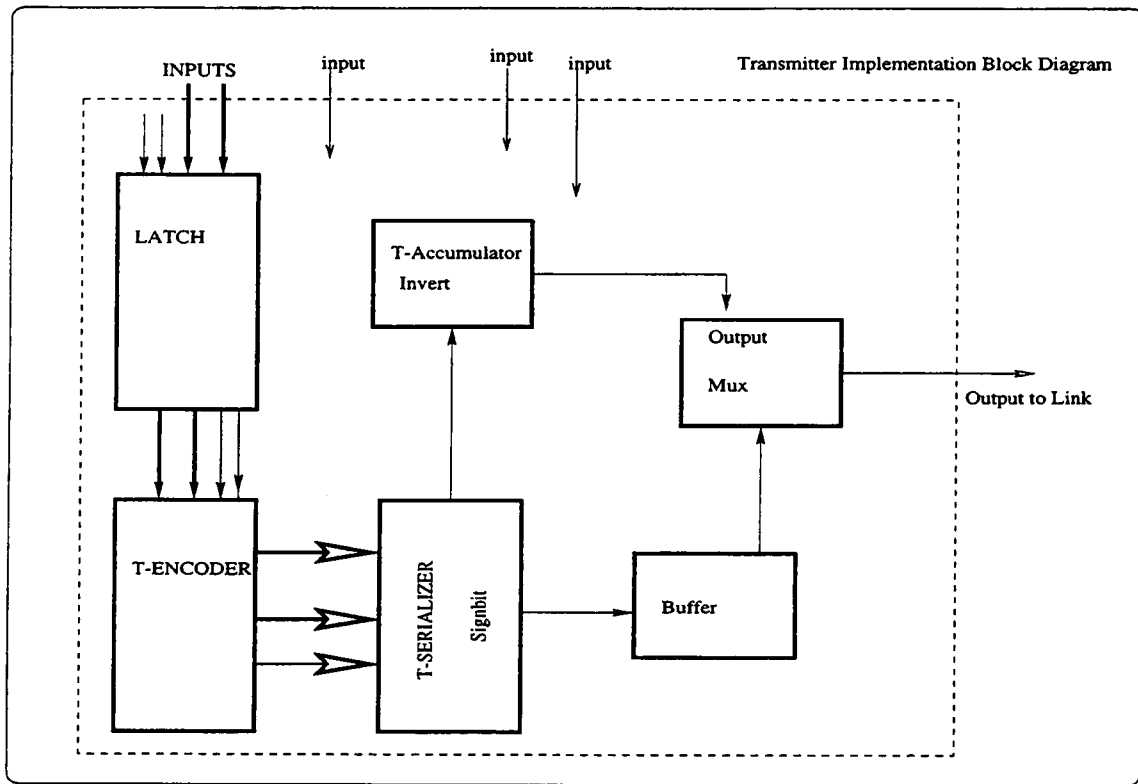


Figure 16 - Transmitter Chip Implementation

However, it must be noted that the adopted method is the model's way of maintaining a DC balance close to the 50% duty cycle of the Link. The model may not be able to maintain a close fit to the needed balance level but always attempts to drive the balance towards the average value.

Buffer unit: The buffer unit simply delays the serial output by one frame clock period so as to let the accumulator / Sign block to give its output.

Accumulator / Invert: This block works on the SIGNBIT output from the tserializer block to decide the need for maintaining DC link balance by inverting the serial output, if necessary. The

following table 4 illustrates the calculation of the requirement for INVERT for the current frame.

Output multiplexer: This block selects the serial output to go to pin DOUT or LOUT.

4.5.2 Non – modeled Input / Output Pins

The pins, inputs or outputs, interfacing the analog subsystems were the ones that have primarily not been implemented. These pins are DIV0, DIV1, CAP pins, EHCLKSEL, HCLK, HCLKON, MDFSEL, LOCKED, STRBOUT. However, the simulation model symbol includes these pins at the top level. Provision is made at an underlying level to incorporate the analog subsystems and create a mixed mode model by connecting these pins and the digital subsystems with the analog section.

Table 4: Transmitter Accumulator Inversion Calculation

Signbit	Prev. Invert	High Sent	Low Sent	Current Invert	Current High-Low
1	1	0	1	0	10
1	1	1	0	1	01
1	0	0	1	0	10
1	0	1	0	1	01
0	1	0	1	1	10
0	1	1	0	0	01
0	0	0	1	1	10
0	0	1	0	0	01

4.6 Simulation Results for transmitter model implementation

Tx Simulation Annotations: From the simulation figure for the transmitter chip, instand alone mode, we see that the output is available after 2 frame clock cycles. Also seen is SIGNBIT signal, internal to the model. This shows the sign of the current frame to be transmitted and lets the chip transmit decide on its transmission sign. The INV signal implements the transmission sign. The implemented transmitter simulation is shown in Figure 17.

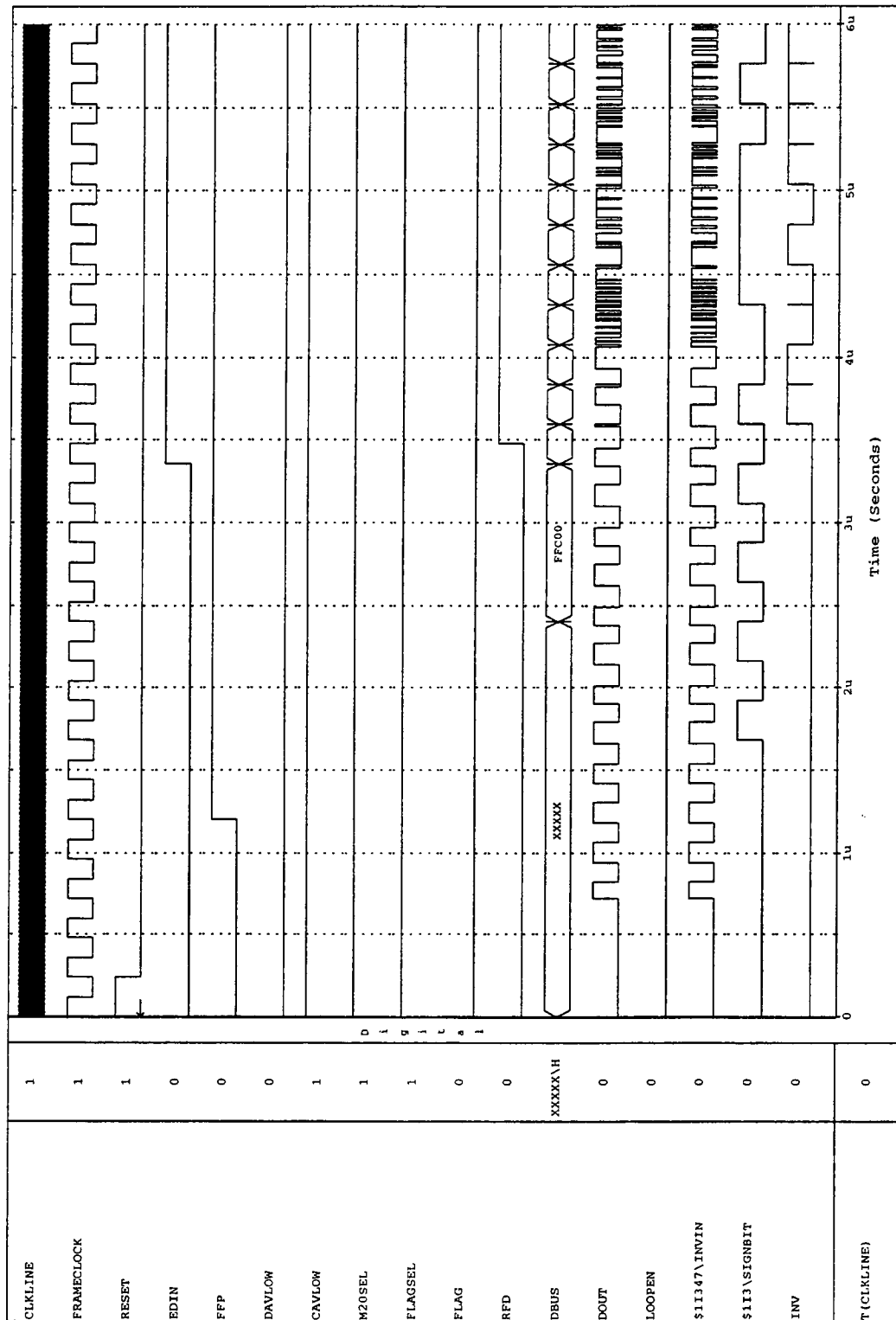


Figure 17 - Overall view of a Simulation of the Implemented GLINK Transmitter Chip

There are some glitches that can be noticed on the INV signal. This is caused due to the way the model blocks are connected. This glitch occurs every time the SIGNBIT signal, internal to the model, changes state from '0' to a '1' or a '1' to '0'. The SIGNBIT output is fed to a following block with just one high speed clock period delay. This occurs synchronously with the rising edge of the Frame clock to the receiving block. The arrival time catches the simulator in a confused state for a point time width before it settles the value to the correct one.

4.7 Model for the Phase Locked Loop

The model for the PLL has been incorporated from META Software's HSPICE manual. We discuss the model here to enable a qualitative discussion on the integration of analog sub-systems and bring about mixed mode simulation enabling mixed signal products to be modeled and simulated for the purpose of integration into a larger system.

The basic unit of a PLL consists of a Phase detector, a Low pass filter, a VCO and a comparator. The Phase detector outputs a voltage proportional to the difference in the frequency at its inputs. This voltage is the controlling voltage for the VCO to produce a frequency that tries to match the input frequency and phase and enable a Locked condition to exist. The modeled functionality here just enables the simulation of a PLL. A block diagram of the PLL unit is in Figure 18 to enable a block view of the PLL sub systems.

From the simulation we see that the output is a clock wave which can be conditioned to obtain a digital clock signal. On the schematic, this signal conditioner will be an A2D net. The A2D net is a special net with a model to convert analog signals to digital outputs. Its relative is the D2A net which enable digital components to talk to analog inputs.

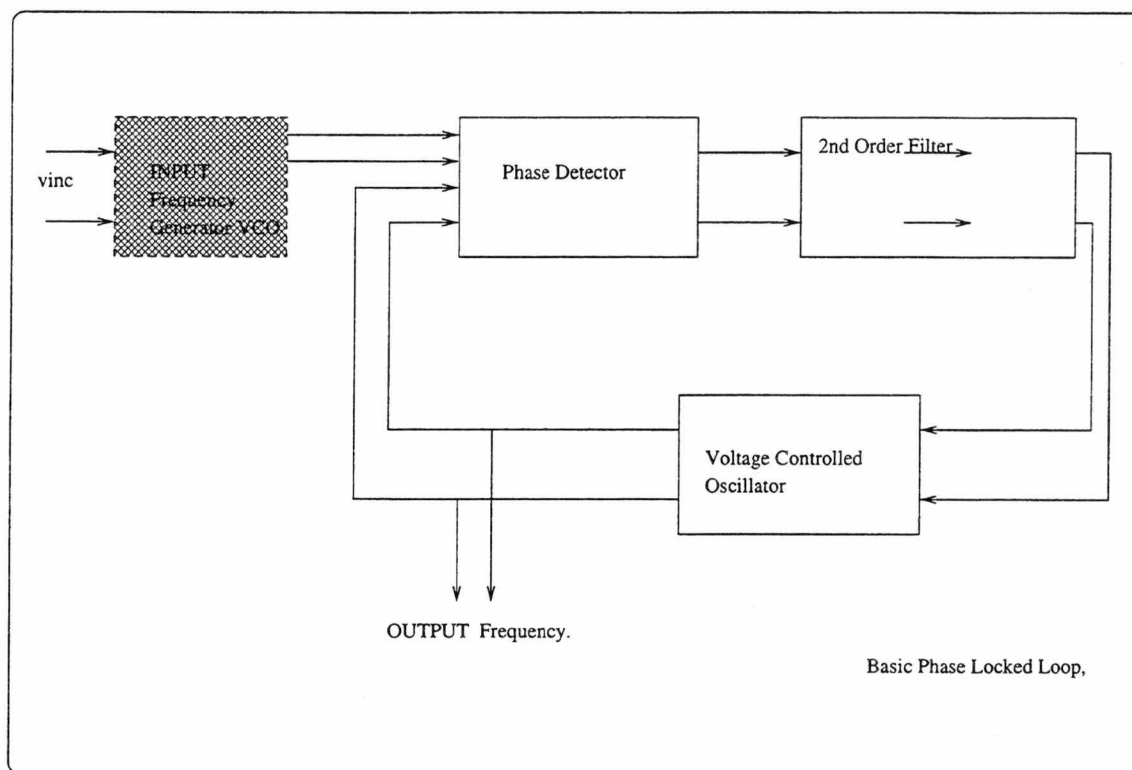


Figure 18 - Block Diagram of a Basic Phase Locked Loop Circuit

Chapter 5

Simulation Integration, Results and Comparison

5.1 Integration Flow for FPGA models and GLINK models

FPGA Models: The FPGA models had to be imported into the ViewLogic Powerview CAD tool. Since, the Powerview tools did not support the XL family of Xilinx, a work around had to be found. Two workaround solutions were determined and the models were successfully imported into the Powerview tool. In the pages to follow, a brief discussion of the two approaches is provided.

Method 1: While using the Foundation Series tool and implementing the design, choose the Implementation option for “interface output” and set it to VHDL of Generic or ViewLogic format. This will generate a VHDL file of the placed and routed design together with another file with the standard delay format (.sdf). These files and a few other files (namely, SIMPRIM_VCOMPONENTS and SIMPRIM_VPACKAGE) from the Xilinx libraries were transferred to the Powerview directory. From here on we use the ViewLogic simulation methodology.

Method 2: During the process of placing and routing the design, Xilinx tools produce intermediate format descriptions of the design. These design formats called, Native Generic Design, contain delay information too. The .ngd file is transformed for the ViewLogic environment using certain libraries. The resulting file is exported to the industry-standard EDIF netlist format. This file then is imported into the Powerview project and reconverted into the ViewLogic format.

Certain library files, such as the Xbuiltin, builtin and SIMulation PRIMitiveS files were copied and referenced in the Powerview project work area. This helps the EDIF to ViewLogic translator to reference the correct instances of parts and rebuild a WIR file. This WIR file can be used to build a ViewGen Symbol and later to create a .VSM file for simulation purpose.

5.2 Simulation of GLINK Chips: Configurations

The GLINK chipset was simulated under different configurations to verify proper functionality under each test case. The following architecture implementation were simulated and corresponding results shown:

- ◆ **Simplex Connection with Low speed return paths** - This mode of operation was implemented. When Tx is locked, the Rx is enabled via the LOCKED line. The Rx STAT outputs are zero, reflecting frequency acquisition mode, causing the local Tx, to send FF0. When Rx is frame locked, STAT1 is raised which disables frequency detector, sets itself to active mode and tells Tx to send FF1's. Upon receiving, FF1s enabling phase lock, STAT0 is raised, which enables the Tx ED input making link ready for data transmission.

Performance: The Tx and Rx models performed within the specified constraints of the model. The synchronous outputs were delayed as required. Also, though not clearly visible on the main waveform plot, the synchronous signals are delayed by 2ns and the STAT lines have been delayed by 4ns after the falling edge of STRBOUT. These are typical delay values being emulated. However, they are one frame clock faster than actual Chipset since the error module was not modeled, for reasons mentioned before/after. However, the inability to correct the data and, the associated loss of exactness do not, in any manner affect the functionality of the model. Figures 19 and 20 represent the simulation for this mode of operation.

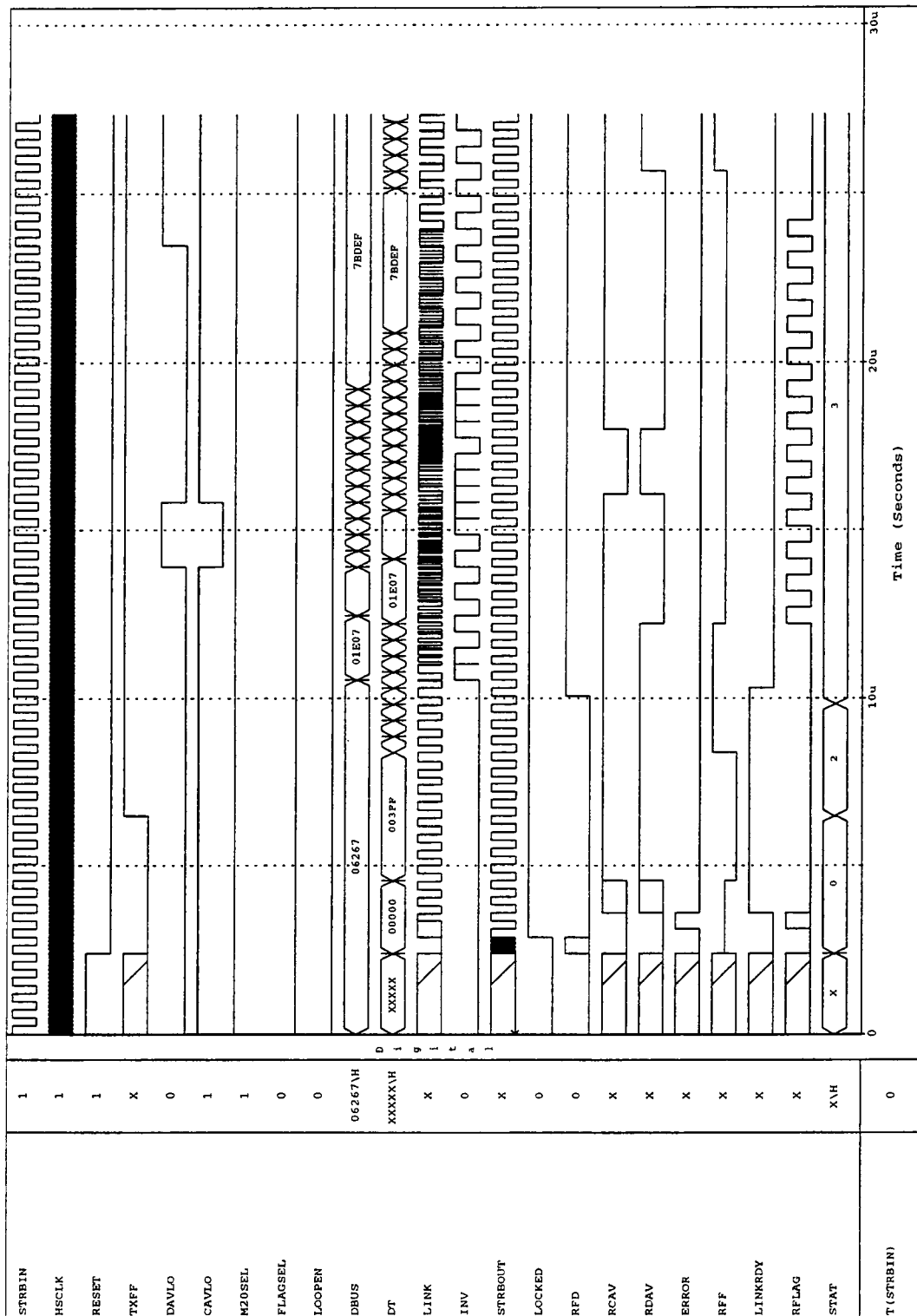


Figure 19 - Overall Simulation of Simplex 1 Link Configuration with Low-Speed Return Path

- ◆ **Simplex connection with periodic synch pulses** - This configuration of the Chipset was also modeled. Another configuration of operation is tested here with the models. Using the Tx chip's FF pin tied High the Rx achieves frame and phase lock. The ED pin is pulsed on a regular basis to maintain link. However, it is necessary that the ED pulse duration be long enough to establish the link. The necessary pulse duration can be estimated from typical lock up times for different frequency bands. The length of time between two ED pulses will determine how long the user needs to wait before the lock is re-established allowing. During ED pulse being low, data transmission is disabled since link lock is being checked and reestablished. Figures 21 and 22 represent the simulation data corresponding to this mode.

- ◆ **Simplex method with Reference Oscillator** - A modification will need to be done to the Rx resident SMC link establishment error conditions. This modification dealt with allowing data or control words to be received during the phase detection state (STAT = 2) also. It was decided not to include this modification considering the situation and a simple constraint put on the user to supply data accordingly. The required data being the use of FF1 fill frames until a reasonable amount of time for the Link to be established. This can be achieved by disabling the CAV / DAV lines and not using them for as long as the link takes to get established even though ED input is asserted HIGH.

The double frame mode has not been modeled or verified explicitly. The current model can be used in the double frame mode by using a frame clock that is half the double word frame. This will imitate the clock generator unit's feature of doubling the input frame clock upon the assertion of the MDFSEL pin.

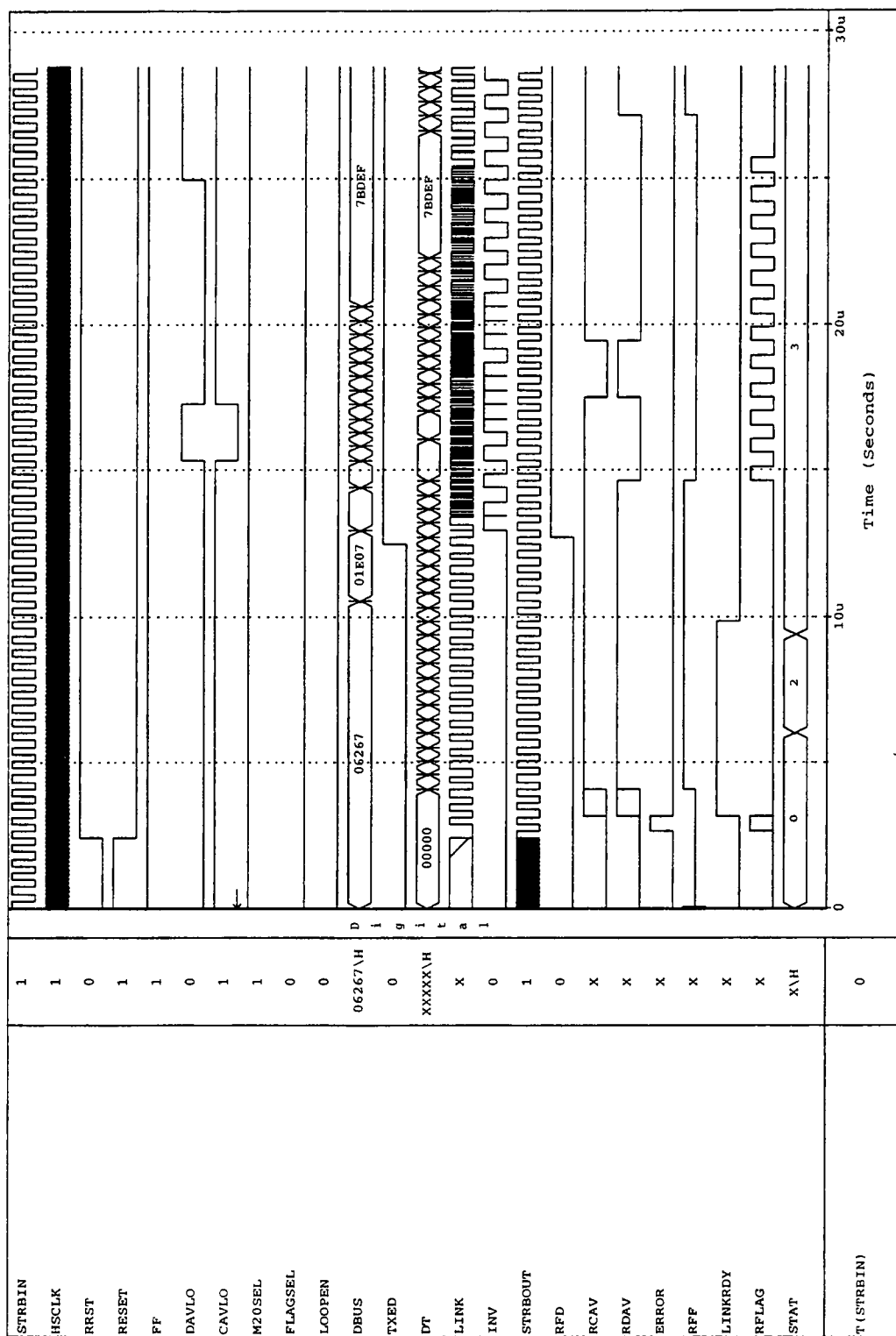


Figure 21 - Overall View of Simulation of Simplex Connection with Periodic Synch Pulses

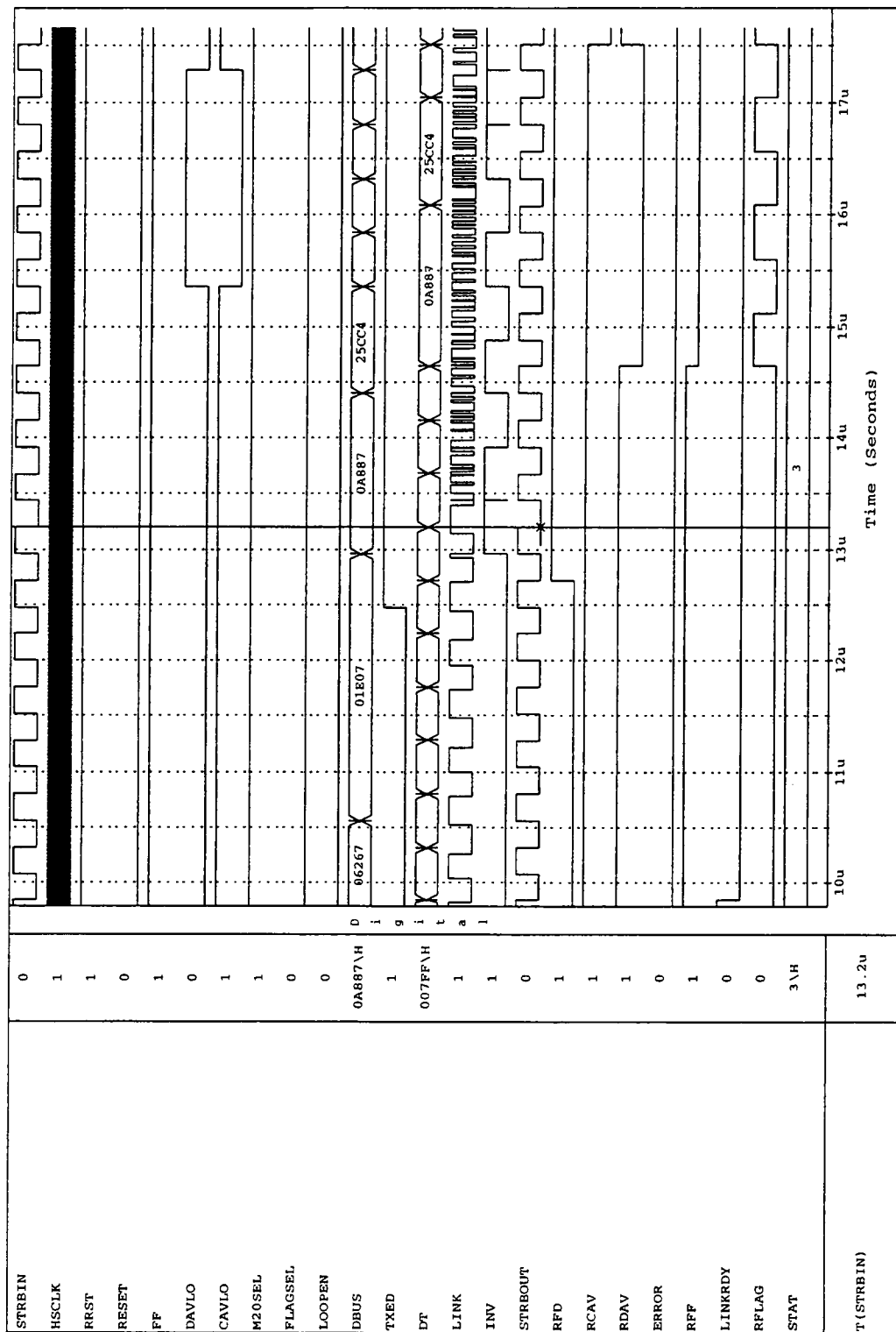


Figure 22 - Sample Zoom View of Simulation of Simplex Link Configuration with Periodic Synchronizing Pulses

5.3 Analog modules of the Chips

A simulation of an analog SPICE behavioral model has been incorporated here. This analog model allows for the Phase Locked Loop to be modeled and obtain a digital clock output. The analog model for the PLL has been incorporated from Meta Software HSPICE manual and has been used to show the possibility of Mixed mode Simulation Model.

Neither the complete mixed mode simulation model nor a complete digital model has been developed. Figure 23, 24 and 25 show different sections of a HSPICE simulation viewed in ViewTrace for the PLL analog behavior model.

5.4 Board Level Simulation

The models for the four major components have been integrated into a single design environment, namely, ViewLogic Powerview on SUN OpenWindows OS. Some peripheral logic might have to be added or modeled to complete the interconnectivity of the major components.

The lack of Xilinx libraries for the ViewLogic environment is hindering the board level netlisting of the components. After the libraries are installed, netlisting can be done as for any normal ViewLogic based design and simulated.

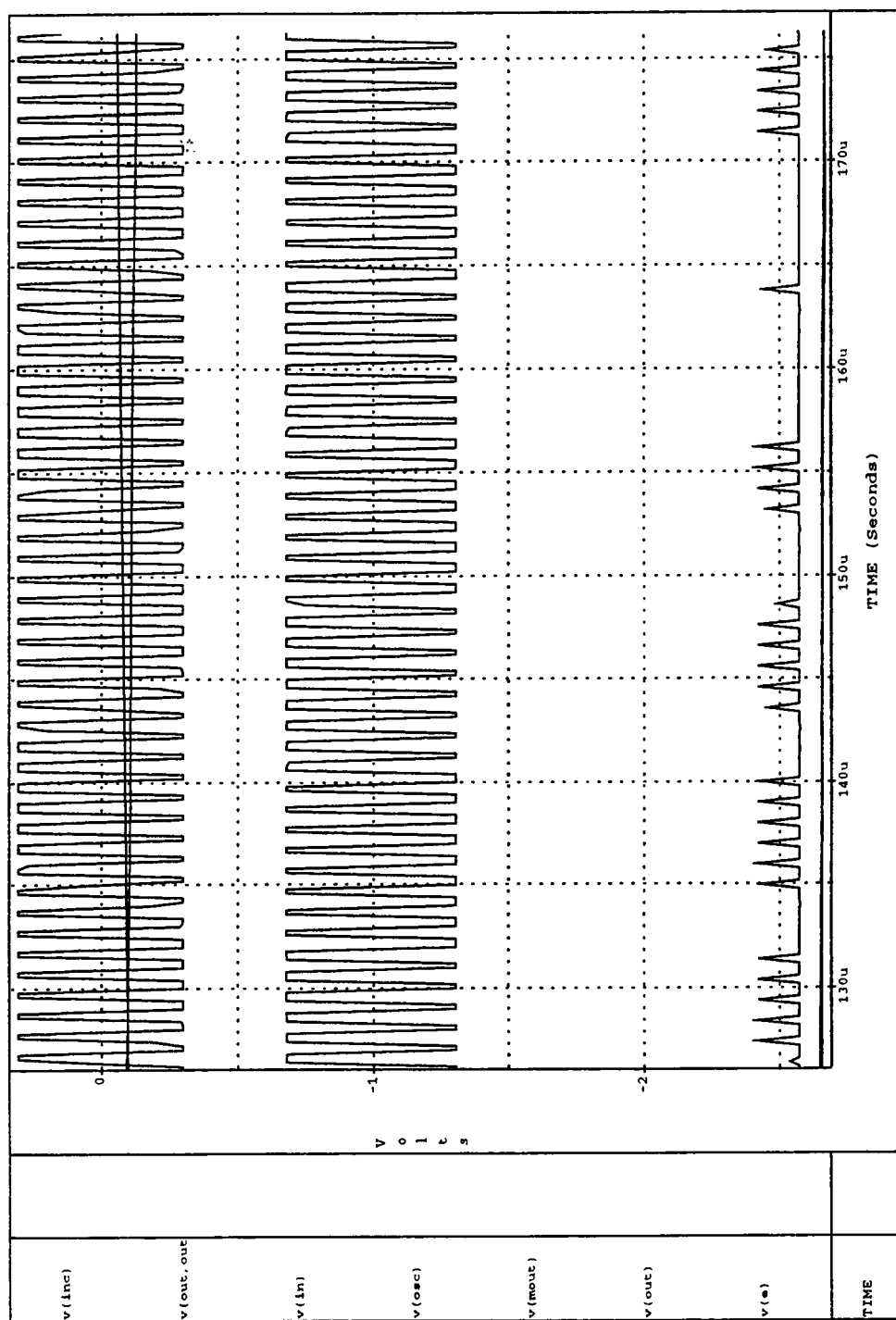


Figure 23 - PLL simulation - Before Locking to Input Frequency

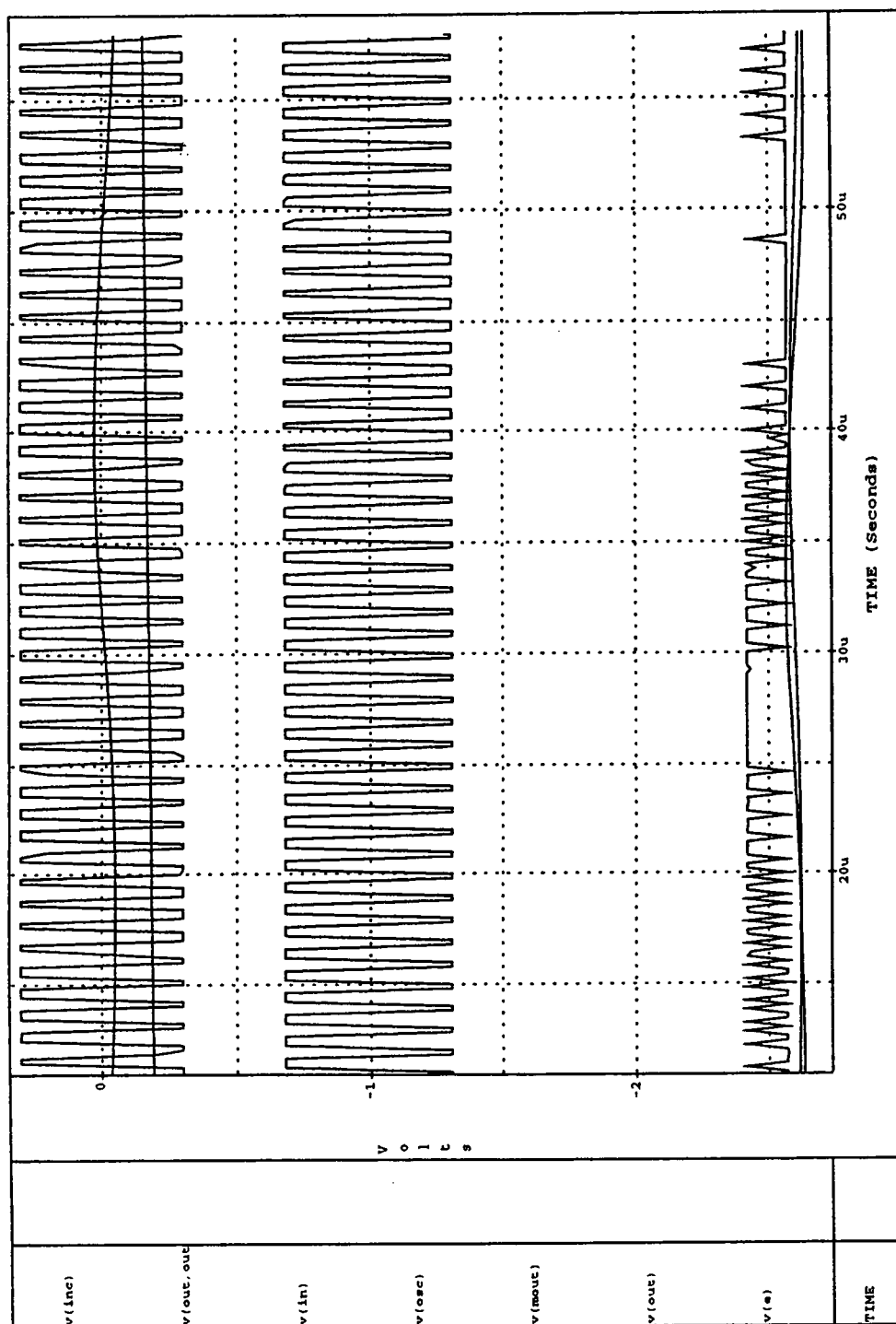


Figure 24 - PLL simulation - Just before Locking begins for the Dynamic Range

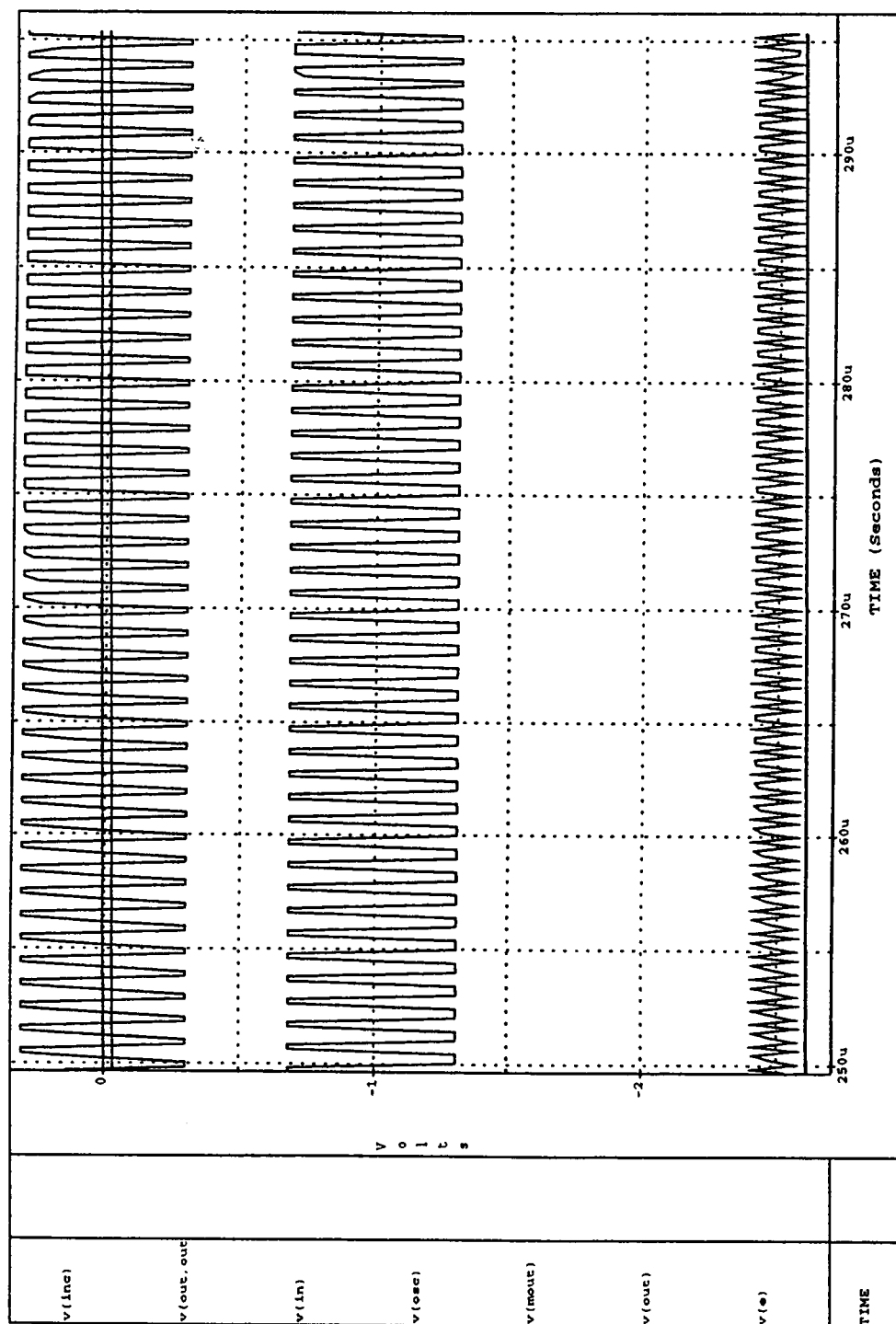


Figure 25 - PLL simulation - During Locked Condition

Chapter 6

Summary, Conclusions and Future Work

6.1 FPGA modeling

The state machine was designed for the Heap Manager FPGA in the Heap Manager daughter card with the available specifications. This design was tested and the operation was satisfactory and met design constraints. The methodology to make the design data available appropriately for simulation at the board level in Powerview was discussed and implemented. The design netlists for both the ALM FPGA and the Heap Manager FPGA were successfully imported into the ViewLogic environment. However, the digital simulator ViewSim does not load the **.vsm** simulation netlist correctly.

6.2 GLINK models

6.2.1 GLINK Receiver model

The GLINK receiver behavioral model was created in VHDL and was simulated for functionality in different configurations. The model performed satisfactorily in terms of functionality in the tested cases. There are some limitations to the model of the chipset since these were not coded. One of the limitations that stand out is the Error correction feature.

Timing information was included only to the extent that was described in the data sheets of the chipset. This timing information, for a behavioral model is sufficient since this model does not go into gate level or circuit level simulations.

Error Correction modeling: The modeling of error correction has not been done. It is also left out since error introduction is random in nature. The flagging of two continuous erroneous

frames received and resultant reset operation has not been modeled. This requires that the simulation model user, incorporate these as per needs, or else live with it.

A cycle delay trade off has been implemented. This is due the fact that an error correction module has not been added. This is by the assumption that the error occurrence is caused due to the link line being perturbed by random noise. Moreover, the error correction facility works for single bit errors only and that too for very few bit positions.

The architecture trade off makes the model for Rx faster by one frame clock. This results in a cycle inaccurate model when simulations are done. However, this does not affect the models working, in terms of functionality. In this pipeline architecture, the model has a latency of one frame clock from input to output instead of the two frame-clock delay of the Rx chip.

6.2.2 GLINK Transmitter model

The GLINK transmitter model was created at the behavior level with VHDL. The simulations of the various link configurations have been successful in showing that that the model operated with reasonable functionality of the original transmitter chip protocols.

Timing information was not included. The only available information on timing is the minimum setup time for the transmitter inputs and the delay between input and output frame clock. The former information has been transformed into another form in the model. This is an artificial delay of one clock cycle to sense the changes in the CAV or DAV control inputs once link is established.

6.2.3 Analog subsystem model

The analog subsystem model for both the transmitter and receiver was limited to a showcase SPICE behavioral model of a Phase locked loop circuitry. This model was not developed but was available previously. The interfacing of the analog subsystem model has been left out for the time. However, the method and interfacing requirements of A2D and D2A nets have been discussed. This interfacing method would be expected to be common to CAD tools supporting mixed mode simulation.

6.2.4 Board Level Simulation

The three main components can be simulated together once the FPGA components are imported for simulation. The simulation would allow for verification of functionality and some timing issues of the FPGAs since importing the designs for the programmable logic included the timing details.

In the present condition, the board has already been fabricated and is being used for design verification by iteratively debugging the design until design specifications are met. In normal circumstances, system functionality could be evaluated even before the prototype boards need to be fabricated.

6.3 Scope for future work

The scope for future work with regard to improving these models exist. Some of them include

- ◆ Error correction feature.
- ◆ Timing information on the internal nets of the models to simulate internal delays.
- ◆ Interfacing of the analog subsystems with the digital subsystems.
- ◆ Correction of model to achieve cycle accuracy and independence from frame count.

- ◆ Modification of the model to make it simulation resources efficient.
- ◆ Electrical simulation model

Each of the above points is discussed briefly, in the following paragraphs, to provide a possible solution or suggestion.

6.3.1 Error correction feature

Error correction feature can be easily added to the state machine module resident in the receiver chip model. The input data pins are available. However, if output pins are needed, these will have to be added. Also, the data flow from the frame generator module will have to be diverted to the state model so as to correct errors, if any, before decoding process begins.

Also, the error flagging and reset operation on the occurrence of two consecutive data frame errors will have to be added. This will be an extension of the error module described above.

6.3.2 Timing information on internal nets

This information added will be mainly intuitive based on some theoretical or guesstimates. This is so because of the many possible values on each net in the actual chipset. However, this addition, will provide some delay, which probably is the best that can be achieved.

6.3.3 Analog subsystems model interface

The analog subsystems model interfaces (inputs and outputs) include some pins such as the ACTIVE, FDIS, PHI , STRBOUT and so forth. The inclusion of these signals in the simulation model will go a long way in developing a true mixed simulation model.

6.3.4 Cycle accurate model

Cycle accurate model can be achieved, for most parts, once the error correction feature is coded.

This should typically, result in a cycle equivalency in terms of clock cycle delays.

6.3.5 Simulation Resource efficiency

This can be done only after looking into the present model's memory requirements, evaluation of the model code for simulator inefficient coding and modifying them. The model probably does not use much memory that dynamic memory-management techniques be incorporated into the model code. A wrapper code for this model that makes the models usage more user friendly would be helpful. Some examples, would be the inclusion of internal signal and variable monitors so as to aid debugging of system designs using this model.

6.3.6 Electrical Simulation model

The development of electrical simulation model would be beneficial. Once included, the model could be used with CAD tools that enable electrical and signal integrity simulation for MCM's and, possibly, other systems on chip designs.

References

1. XC4000E and XC4000X Series Field Programmable Gate Arrays, November 10, 1997,
(Version 1.4), Xilinx, Inc.
2. Field Programmable Gate Arrays: Reconfigurable Logic for Rapid Prototyping and
Implementation of Digital Systems – Oldfield, John V., Dorf, Richard C., John Wiley & Sons,
Inc, 1995, pp. 18.
3. ViewSpice Users' Guide: Mixed Signal Design with ViewSpice, (pp. 8-5 – 8-10), ViewLogic
User's Guide.

Appendix

SPICE BEHAVIOR MODEL LISTING:

Following is the listing of the behavior model used from the HSPICE manual for simulating a Phase Locked Loop circuit.

```
$ Phase locked loop
.option probe acct
.option CSDF=2 INGOLD=2
.option relv=1e-5
$
$ .option CSDF=2 INGOLD=2 added to produce ViewTrace Readable format output from HSPICE.
$ wideband FM example, Grebene gives:
$ f0=1meg kf=250kHz/V
$ kd=0.1 V/rad
$ R=10K C=1000p
$ f_lock = kf*kd*pi/2 = 39kHz, v_lock = kd*pi/2 = 0.157
$ f_capture/f_lock ~= 1/sqrt(2*pi*R*C*f_lock) = 0.63, v_capture ~= 0.100

*.ic v(out)=0 v(fin)=0
.tran .2u 500u
.option delmax=0.01u interp
.probe v_in=v(inc,0) v_out=v(out,outb)
.probe v(in) v(osc) v(mout) v(out) v(e)

vcc vcc 0 6
vee vee 0 0

$ input
vin inc 0 pwl 0u,-0.2 500u,0.2
xin inc 0 in inb vco f0=1meg kf=125k phi=0 out_off=0 out_amp=0.3

$ vco
xvco1 e eb osc oscb 0 vee vco1
.ic v(osc)=-1.4 v(oscb)=-0.7

$ phase detector
xpd1 in inb osc oscb mout moutb vcc vee pd1

$ filter
rf mout e 10k
cf e 0 1000p
rfb moutb eb 10k
cfb eb 0 1000p

$ final output
rout out e 100k
cout out 0 100p
routb outb eb 100k
coutb outb 0 100p
```

```
.macro vco in inb outb f0=100k kf=50k phi=0.0 out_off=0.0 out_amp=1.0
gs 0 s poly(2) c 0 in inb 0 '6.2832e-9*f0' 0 0 '6.2832e-9*kf'
gc c 0 poly(2) s 0 in inb 0 '6.2832e-9*f0' 0 0 '6.2832e-9*kf'
cs s 0 1e-9
cc c 0 1e-9
e1 s_clip 0 pwl(1) s 0 -0.1,-0.1 0.1,0.1
e out 0 s_clip 0 out_off '10*out_amp'
eb outb 0 s_clip 0 out_off '-10*out_amp'
.ic v(s)='sin(phi)' v(c)='cos(phi)'
.eom
```

```
.macro pd in inb in2 in2b out outb kd=0.1 out_off=0
e1 clip1 0 pwl(1) in inb -0.1,-0.1 0.1,0.1
e2 clip2 0 pwl(1) in2 in2b -0.1,-0.1 0.1,0.1
e3 n1 0 poly(2) clip1 0 clip2 0 0 0 0 0 '78.6*kd'
e4 outb 0 n1 0 out_off 1
e5 out 0 n1 0 out_off -1
.eom
```

```
.macro vco1 in inb e7 e8 vcc vee vco_cap=228.5p
qout vcc vcc b7 npn1
qoutb vcc vcc b8 npn1
rb vcc c0 5k $ 1ma
q0 c0 b0 vee npn1
q7 vcc b7 e7 npn1
r4 vcc b7 1k
i7 e7 0 1m
q8 vcc b8 e8 npn1
r5 vcc b8 1k
i8 e8 0 1m
q9 b7 e8 e9 npn1
q10 b8 e7 e10 npn1
c0 e9 e10 vco_cap
q11 e9 in 2 npn1 $ ic=i0
q12 e10 in 2 npn1 $ ic=i0
q15 2 c0 b0 npn1 $ ic=2*i0
q16 3 c0 b0 npn1 $ ic=2*i0
rx 2 3 8k
q13 vcc inb 3 npn1
q14 vcc inb 3 npn1
rt b0 vee 350 $ i=4*i0=2m
.eom
```

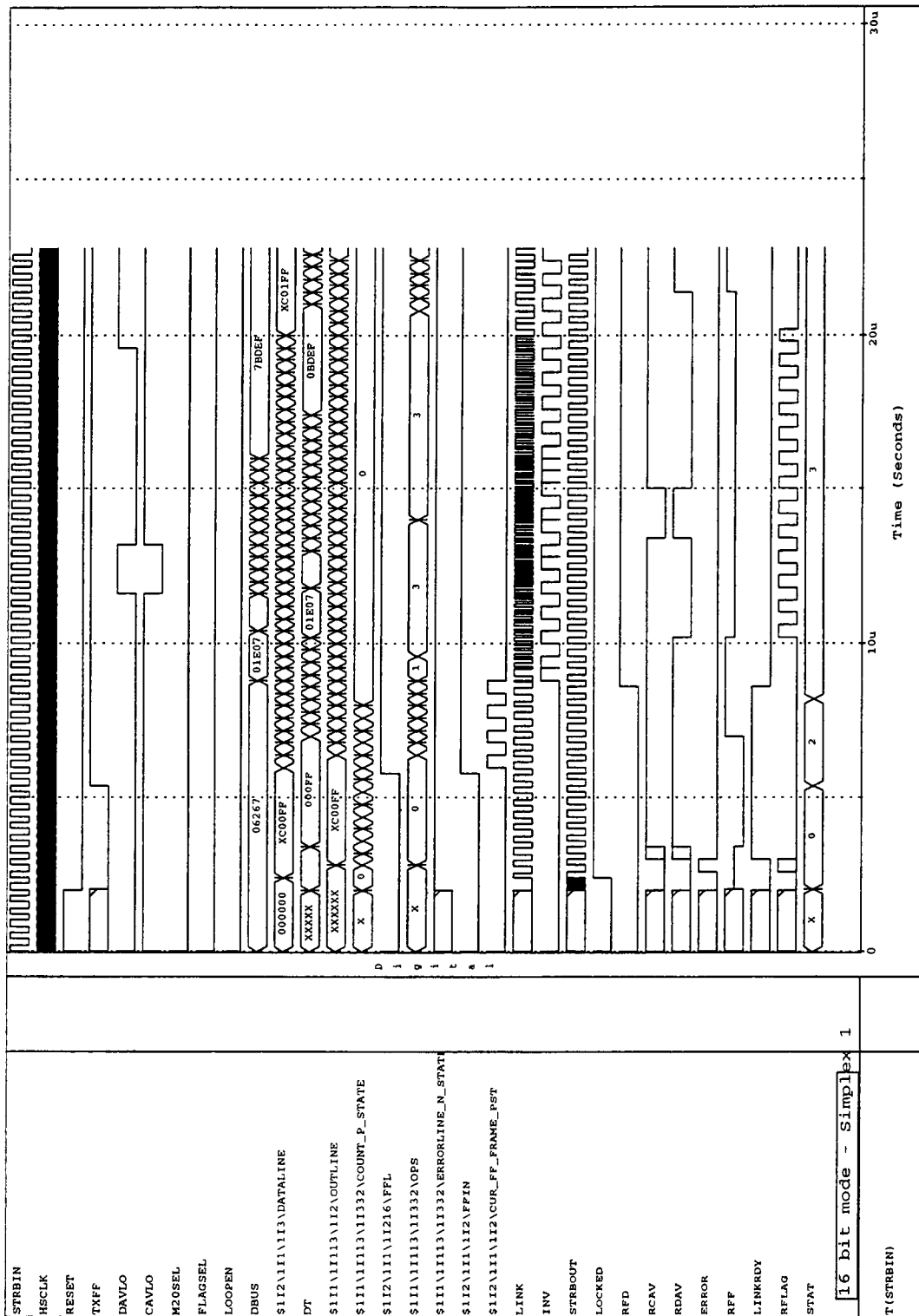
```
.model npn1 npn eg=1.1 af=1 xcjc=0.95 subs=1 cjs=0 tf=5p tr=500p cje=0.2p cjc=0.2p fc=0.8
+ vje=0.8 vjc=0.8 mje=0.33 mjc=0.33 rb=0 rbm=0 irb=10u is=5e-15 ise=1.5e-14 isc=0
+ vaf=150 bf=100 ikf=20m var=30 br=5 ikr=15m rc=0 re=0 nf=1 ne=1.5 nc=1.2 tbf1=8e-03
```

```

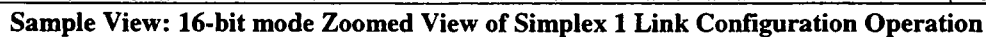
.macro pd1 in inb in2 in2b out outb vcc vee
r1 vcc n1 1k
r1b vcc n1b 1k
q3 n1 in c1 npn1
q4 n1b inb c1 npn1
q5 n1 inb c2 npn1
q6 n1b in c2 npn1
q1 c1 in2 e npn1
q2 c2 in2b e npn1
ie e 0 0.5m
c1 n1 0 1p
c1b n1b 0 1p
q7 vcc n1 e7 npn1
q8 vcc n1b e8 npn1
r1 e7 out 625
r2 out vee 300
r1b e8 outb 625
r2b outb vee 300
.eom

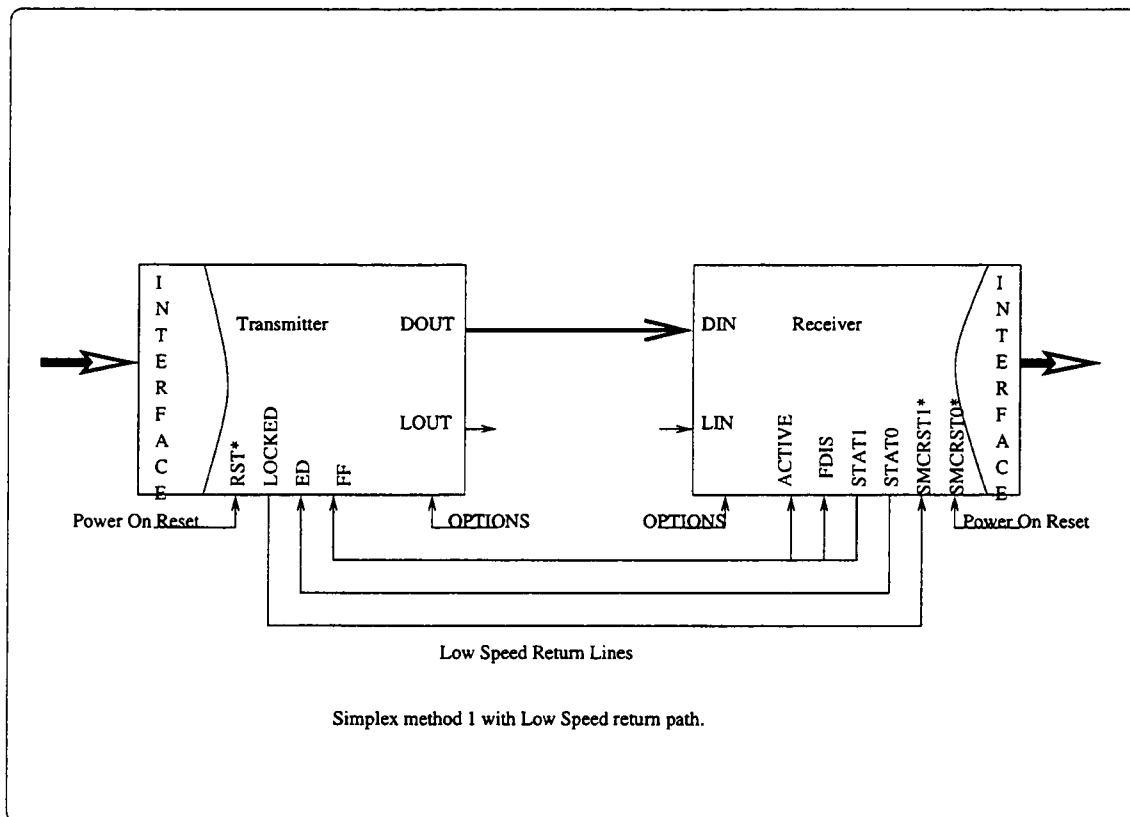
.end

```

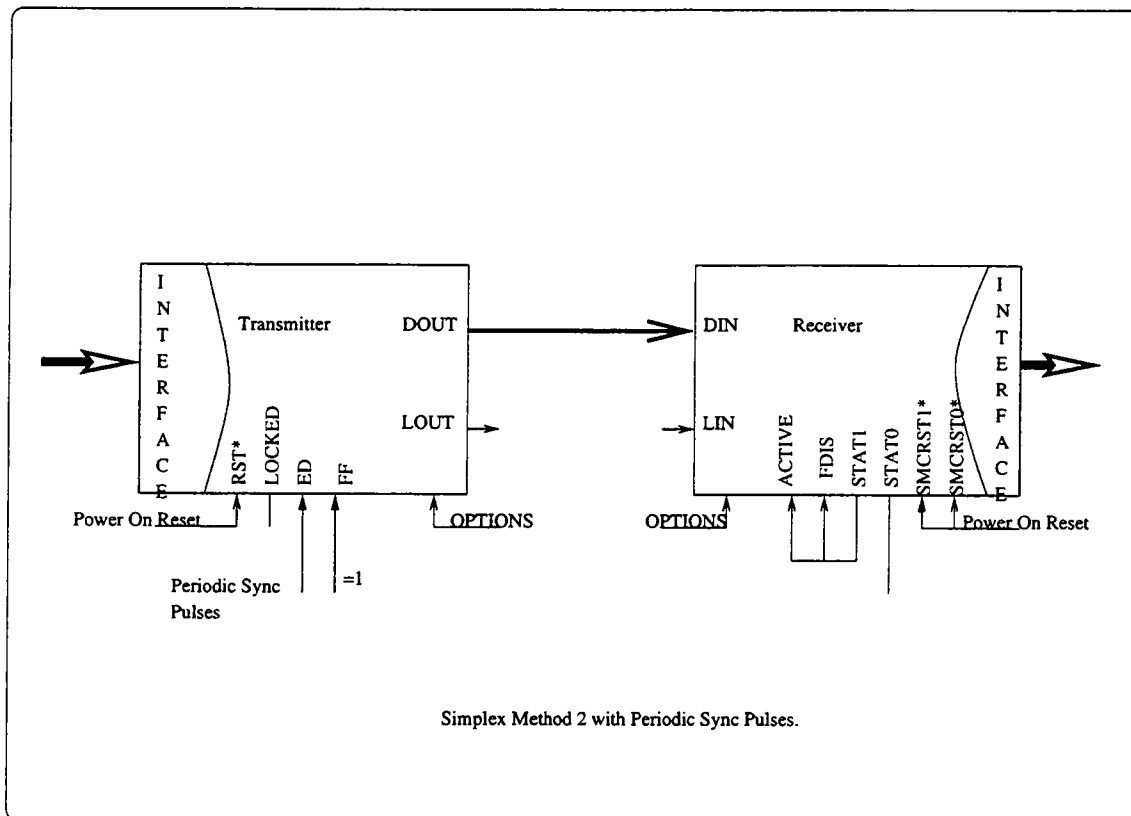


Full View: 16-bit Mode Operation for Simplex 1 with Low Speed Return Path

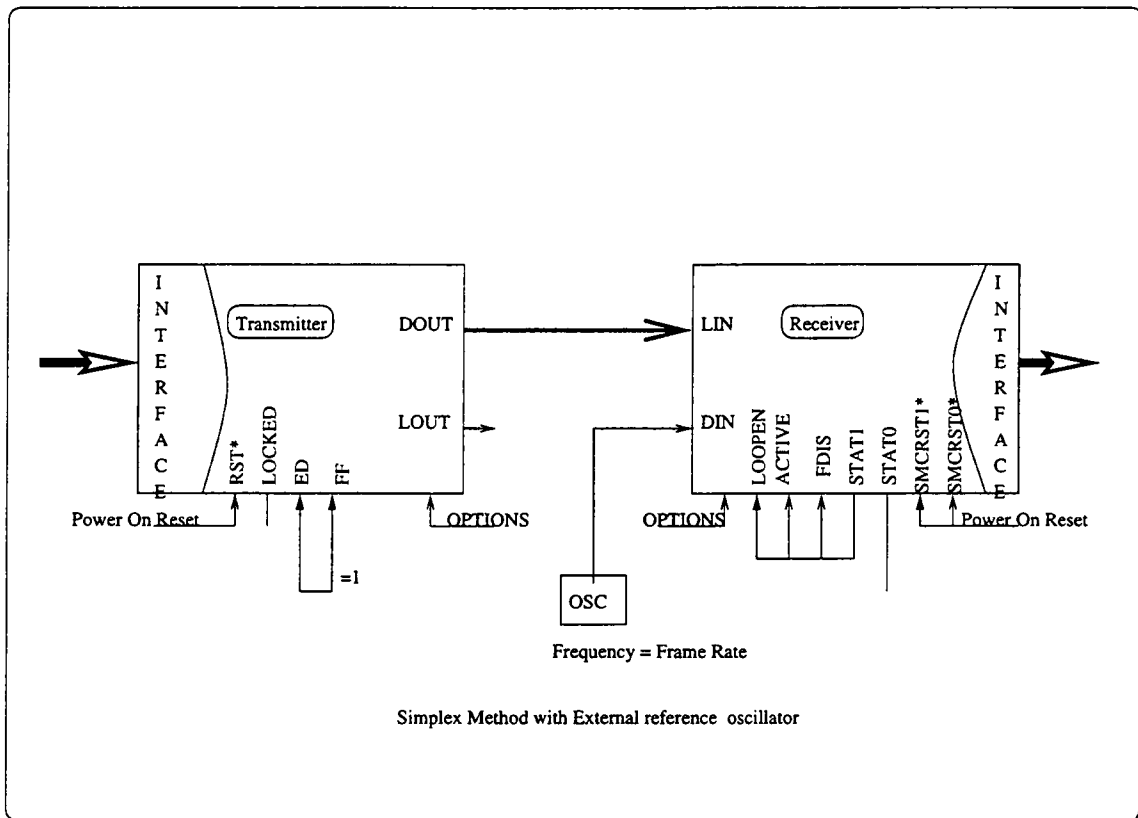




Simplex 1 configuration with Low Speed Return Paths



Simplex Connection with Periodic Synch Pulses



Simplex Connection with Reference Oscillator

Frame Word Structures

Control Frame Word Structure

M20SEL not asserted: 16 bits mode

D – Field			C – Field
D0 – D6	D7 – D8	D9 – D16	C3 – C0
D0 – D6	01	D7 – D13	1100
NOT(D0 – D6)	10	NOT(D7 – D13)	0011

M20SEL asserted: 20 bits mode

D – Field			C – Field
D0 – D8	D9 – D10	D11 – D19	C3 – C0
D0 – D8	01	D9 – D17	1100
NOT(D0 – D8)	10	NOT(D9 – D17)	0011

Data Frame Word Structure

M20SEL not asserted: 16 bits mode

Data Status	Flagbit	D – Field	C – Field
True	0	D0 – D15	1011
Inverted	0	NOT(D0 – D15)	0100
True	1	D0 – D15	1101
Inverted	1	NOT(D0 – D15)	0010

M20SEL asserted: 20 bits mode

Data Status	Flagbit	D – Field	C – Field
True	0	D0 – D19	1011
Inverted	0	NOT(D0 – D19)	0100
True	1	D0 – D19	1101
Inverted	1	NOT(D0 – D19)	0010

Fill Frame Word Structure

M20SEL not asserted: 16 bits mode

Fill Frame	D – Field (D0 – D15)			C – Field
0	1111111	10	0000000	1100
1a	1111111	11	0000000	1100
1b	1111111	00	0000000	1100

M20SEL asserted: 20 bits mode

Fill Frame	D – Field (D0 – D19)			C – Field
0	11111111	10	00000000	1100
1a	11111111	11	00000000	1100
1b	11111111	00	00000000	1100

VITA

Badhri Narayanan Uppiliappan was born in Ernakulam, Kerala, India in 1975 and graduated from High School in May 1992. He then entered the Regional Engineering College, Tiruchirappalli, Tamil Nadu, India in August 1992 where he obtained his Bachelor of Engineering in Electrical and Electronics Engineering (First Class with Distinction) in May 1996. He served as a student executive member of the Electrical and Electronics Engineering Association during 1993 - 1994. He was a founding editor of the departmental newsletter, "TRENDS", in 1994 and was Chief Editor in 1995 - 1996. He served the coordination committee for a National Level Student's Symposium, "CURRENTS '96", organized by the Electrical and Electronics Engineering Association of Regional Engineering College, Tiruchirappalli, Tamil Nadu, India.

In August 1996, he entered The University of Tennessee, Knoxville, Tennessee, U.S.A. and obtained his M.S.E.E. in December 1998. He held the position of President, Indian Students Association at the University of Tennessee, Knoxville during 1997 - 1998.