Optical Bus Communication Modeling and Simulation

Brian J. d'Auriol
Department of Computer Science
The University of Texas at El Paso
El Paso, TX, USA 79968
Email: dauriol@acm.org

Maria Beltran
CAS Inc.
El Paso, TX, USA
Email: maria.beltran@cas-west.com

***Abstract*— Multiple independent communications are studied on optical bus parallel computing models. These models make use of an optical bus interconnect to provide for communication in a multi-processor system. The communication traffic studied in this paper results in potential frequent bus collisions. The well known collision detect mechanism is ineffective however due to the communication pipelining aspect of the optical bus. An analysis of the conditions necessary for bus collision and the subsequent communications model that represents the necessary information to determine potential bus collisions are presented. A simulation based on this model as well as results from the simulation are discussed. The results indicate collision patterns and trends in sequences of general communications. The model and simulation considered in this paper establishes a mechanism that enables future analysis of general communication usage of optical buses.**

I. Introduction

Parallel computational models provide a tool to design and analyze algorithms for multiprocessor computers. Many parallel models have been proposed in the literature, for example, PRAM. The optical parallel bus model is an emerging type of parallel computational model that takes advantage of optical communications technology in its interconnection structures. Optical bus models provide for improvements in terms of speed and bandwidth over other types of interconnection networks used in multiple-processor systems. The growing popularity of these optical bus models is reflected in the more than seventy-five publications in the area in the past decade. An extensive bibliography appears in [1].

Optical bus parallel models consist of an array of processors that communicate through an optical bus interconnection network. Most such models have a one dimensional linear topology of processors, although multi-dimensional configurations also are possible. The interconnection network uses one or more waveguides, i.e. optical fibers. Specific waveguides are allocated for messages or for addressing.

Optical bus models represent an improvement over conventional electrical bus parallel models. Advantages include: pipelined concurrent access as opposed to exclusive write access, low error probability and gigabit transmission capacity [2]. Two important properties of light that lead to these advantages are the unidirectional propagation and predictable propagation delay per unit length. We note that some of these advantages as described in the literature may now be somewhat offset due to technological developments in newer electrical buses at very high frequencies.

Many algorithms have been devised for various optical bus models. Such algorithms share a common characteristic, that bus communication occurs in a defined communication phase within particular bus cycle time frames. One major reason for this is that bus collisions due to multiple independent communications are eliminated. However, many of the optical bus models allow for more general communications, for example, as would be encountered in MIMD operations. Collision detect, a well known solution to bus contention, does not work with optical buses due to the inherent pipelining of the communication. This is because a communication could be already on the bus, but 'behind' the processor initiating a new communication.

This paper considers a communication model that specifies the necessary conditions for bus collisions. In particular, a simulation of the optical bus communications is developed and some preliminary results from the simulation regarding bus collisions are presented in this paper. We believe that this represents the first simulated study published regarding bus collisions arising from multiple independent bus accesses.

This paper is organized as follows. An optical bus parallel model is discussed in Section II. This model abstracts many of the common features found in the models proposed in the literature. The simulation is discussed in Section III. Conclusions are given in Section IV.

II. An Optical Bus Parallel Model

The proposed optical bus model used for the simulation is a generic model that captures many of the most common characteristics of the optical buses described in the literature. Table I lists the buses that have been proposed in the literature. The folded bus using the coincident pulse addressing technique is a common architecture found in many of the existing models.

A folded bus architecture consists of a linear array of processors interconnected via one or more waveguides. Processors are connected to a waveguide at two points: to the transmitting segment via light injectors and to the receiving segment via light detectors. The processor at one end of the array is

nearest the fold. The fold describes the bend in the waveguide that directs light pulses from the transmitting segment to the receiving segment. Fig. 1 illustrates the fold architecture.

In coincident pulse addressing, fixed delays of duration ω are placed between every pair of detectors on the reference and message waveguides. First, the source processor sends a pulse on the reference waveguide together with message pulses on the message waveguide. Then, the processor waits integral ωs, depending on the destination processor it wishes to communicate with, to send a pulse on the select waveguide. The pulse on the reference waveguide is delayed by the delays on the receiving segment such that it will be detected by the intended destination processor at the same time as the select pulse. At that point, the message (which also arrives at the intended processor at the same time) is read. Therefore, to address the processor nearest to the fold, zero delays are introduced whereas to address the next processor, a delay of one ω is introduced.

The generic model comprises the following six characteristics; the first five of which abstract common elements from the existing models, while the sixth describes a new communications model. The communications model is detailed subsequently.

- Folded Bus: The folded bus is commonly used from 1996 onwards. Optical bus models incorporating the folded bus are ASOS, RASOB, LPB, AROB, LARPBS, POB, and LAPOB.
- Coincident pulse: The coincident pulse technique is the preferred mode of addressing in APPB, APPBS, LARPBS, LAPOB, LPB, AROB, and POB.
- Optical pulse: All the models surveyed use digital signals where the presence of an optical pulse denotes a 'one' and the absence, a 'zero'.
- Linear Array: The one dimensional linear array is highly popular.
- Fixed delays: Coincident pulse is implemented by the use of fixed delays on the receiving segment of the folded bus.
- Communications Model: A communications model is developed and associated with the generic optical bus architecture.

The generic model is made up of three folded waveguides (the folded bus) that are referred to as the select, reference and message waveguides, respectively. The former two waveguides are used to implement the coincident pulse technique of addressing. The processors are arranged in a linear order and are labeled $P_{N-1}, ..., P_0$ where P_0 is placed nearest to the fold. The propagation time between any pair of injectors or any pair of detectors, τ, is a constant. The propagation time for the folded segment of the bus, γ, is another constant. There are $N-1$ fixed delays, each of which are ω time units in duration, and are placed on the receiving segment of the reference and message waveguides. The delays are located between pairs of detectors. Two necessary preconditions are:

- τ is greater than $(N-1)\omega$ in order to ensure a proper broadcast operation. A similar condition is set in [9].

TABLE I

OPTICAL BUS MODELS PROPOSED IN THE LITERATURE

Name	Acronym
Array of Processors with Pipelined Buses [3]	APPB
Array of Processors with Pipelined Buses Using Switches [3]	APPBS
Array Structure with Synchronous Optical Switches [4]	ASOS
Linear Pipelined Bus [5]	LPB
Reconfigurable Array with Spanning Optical Buses [6]	RASOB
Array with Reconfigurable Optical Buses [7]	AROB
Linear Array with a Reconfigurable Pipelined Bus System [8]	LARPBS
Pipelined Optical Bus [9]	POB
Linear Array of Pipelined Optical Buses [10]	LAPOB
Pipelined Reconfigurable Mesh [11]	PR-MESH

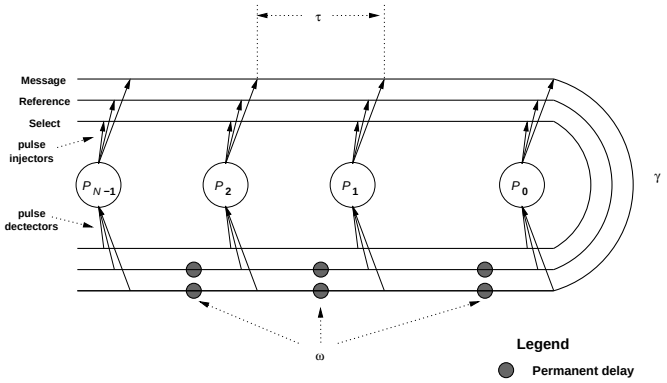

Fig. 1. Proposed Generic Optical Bus Model

- As in the literature, one simple case of collision is addressed by defining b as the maximum size of a message such that $b\omega < \tau$.

Fig. 1 illustrates this model, with the transmitting segment shown at the top and the receiving segment shown at the bottom.

In addition, a *reference clock* is introduced. The reference clock is defined as a clock that is global to the generic bus model network. Each processor is able to access the reference clock in order to measure the passage of time.

A single communication event C is defined as a single source processor $P_s, 0 \leq s \leq N-1$, sending a single message to one or more destination processors, $P_{d_0}, ..., P_{d_M}, 0 \leq M, 0 \leq d_M \leq N-1$. Recall that pulses on each of the three waveguides are required to accomplish such a communication. The triple (r, S, m) models a communication C and is composed of values from one of two time coordinate systems that are described subsequently: r denotes when P_s injects a single pulse on the reference waveguide, m denotes when P_s starts to inject the message to be sent, and $S = (s_0, ..., s_A)$ denotes when P_s injects one or more pulses onto the select waveguide, with A being the number of select pulses injected. For the case of a unicast communication $|S| = 1$. For the case when a single processor sends the same message to more than one processor (i.e., a multicast or broadcast) $|S| > 1$. Note

that in all cases, $|S| > 0$, and r and m cannot be null.

Multiple communications are produced by various source processors communicating. These are modeled by adding triples to a set T called the *active communication set*: $T = \{(r_0, S_0, m_0), (r_1, S_1, m_1), ..., (r_I, S_I, m_I)\}, 0 \leq I$, or alternatively $T = \{C_0, C_1, ..., C_I\}$. T provides a snapshot view of all active communications on the bus. A group of select pulses S_i, $0 \leq i \leq I$, can be alternately referred to as $S_i = (s_{i,0}, ..., s_{i,A})$. An event C is removed from T only when all of its pulses leave the bus.

Two time-coordinate systems are proposed to model the chronological progress of the optical pulses along the waveguides. *Processor time*, denoted by superscript p, is the time measured with respect to the reference clock at which a communication event occurs. The origin of the processor time is the time when the very first communication event is injected onto a bus with no previous communication. The second time coordinate system, *waveguide time*, denoted by superscript w, represents communication events relative to other communications on the bus. The origin of waveguide time corresponds to the case when the very first communication is injected by P_0.

Any communication C_j^p, however, is generated by a processor that has no knowledge of existing communication events and their place on the bus. Given $T = \{C_0^w, ..., C_i^w\}$ and C_j^p, the components of C_j^p need to be transformed from processor time into waveguide time. Each waveguide time for C_j^p can be computed given each processor time from C_j^p and the index of the generating processor P_s where C_j^p was injected: $f(t^p, s) \mapsto t^w$, where $f(t^p, s) = t^p + s\tau$ and the inverse $f^{-1}(t^w, s) \mapsto t^p$, where $f^{-1}(t^w, s) = t^w - s\tau$.

An example is given. Let $C_0^p = (0, (\omega, 2\omega), 0)$ be a communication event consisting of a multicast from P_0 to P_1 and P_2. Let $C_1^p = (\omega, (3\omega), \omega)$ be a communication event consisting of a unicast from P_1 to P_3. Initially $T = \{C_0^p\} = \{(0, (\omega, 2\omega), 0)\}$. Let C_1^p be an event that P_1 initializes. The result of adding C_1^p to T is $T' = \{(0, (\omega, 2\omega), 0), (\omega, (3\omega), \omega)\}$. Converting to waveguide time, $T' = \{C_0^w = (0, (\omega, 2\omega), 0), C_1^w = (\tau + \omega, (\tau + 3\omega), \tau + \omega)\}$.

Signals can interfere with each other in two ways: physical collisions and incorrect coincidence. With respect to incorrect coincidence, there are only four cases to consider. This is because incorrect coincidence behavior depends only on the select and reference waveguides. The following discussion details these cases. In all cases, C_j is a communication to be introduced into a bus that already contains a single communication C_i, i.e., $T = \{C_i\} \bigcup C_j$. Subsequent, the situation with respect to physical collisions is described.

Conflict can only occur when two optical signals coincide in such a way that incorrect communication occurs. Such a conflict can only occur as detailed by the following four cases.

1) One waveguide with no delays: In this simplest case, if C_j is deemed as safe upon an injection, then the propagation of signals remains constant for the signals that make up C_i and C_j; and signals maintain their positions in relation to each other. Recall that the signal propagation is constant throughout a waveguide. The state of non-collisions is maintained on the bus.
2) One waveguide with delays: This case is analogous to the first one. The signals that compose C_i will be delayed by ω time units when they encounter a delay. The spacing between C_i's signals and those of C_j will change, but the original spacing will be restored when the signals from C_j are also delayed by ω upon encountering the same delay. The state of non-collisions is maintained on the bus.
3) Two waveguides with no delays: This case is an extension of Case 1. The state of non-collisions is maintained on the bus.
4) Two waveguides with delays on one: This case is more complex than the previous ones. If signals from C_i are traveling on one waveguide while signals from C_j are traveling on the other, the positions of signals from C_i and C_j in relation to each other will change across waveguides due to the delays. This may cause unintended or incorrect coincidence of signals at processor detectors, which in turn causes incorrect routing of communications.
   The incorrect coincidence of signals can occur in one of the two following ways:
   a) The number of delays on a waveguide causes the relative positions between C_i and C_j to be such that the relative delay time between them eventually exceeds τ.
   b) The arbitrariness of the injection time of C_j could lead, upon the effect of one or more delays, to the situation that a select pulse of C_i coincides with a reference pulse of C_j, or vice versa. See Fig. 2 for an example.

The addition of a new communication event to T will not change the time values for the signals previously existing in T. That is, each communication event C is independent of any other communication event.

In considering how to check for problems related to physical collisions, there are two different points in time that are identified as significant when a pulse or group of pulses passes through an injection point:

1) the moment when the pulse or group of pulses starts to pass the injection point, and
2) the moment when the pulse or group of pulses finishes passing the injection point

In order to avoid a collision on the bus from pulses injected at the injection point, the pulse or group of pulses must be completely injected before Condition 1 occurs, or start to be injected anytime after Condition 2 occurs.

In summary, the parameters of the communication model are: τ (separation between processors), ω (length of unit delay), γ (time to traverse the optical bus fold), $b\omega$ (b being the maximum number of pulses in a set), and N.

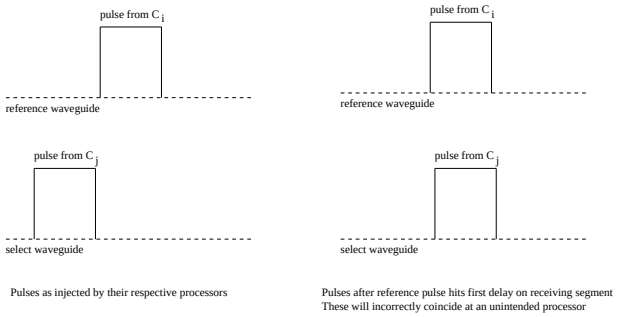

Fig. 2. Example of Incorrect Coincidence

III. Implementation and Simulation

An optical bus communication simulation is developed to provide for an analysis of bus communications. The simulation is based on the communications model described in the previous section. The precise specification for the collision conditions appears in [1] and is omitted here for space reasons.

The simulation provides us the immediate ability to study and analyze bus collision patterns arising from random collective communication operations such as unicast and broadcast. This provides a 'first-approximation' to assessing the collision impact that general algorithms would experience. Further enhancements to the simulation that take into account predicted collision behavior based on a specific algorithm would, we feel, better characterize the collision impact.

Coding details are described in Section III-A. Simulation results are presented in Section III-B. The parameters from the previous section are held constant in the simulations: $\tau = 50, \omega = 4$ and $N = 10$. These parameter values are assumed since optical bus parallel computing models are theoretical models.

A. Coding Details

Coding is done in a Windows environment, using a `gcc` compiler running on the Cygwin Linux emulator. Two `C` programs are written as part of the simulation: the *generator* and the *checker*. A single communication event, C, is represented by a struct data type. The event set T is represented by a linked list called made up of these structs. A 'clock' variable is used to simulate the progression of time.

The generator program generates simulated sets of bus communications according to a policy, for example, consisting of only unicast events, only broadcast events, only multicast events, or a mix of all three types. The output is placed into a flat text file in which the events are sorted by processor time. This file serves as input to the checker process. The file is also in human readable format and therefore may be hand crafted. Details about the file format are given in Section III-A.1.

The checker program implements the safe communication checks presented earlier. It reads in the file from the generator process. For every event in the file, the checker performs the following:

1) Read C into the single communication data structure.
2) "Fast-forward" time by setting the clock variable to the processor time for the reference pulse component from C.
3) Reference the clock in order to determine which communications have expired, that is, have traversed the length of the bus. All such communications are removed from the event set linked list.
4) Call the safe communication check function with the event set, T and the single (new) communication, C as inputs.

The safe communication function performs the checks in the order: coincidence at the wrong processor, reference overlap, select overlap and message overlap. The new event `C` is checked against each existing event in `T`. If `T` is found to be safe when including `C`, then `C` is added to `T`, otherwise `C` is not added. Upon encountering a check that fails, the checker process immediately returns with an unsafe result, and the remaining checks are not done. The algorithm is therefore linear in the number of communications on the bus.

The output of the checker program provides details concerning communications deemed unsafe by the communication model. Details about interpreting the output are given in Section III-A.2.

1) File Format: The first line of the file always contains the number of communication events in T. There is one communication event per line. The first number in each line is the processor that initiates the event, followed by the reference component of the event, then the select vector component enclosed by brackets, then the message component and finally the length of the message component. Times are given in processor time and are sorted. Refer to Fig. 3 for an example of the file format. Consider the fifth communication event which represents a unicast communication from Processor 4 to Processor 6; the difference in time between the reference and select pulses is 28 time units, dividing by ω gives 7 and since the processor nearest the fold is Processor 0 requiring zero time unit signal difference, the 28 time unit signal difference addresses Processor 6. The other communications represent broadcasts, for convenience, the source processor transmits to itself during a broadcast.

```
6
5: 161 [ 161 165 169 173 177 181 185 189 193 197 ] 161 46
5: 230 [ 230 234 238 242 246 250 254 258 262 266 ] 230 46
6: 267 [ 267 271 275 279 283 287 291 295 299 303 ] 267 46
4: 290 [ 290 294 298 302 306 310 314 318 322 326 ] 290 46
4: 302 [ 330 ] 302 46
0: 476 [ 476 480 484 488 492 496 500 504 508 512 ] 476 46
```

Fig. 3. Generator Output

2) Unsafe Communication Output: The program produces two types of outputs. The first type writes detailed messages to the standard output that serve as a log of simulation activity. The log details many steps, including the removal of expired communications, conversions of `C` to waveguide time, and results of the various safe communication checks. Refer to Fig. 4 for the log output that results from running

```
====================================
P_5, C^p_0:
   ( 161 [ 161 165 169 173 177 181 185 189 193 197 ] 161 )
P_5, C^w_0:
   ( 411 [ 411 415 419 423 427 431 435 439 443 447 ] 411 )
====================================
P_5, C^p_1:
   ( 230 [ 230 234 238 242 246 250 254 258 262 266 ] 230 )
P_5, C^w_1:
   ( 480 [ 480 484 488 492 496 500 504 508 512 516 ] 480 )
====================================
P_6, C^p_2:
   ( 267 [ 267 271 275 279 283 287 291 295 299 303 ] 267 )
P_6, C^w_2:
   ( 567 [ 567 571 575 579 583 587 591 595 599 603 ] 567 )
====================================
P_4, C^p_3:
   ( 290 [ 290 294 298 302 306 310 314 318 322 326 ] 290 )
P_4, C^w_3:
   ( 490 [ 490 494 498 502 506 510 514 518 522 526 ] 490 )
overlap between selects of C_1 and C_3
C_3 =><= C_1
====================================
P_4, C^p_4:
   ( 302 [ 330 ] 302 )
P_4, C^w_4:
   ( 502 [ 530 ] 502 )
overlap between msgs of C_1 and C_4
C_4 =><= C_1
====================================
P_0, C^p_5:
   ( 476 [ 476 480 484 488 492 496 500 504 508 512 ] 476 )
P_0, C^w_5:
   ( 476 [ 476 480 484 488 492 496 500 504 508 512 ] 476 )
C_5 (select 476) and C_1 (reference 480) incorrectly
                                          coincide at P_1
C_5 =><= C_1

number of wrong coincides: 1
number of reference overlaps: 0
number of select overlaps: 1
number of msg overlaps: 1
```

Fig. 4. Checker Log Output

the checker on the file shown in Fig. 3. In this example, the output consists of six detail sections corresponding to each of the communication sets in the input file. For each detail section, the input set is first printed with times given in processor times followed by the same information converted and printed with times given in waveguide times. The detail section also indicates if one of four errors exist. Lastly, the log file contains summary statistics about the total numbers of errors that exist. The second type of output is produced only when an overlap is found. Each case is written to a separate file that graphically depicts the overlap. Refer to Figs. 5 and 6 for the graphical output that results from running the checker on the file in Fig. 3. The first graphical output corresponds to the first overlap appearing in the log output file. The first communication set, C1, is shown above the second, C3. Observation of the log output file indicates that the third select pulse of C1 at waveguide time 488 will overlap with the first select pulse of C3 at wave guide time 490 since ω is set to four. This situation is clearly shown in Fig. 5. The second graphical output corresponds to an overlap in the message waveguide.

B. Simulation Results

Several tests were conducted in [1] to ensure that the simulation correctly identified known cases of communication

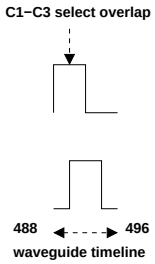

Fig. 5. Checker graphical output

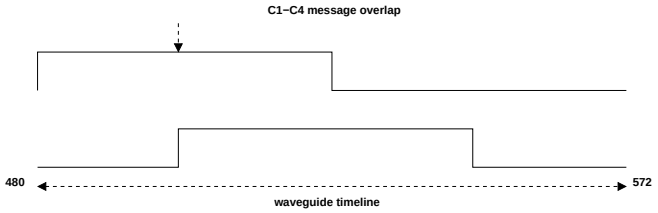

Fig. 6. Checker Fig Output

interference. These tests included one for safe communication, i.e., that two known unicasts do not interfere; and two for determining interference due to known overlapping signals and incorrect coincidence.

The simulation involves thirty randomly generated test cases for each type of operation, unicast, multicast, broadcast. The test cases are constructed such that ten of the thirty contain five communication events, ten more contain 50 events, and the final ten cases contain 500 events. In addition, some special tests for each of the three operations are also included.

1) Multiple Unicasts: A unicast is a point-to-point operation. This is modeled as a single communication event $(r, S = (s_0), m)$ where $r = m$ and $r \leq s_0 \leq r + (N-1)\omega$. A single unicast is safe according to the communication model. Multiple unicasts are modeled by multiple communication events in T.

Table II details the simulated results of potential bus conflicts. For 50 communications, an average of 12.4 or 24.8% conflicts occur whereas for 500, an average of 352.4 or 70.5% conflicts occur. As expected, the greater the number of communication events on the bus, the greater the number of expected conflicts.

Message overlaps account for the largest component at approximately 46%-48% of the total number of conflicts. This is not surprising since message signals use up more waveguide 'space' than either reference or select pulses. Collisions in this case should be lessened when the message size is reduced in relation to τ. To observe the expected reduction, the set of ten cases containing the 50 unicast events is modified to produce a new set of ten test cases where the message size is half of that originally used. Simulation results on this new set show a reduction from an average of 5.9 in the original set to 3.0, or approximately, one half the number of bus conflicts. A linear relationship between the message size and the number of conflicts is suggested by these results. The total average

TABLE II
UNICAST RESULTS

No. of Comms	Run ID	Overlaps			Wrong Coinc.	Total Unsafe
		Ref.	Sel.	Msg.		
5	1	0	0	0	0	0
5	2	0	0	0	0	0
5	3	0	0	0	0	0
5	4	0	0	0	0	0
5	5	0	0	0	0	0
5	6	0	0	0	1	1
5	7	0	0	0	0	0
5	8	1	0	0	0	1
5	9	0	0	0	0	0
5	10	0	0	0	0	0
50	11	1	0	9	3	13
50	12	1	0	5	8	14
50	13	1	3	7	2	13
50	14	0	2	10	2	14
50	15	0	1	7	7	15
50	16	1	2	5	2	10
50	17	1	1	5	6	13
50	18	3	1	4	6	14
50	19	0	0	3	4	7
50	20	0	0	4	7	11
500	21	43	32	164	117	356
500	22	49	31	161	111	352
500	23	41	47	150	122	360
500	24	46	33	158	112	349
500	25	40	29	171	106	346
500	26	40	38	160	117	355
500	27	36	33	165	114	348
500	28	30	47	177	110	364
500	29	46	36	164	102	348
500	30	38	34	149	125	346

unsafe messages drops from an average of 12.4 in the original set to 9.7 in the new set; or about a 22% drop in bus conflicts.

2) Multiple Broadcasts: A broadcast is a one-to-all operation. This is modeled as a single communication event $(r, S = (s_0, ..., s_{N-1}), m)$ where $r = m$ and $s_0 = r, s_1 = r + \omega, s_2 = r + 2\omega, ..., s_{N-1} = r + (N-1)\omega$. A single broadcast is safe according to the communication model. Multiple broadcasts are modeled by multiple communication events.

Table III details the simulated results of potential bus conflicts. For 50 communications, an average of 16.6 or 33% conflicts occur whereas for 500, an average of 417.8 or 83.5% conflicts occur. As with unicasts, the greater the number of communications, the greater the number of bus conflicts.

Table III also indicates that select pulse collisions account for the greatest number of conflicts, on average 230.5, or about 55% of the collisions for 500 communications. There are also a significant number of wrong coincidences. First, broadcasts use significant select waveguide 'space' thereby severely limiting availability. The simulation confirms this observation. Second, each broadcast communication consists of one reference pulse but many select pulses. The simulation confirms the expectation that many wrong coincidences occur during multiple broadcasts. Note that the simulation halts checking upon the first collision detected and that wrong coincidences are checked first. Also note that message overlaps are checked last; this accounts for the very low numbers of message overlaps reported in the table.

3) Multiple Multicasts: A multicast is a one-to-many operation. This is modeled as a single communication event, $(r, S = (s_0, ..., s_i), m)$, where $r = m$, $1 \leq i \leq N$, $r \leq s_j \leq r + (N-1)\omega$ such that s_j occurs at ω multiples away from r, and, $s_j \neq s_k$, where $j \neq k$, $0 \leq j \leq i$, $0 \leq k \leq i$. The latter clauses states that for any two different select pulses, their times are not the same. A multicast is similar to a broadcast differing only in that the select vector is a subset of the broadcast select vector. A single multicast is safe according to the communication model.

4) Mix: This simulation consists of a set of random types of events made up from unicasts, multicasts and broadcasts. Table V details the simulation results. As expected, these results follow closely those presented earlier.

5) Simulation Summary Comments: The simulation considered herein establishes several interesting observations. First, anticipated behavior has been confirmed, that is, increases in bus conflicts are observed with increases in amount of bus traffic. In particular, the collisions due to message waveguide signal conflicts indicate an upper bound on the effective

TABLE III
BROADCAST RESULTS

No. of Comms	Run ID	Overlaps			Wrong Coinc.	Total Unsafe
		Ref.	Sel.	Msg.		
5	1	0	1	0	0	1
5	2	0	0	0	1	1
5	3	0	0	0	0	0
5	4	0	0	0	0	0
5	5	0	0	0	0	0
5	6	1	0	0	0	1
5	7	0	0	0	0	0
5	8	0	0	0	0	0
5	9	0	1	0	1	2
5	10	0	0	0	0	0
50	11	1	7	0	7	15
50	12	1	9	1	12	23
50	13	2	3	0	9	14
50	14	0	10	0	5	15
50	15	1	7	0	8	16
50	16	1	7	0	3	11
50	17	4	8	0	7	19
50	18	3	8	0	9	20
50	19	2	9	0	3	14
50	20	0	13	0	6	19
500	21	26	231	2	159	418
500	22	43	225	2	148	418
500	23	33	222	0	161	416
500	24	16	247	3	155	421
500	25	30	223	1	158	412
500	26	22	239	1	162	424
500	27	22	241	2	155	420
500	28	29	223	0	164	416
500	29	26	212	5	172	415
500	31	21	241	0	156	418

bandwidth of the optical interconnection network. Additional measures of controlling or interleaving communications, or, augmenting the network with additional capacity, may be needed to offset the collision issue. Second, the deviation in the number of collisions for each set is low, perhaps suggesting stability in bus collision traffic. Third, comparisons between the four types of communication events, unicast, broadcast, multicast and mixed, have confirmed anticipated impacts due to the four collisions types.

In comparison of broadcasts with unicasts: (a) the greater bus usage of broadcast communication leads to greater bus collisions, this observation is borne out by the approximately 34% increase for 50 communications and the 18.5% increase for 500 communications over unicast bus conflicts; (b) an increase in the number of wrong coincidences due to the reasons stated earlier.

Multiple multicasts are modeled by multiple communication events. Since this type of operation generally lies in between unicast and broadcast in terms of the number of select pulses generated for a communication, it is expected that the number of collisions lie somewhere in between as well. The simulation results confirm this expectation, in particular, for total bus conflicts, wrong coincidence and select overlaps. Table VI details these results. In particular, the 414.1 average bus conflicts for 500 communication events is less than the 417.8 average corresponding to the broadcasts, but greater than the 352.4 average corresponding to the unicasts. Note that a random number is used in the generator program to generate the sequence of multicast select pulses.

This observation is extended by ranking the effect of the four communication types as given in Table VI. Interestingly, the mix communication type has fewer collisions than either purely multicast or broadcast. However, the mix, multicast and broadcast types are somewhat clustered, suggesting that any sequence of communication events that consists of multiple select pulses will increase the likelihood of bus collision.

TABLE IV
MULTICAST RESULTS

No. of Comms	Run ID	Overlaps			Wrong Coinc.	Total Unsafe
		Ref.	Sel.	Msg.		
5	1	0	0	0	0	0
5	2	0	0	0	0	0
5	3	0	0	0	0	0
5	4	0	0	0	0	0
5	5	0	0	0	0	0
5	6	0	1	0	0	1
5	7	0	0	0	1	1
5	8	1	0	0	0	1
5	9	0	0	0	0	0
5	10	0	0	0	0	0
50	11	0	11	1	5	17
50	12	2	7	1	6	16
50	13	2	8	0	2	12
50	14	2	7	1	6	16
50	15	2	7	0	4	13
50	16	1	9	0	7	17
50	17	2	7	0	8	17
50	18	1	10	1	8	20
50	19	1	8	1	4	14
50	20	1	9	2	3	15
500	21	35	205	21	145	406
500	22	26	210	19	150	405
500	23	35	194	20	153	402
500	24	36	208	16	145	405
500	25	24	209	13	157	403
500	26	35	188	31	149	403
500	27	28	214	17	143	402
500	28	32	188	27	157	404
500	29	32	205	15	152	404
500	30	37	212	24	134	407

TABLE V
MIXED TYPES RESULTS

No. of Comms	Run ID	Overlaps			Wrong Coinc.	Total Unsafe
		Ref.	Sel.	Msg.		
5	1	0	0	0	0	0
5	2	0	0	0	0	0
5	3	0	0	0	0	0
5	4	0	0	0	0	0
5	5	0	0	0	0	0
5	6	0	0	0	0	0
5	7	0	0	0	0	0
5	8	0	0	0	0	0
5	9	0	0	0	0	0
5	10	0	1	0	0	1
50	2	1	8	0	4	13
50	3	0	5	2	5	12
50	4	3	4	4	4	15
50	5	0	7	3	3	13
50	6	0	5	2	6	13
50	7	0	6	1	2	9
50	8	2	5	0	5	12
50	9	3	6	4	5	18
50	10	1	3	2	6	12
50	11	0	6	6	3	15
500	2	38	167	56	139	400
500	3	30	158	64	137	389
500	4	38	156	50	150	394
500	5	23	158	55	155	391
500	6	26	173	51	143	393
500	7	37	159	40	156	392
500	8	34	160	54	138	386
500	9	32	152	49	164	397
500	10	34	156	61	144	395
500	11	30	141	55	166	392

TABLE VI
AVERAGE COMPARISONS

Communication Type	Total Unsafe	
	No. = 10	No. = 500
Unicast	12.4	352.4
mix	13.2	392.9
multicast	15.7	414.1
broadcast	16.6	417.8

IV. Conclusion

The optical bus parallel computing models are based on the use of a waveguide to direct optical pulses between processors. Since a common communication medium connects all of the processors, the interconnect is a bus. Although there are variations in the proposed architectures for these models, the majority make use of the coincident pulse technique for processor addressing. The inherent pipelining of communication due to the unidirectional property of light renders the usual bus collision detect method ineffective for multiple independent communications. It appears that much of the algorithms developed under these optical bus models make use of relatively simple communication patterns in a lock-step sequence with computational steps. In order to enable more general use of the optical medium, bus contention issues need to be solved. The approach taken in this paper is to consider collision avoidance.

In this paper, an analysis of the conditions necessary for bus collision and the subsequent communications model that represents the necessary information to determine potential bus collisions are presented. A simulation based on this model has been implemented. Results of the simulation indicate collision patterns and trends in sequences of general communications. We believe that this publication represents the first optical bus communication simulation results published.

The model and simulation considered in this paper establishes a mechanism that enables future analysis. Existing algorithms could be analyzed for sequences of unicasts, broadcasts and multicasts and compared. However, a more useful study would be to consider new algorithms which use more general independent communications.

A limitation with our approach is the reliance on the global reference clock. To do so nicely allows the checking for bus collisions. However, in a truly MIMD environment, such a global clock poses problems of synchronization together with the issues of slower clock and control circuit speeds compared with the optical bus network. Either implementation feasibility of the global clock architecture or enhancements to our approach to eliminate the global clock must be considered in future work. We are considering two alternatives, first, decouple the waveguide time from the processor time in the conversion expressions, second, to consider the feasibility of newer optical technology, for example, the use of an optical clock.

References

[1] M. Beltran, "A safe communication model for optical buses," Master's thesis, Dept. Computer Science, The University of Texas at El Paso, December 2003.

[2] Y. Pan and M. Hamdi, "Quicksort on a linear array with a reconfigurable pipelined bus system," in *Proc. of the IEEE International Symposium on Parallel Architectures, Algorithms, and Networks*, G.-J. Li, D. Hsu, S. Horiguchi, and B. Maggs, Eds., Beijing, China, June 1996, pp. 313–319.

[3] Z. Guo, R. G. Melhem, R. W. Hall, D. M. Chiarulli, and S. P. Levitan, "Array processors with pipelined optical busses," in *Proc. 3rd Symposium on Frontiers of Massively Parallel Computation (Cat. No.90CH2908-2)*, J. Jaja, Ed., College Park, MD, USA, October 1990, pp. 333–342.

[4] C. Qiao and R. G. Melhem, "Time-division optical communications in multiprocessor arrays," *IEEE Transactions on Computers*, vol. 42, no. 5, pp. 577–590, May 1993.

[5] Y. Pan, "Order statistics on optically interconnected multiprocessor systems," *Optics and Laser Technology*, vol. 26, no. 4, pp. 281–287, August 1994.

[6] C. Qiao, "Efficient matrix operations in a reconfigurable array with spanning optical buses," in *Proceedings. Frontiers '95. The Fifth Symposium on the Frontiers of Massively Parallel Computation*, February 1995, pp. 273–280.

[7] S. Pavel and S. G. Akl, "On the power of arrays with reconfigurable optical buses," *Technical Report No. 95-374, Queens University, Kingston, Ontario, CANADA*, February 1995.

[8] Y. Pan and K. Li, "Linear array with a reconfigurable pipelined bus system — concepts and applications," in *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'96), Vol. III*, H. Arabnia, Ed., Sunnyvale, California, USA, August 1996, pp. 1431–1441.

[9] Y. Li, Y. Pan, and S. Zheng, "A pipelined TDM optical bus with conditional delays," in *Proceedings of the Fourth International Conference on Massively Parallel Processing Using Optical Interconnections*, J. Goodman, S. Hinton, T. Pinkston, and E. Schenfeld, Eds., Montreal, Canada, June 1997, pp. 196–201.

[10] H. ElGindy, "An improved sorting algorithm for linear arrays with optical buses (extended abstract)," *(Manuscript)*, April 1998.

[11] J. L. Trahan, A. G. Bourgeois, and R. Vaidyanathan, "Tighter and broader complexity results for reconfigurable models," *Parallel Processing Letters*, vol. 8, no. 3, pp. 271–282, 1998.