Communication Driven Interconnect Synthesis

Chuck Monahan and Forrest Brewer

1.0 Introduction

Until recently, the emphasis in automated datapath construction was optimization through reduction of resources due to area constraints. Lately, this constraint has relaxed somewhat with the reduction of minimal feature sizes and the expansion of die sizes. This shift has led to a reevaluation of existing algorithms as designers construct higher performance, faster designs. In particular, the relative delays of data-path function units versus that of the interconnection and bussing overhead has decreased enormously and is predicted to do so for some time to come. While the trend in chip and MCM designs has begun to favor global point to point bussing, it is clear that these techniques do use up a great deal of area and therefore increase the communication delay through fringing capacitance and growth of physical interconnect distance. Linear data-path bussing offers a favorable topological density and ease of design, but current tools tend to minimize the interconnect hardware without regard to the effects such minimization has on the performance of the design. These circumstances require more powerful tools to allow flexible design while modeling the performance and area costs accurately. In particular, for linear data-path structures, we wish to accurately characterize the delays of the switching elements, their sizes, the delays of the physical busses distributed RC delays, physical positioning information of the components, and critical path delays for scheduled operations including control unit and interconnection overhead.

Achieving these ideals requires a new approach to datapath synthesis. First off, the system must be modeled in more exacting detail. Communication resources must be accounted for in the design and these resources must comprise a larger family of components reflecting the trade-offs of high performance design. Specifically, components such as bus drivers, receivers, switches and buffers must be modeled as well as conventional multiplexers. Furthermore, each assigned communication must impart positional information to the model since transmission delays are highly related to the distributed RC delay of the bus as well as the switching elements. Towards this aim, a pair of programs have been created: Timing Driven Communication Placement (TDCP) and Timing Driven Communication Interconnect (TDCI) systems [SeB91]. In these systems, the process of datapath synthesis has been broken down into two tasks. The first task, performed by TDCP, takes scheduled register transfer language commands in addition to a list of resources and constructs the placement, function unit and register bindings for the datapath. While TDCP adopts a crude rating of the required interconnect as it judges designs, it does not create the actual interconnect. Instead it relies on a heuristic model of the track density and communication times specific to the implementation technology.

The second stage of the process is the focus of this report and is conducted by TDCI. TDCI assigns the communications proposed by TDCP into a realizable interconnection structure using a very general connection model to meet the design requirements imposed by TDCP. These constraints are described in terms of TDCI's objective functions. The

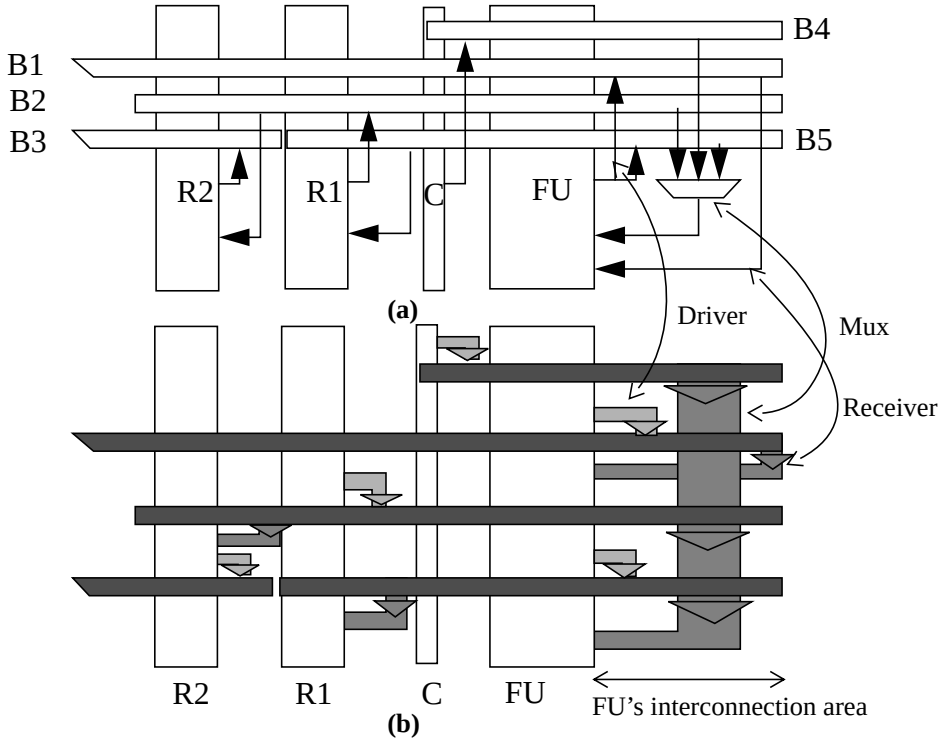
available objective functions include critical path delay, datapath physical length, aggregate wire length, and track density in addition to the standard measures of multiplexer, driver, and receiver counts. The resulting output includes the layout of the cell interconnect and bus assignments for each communication qualifying it as a placed net-list capable for implementation by a conventional silicon assembly tool.

2.0 Model and Technology Issues

The target architecture is designed to make it diversified and yet consistent with real world techniques. This section is intended to describe the types of designs TDCI is capable of producing and what constraints are placed on the solution space.

To make use of simple, efficient floorplanning and routing, a bit slice architecture is adopted. Figure 1 depicts an example of this architecture in both a schematic model and the visual rendition, as drawn by TDCI, in parts (a) and (b) respectively. The bit slice paradigm asserts that the information of an entire design may be summed up by the layout for a single bit that is duplicated vertically to achieve the proper word size.

FIGURE 1. Sample Layout



The *track density* is defined as maximum number of busses crossing any single bit vertical slice of the design. In Figure 1, a four track solution is shown. It is important to also notice that a conservative approach to track counting is adopted in which any two busses ending at a cell require may two tracks. For example, B2 and B3 are not considered to share a track since doing so requires a particular pin arrangement that may not be supported by a given cell set. Although TDCI can accommodate either sort of cell, the examples shown later all assume the conservative case.

Drivers and taps to busses must be modeled carefully if the designs are expected to conform to realizable structures. Common output signals are assumed to require separate drivers for each bus that is connected. Drivers and taps occupy physical space as well

which adds to the length of the busses crossing them. TDCI makes no distinction between a “bus” and a wire. All connections between differing cells occupy physical space and are modeled similarly. Thus a “bus” with one source is still consistently modeled for the electrical loading and area it requires. The number of busses connecting to a input pin is modelled as transmitting through a multiplexer whose size is linearly dependent upon its fanin and whose delay is logarithmically dependent on the fanin. Constants can be either folded into multiplexer inputs using special cells which are constructed by logic synthesis or can be stored in bulk ROM’s which are added to the structure as ordinary cells. In the examples that follow, all constants are explicitly stored into ROM cells to make them visible in the designs.

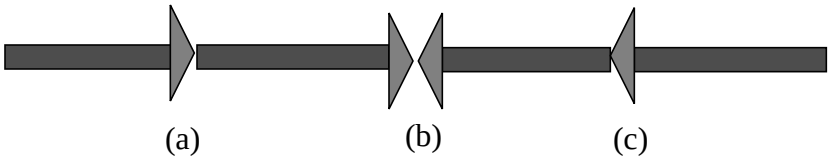
3.0 Interconnect Optimization Techniques

TDCI’s approach to solving the interconnection problem allows for various methods of reducing the complexity. Many of these techniques have been established in previous systems, but were not integrated. TDCI uses the following list of techniques to maximize the performance of the design.

- Bus partitioning and inter-bus communication and buffering
- Commutative operands to modify connection loading
- Redundant operands to optimize communication path selection

Bus partitioning allows the system to select whether or not divide a bus into sub-busses by means of additional switching logic. In actuality, this structure is a series of busses connected with tri-stated drivers or “directional repeaters”. The term directional reflects that repeaters are selected to broadcast in only one or both directions as required. In Figure 2 on page 3, all three type of repeaters are display: (a) up, (b) both, and (c) down direction.

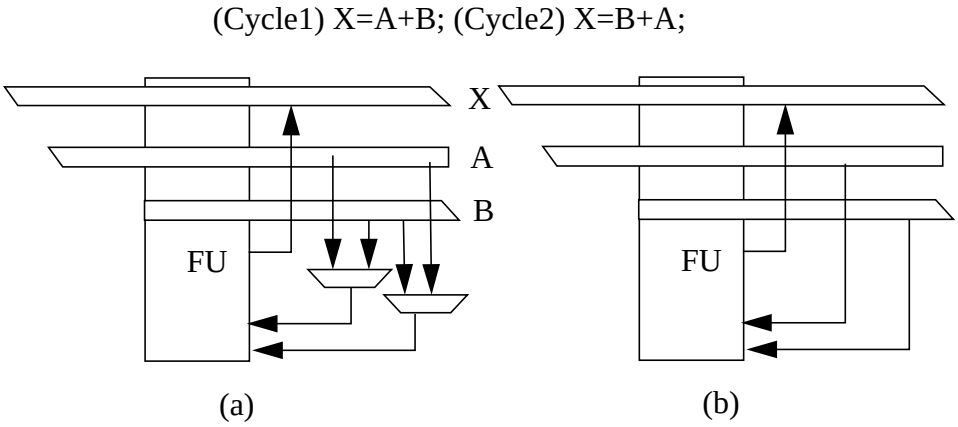
FIGURE 2. Repeater Types



The inclusion of the repeaters allows the system to make more efficient use of it’s tracks. A optimal track solution for a given layout will always be equal to the greatest number of unique communications that simultaneously traverse a cell. Furthermore, it can be shown that this solution is always obtained when unrestricted use of repeaters is permitted. The price for including repeaters is the added area and power consumed by the design. Repeaters can have a very favorable effect on the speed of long busses with significant distributed RC delays. Such delays generally grow quadratically with bus length, while busses with periodic repeaters have linear delays per unit length. Thus for very high performance designs, repeaters my be added to time-critical bus segments to improve overall performance.

By noting which operands are commutative, TDCI can rate the effect of swapping the input pins for subsets of the operands. This ability increases the freedom associated with input multiplexer complexity. An example of the benefit is depicted in Figure 3., where the effective reduction in hardware is derived from swapping the input operands.

FIGURE 3. Commutative swap



Finally, binding operands to multiple registers is a method of reducing interconnect delays during resource binding. The goal is to reduce the interconnect required to fetch operands by increasing their spacial availability. The task for the interconnect tool is to select which of the registers to use as a source to lower the cost or increase the performance of the design. For the TDC system, TDCP report all possible communications in the advent of multiple binding and TDCI must identify and group these sets, selecting a single candidate.

4.0 Interconnection Costs

TDCI computes a variety of measures for a particular design. These measures serve as both objective functions for the TDCI algorithm and as design reviews. The user assigns weights to the parameters, influencing the outcome of the design. The list of cost include the following:

- Limiting clock cycle time
- Minimum track count
- System Length
- Wire length
- Number of muliplexers
- Number of drivers
- Number of receivers
- Number of repeaters

Many times the clock cycle rate is the most important cost, but it is very difficult for most systems to compute. TDCI's use of communications as inputs permits the construction of dataflow graphs for each cycle. The addition of component operation delays as well as technology information allows the computation of the time delay for the critical path of each cycle. By examining each cycle, TDCI identifies the maximal cycle time as the clock cycle of the device

To maximize the effectiveness of this measurement, communications delays are modelled with a high attention to detail. Lumped capacitance models prove overly conservative and misleading in many cases. Likewise, point to point estimates are too liberal. Therefore, TDCI employs a distributed RC model which uses explicit positional and bus modeling information.

Two objective functions monitor the area of the final design. In particular, the system length and the track functions control the x and y dimensions of the design. in conjunction they are very effective in controlling the overall area.

The aggregate wire length function proves to be useful in minimizing designs. Minimizing this term helps maximize the bus utilization of the system. This function must be tempered since utilization can undermine the timing performance of the system. But this term provides a measure of the general interconnect. For more specific measures of the interconnect, the remaining functions measure the occurrence of the specific interconnect devices. These measures can be useful in pursuits of more traditional designs which measure success in terms of the number of components, such as, multiplexers and drivers.

5.0 Results

This section discusses results for two different benchmark examples. For both designs, the supporting parameters for the technology and the library of components must be specified, because of their influence on the results. The values used were based on a 0.8 micron technology and are listed in Table 1.

TABLE 1. 0.80 micron technology w/ 10x Drivers

	10x Driver	**20x Driver**	**Receiver**	**Nx1 Multiplexer**	**Wire**
Width	35 microns	35 microns	10 microns	N*20 microns	n/a
Resistance	3175 ohms	1587 ohms	n/a	n/a	1660 ohms/cm
Capacitance	20 ff	20 ff	7ff	7ff	1760 ff/cm
Time Delay	n/a	n/a	0 ns	0.5 + N*0.3 ns	n/a

5.1 Differential Equation

The differential equation or HAL example was modelled in accordance to published results.[PaK89] This model includes the allocation of a two cycle pipelined multiplier with an imbedded register and a function unit capable of addition, subtraction, and comparison operations. In addition to the registers required, A set of ROM's are provided to store the constants. Finally, the algorithm implemented is the current HLSW standard in which the constant '5' does not appear. A summary of the component library is found inTable 4.

TABLE 2. Differential equation library

Component	**length**	**delay**		
Register	140 microns	1 ns (read)	1 ns (write)	
ROM constants	5 microns	1 ns (read)		
Function unit	170 microns	8 ns (add)	8 ns(sub)	8 ns (less than)
2 stage mult	2000 microns	22 ns (stage 1)	22 ns (stage 2)	

This problem was selected to prove TDCI's ability to compete with existing tools. In this case, TDCI was instructed to minimize the number of drivers and multiplexers as to match Paulin and Knight's solution. [PaK89] In order to compare results, the published solution was manually converted into a datapath layout. As the layout was derived the muxes were converted into multi input busses in order to reduce the number of tracks.

Then, TDCI found the same layout for the case of 10x and 20x driver sizes as depicted in Figure 4 and Figure 5 on page 8.

Table 3 lists the results of these two designs. The first five enters are familiar measures. The multiplexer measure has two values, the first for the number of actual devices and the second relates to the number of two input muxes required for construction. The last three measures are statistical data rating the effectiveness of the design. %Bus Util averages the portion of the bus in use versus the total layout for each cycle. This statistic gives an indication of the amount of bus structure allocated for general communications versus that generated to serve specific, typically critical, communications. Finally, %Crit Path and %Avg Bus Delay measure the percentage of delay caused by the interconnect on the critical path of the system and the average critical path, respectively.

TABLE 3. Differential equation results

	Time	Tracks	Length	Mux #	Driver #	%Bus Util	%Critc Path	%Avg Bus Delay
10x driver	25.41 ns	4	3505 um	0:0	15	54.45%	9.51%	8.45%
20x driver	24.62 ns	4	4030 um	0:0	15	52.73%	6.59%	5.81%

The results show TDCI's capacity for generating and analyzing solutions. TDCI interface with technology files allows the designer to rate changes in performance versus the area implied by changing driver sizes or even technologies.

5.2 Elliptic Filter

For the second example, the elliptic filter, more practical components are employed. The prime example of this is a two stage pipelined multiplier modeled as a pair of components with no internal storage. Each component is capable of computing either the first or second stage of the multiplication requiring. During the second phase. the multiplier is fed both of the original operands as well as an intermediate result stored externally. In addition, a pair of ROM units be allocated to feed both multiplier constants when used simultaneously. While this stipulation increases the validity of the design, it also raises the complexity of the interconnection.

A pair of adders were allocated to achieve a nineteen state schedule. The library component values for this structure are listed inTable 4.

TABLE 4. Elliptic Filter library

Component	length	delay	
Register	140 microns	1 ns (read)	1 ns (write)
ROM constants	140 microns	1 ns (read)	
Function unit	170 microns	8 ns (add)	
2 stage mult	1000 microns	22 ns (stage 1)	22 ns (stage 2)

Initially, two solutions with low cycle times were requested for both 10x and 20x drivers. The results of this analysis appear in Figure 6 and Figure 7 on page 9 for the 10x and 20x driver solutions, respectively. Since the designs varied, the files were manipulated and the opposite driver sizes were substituted. The results in Table 3 reveal a dependence that a solution has upon the input parameters of a given technology. Both solution on page 9

have had their critical paths marked to help interpret this result. In general, these trade-offs are due to the increased distance communications must travel versus the lower driver resistance stemming form the adoption of larger drivers.

TABLE 5. Elliptic filter results

	Time	Tracks	Length	Mux #	Driver #	%Bus Util	%Critc Path	%Avg Bus Delay
10x sol	28.07 ns	11	6845 um	10:23	44	28.72%	14.51%	20.26%
w/ 20x	27.18 ns	11	8490 um	10:23	44	28.92%	11.71%	15.69%
20x sol	26.78 ns	8	9380 um	18:35	55	33.12%	14.12%	14.79%
w/ 10x	28.17 ns	8	7455 um	18:35	55	32.95%	18.35%	19.22%

As a final consideration, note that the percentage of delay due to interconnection is less for the critical path than for the system clock. Both this fact and the measurements of bus utilization are in stark contrasts with the results from the Differential Equation. These figures display TDCI's ability to actually reduce the delay in the second example. This results in the allocation of more specialized busses in order to bring the percentage of delay for the critical path below the average.

6.0 Conclusion

This paper highlights a new system, TDCI, for generating datapath interconnect. TDCI anticipates and accommodates the future needs for accurate timing models for large scale datapath designs. TDCI's utilization of communication and positional information, enables multiple models of the same design. Designers may uniquely specify their interests through a variety of objective function ranging from area to timing costs. As a member of the TDCP/TDCI system, it implements a stage of an alternative approach to interconnect synthesis.

7.0 References

[PaK89] P Paulin, J Knight, "Force-Directed Scheduling for the Behavioral Synthesis of ASIC's," IEEE Transactions on Computer-Aided Design, Vol. 8. No. 6., June 1989

[SeB91] A Seawright and F Brewer, "High Performance Data-Path Synthesis via Communication Metrics," Proc. GLS-VLSI '92, pp 60-7, Feb 1992

FIGURE 4. Differential Equation with 10x drivers

FIGURE 5. Differential Equation with 20x drivers

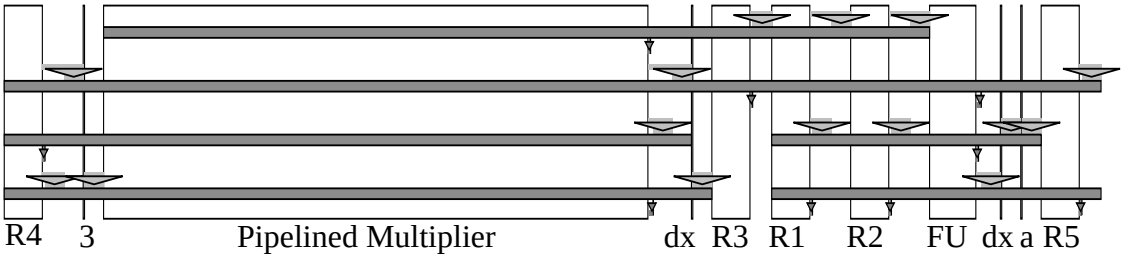


FIGURE 6. 10x design

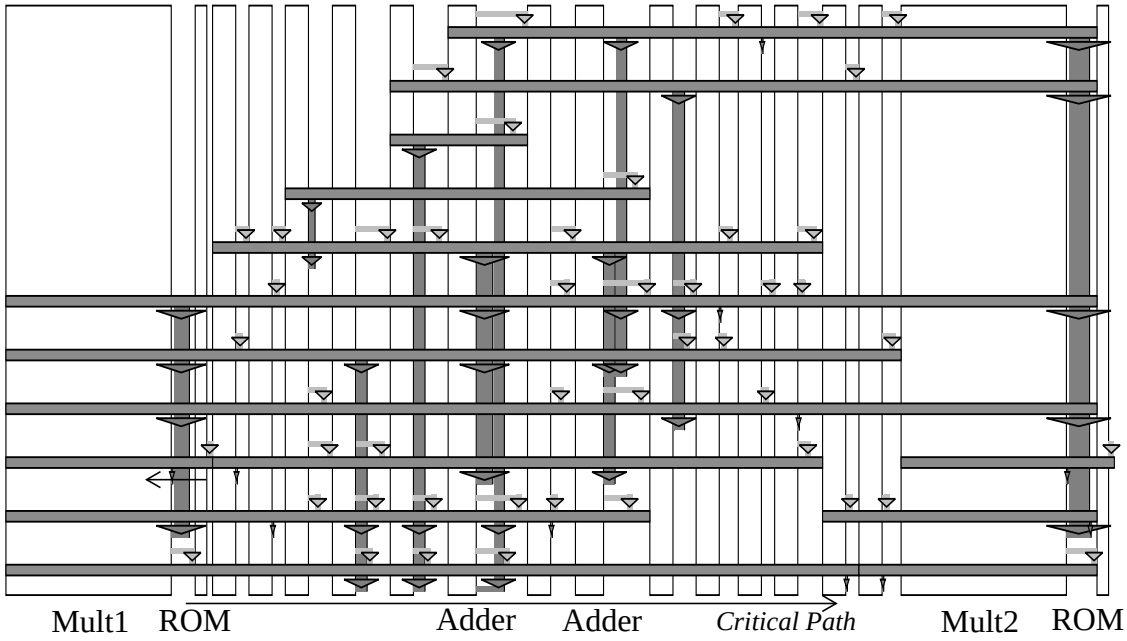


FIGURE 7. 20x design