

Long-Range GasP with Charge Relaxation

Swetha Mettala Gilla, Marly Roncken, and Ivan Sutherland

mettalag@pdx.edu, mroncken@cecs.pdx.edu, ivans@cecs.pdx.edu

Asynchronous Research Center

Maseeh College of Engineering and Computer Science

Portland State University, Portland, Oregon, USA

Abstract

GasP circuit modules communicate handshake signals in two directions over a single state wire. The 2008 Infinity test chip demonstrated GasP in 90 nm CMOS operating at four giga data items per second, but revealed that state wires about 5000 lambda long retard operation by about 10%.

Simulations reported in this paper show that GasP modules will tolerate surprisingly long state wires, albeit at reduced throughput. The modules appear to operate correctly with state wires whose delay exceeds the drive time. With such long wires, the receiving module waits until passive distribution of charge brings the wire within range of the receiver's switching threshold. Having put enough charge into the wire, or vice-versa removed enough charge from it, the sending module may proceed with its next task. This result applies equally to other single-track signaling methods.

This behavior calls for a new kind of relative timing constraint to address when the wire charging or discharging process may cease rather than when the signal reaches the far end of the wire.

Keywords: GasP, Infinity chip, long wires, single-track, on-chip communication, self-timed, asynchronous circuits.

1 Introduction

Since publication of the first GasP family of circuits in 2001 [15] we have built and tested a variety of GasP chips. One chip experiment, called Infinity, consists of two rings of 100 GasP modules each that share a common section of 50 modules. The GasP modules in the Infinity test chip, which includes the Infinity experiment, form the basis of a wide variety of network-on-chip topologies. A sketch of the layout of the Infinity experiment appears in Figure 1(a).

Figure 1(b) offers a canopy diagram for separate operation of the two rings in Infinity. For this experiment data circulate in a single ring, avoiding contention for use of the

50-stage shared section. The work reported in this paper began with the observation of the flat tops in this (measured) canopy diagram. The maximum throughput indicated by the two flat tops is about 10% below the peak throughput in Figure 1(c) as measured for other GasP rings on the chip.¹

Additional experiments traced the 10% loss in throughput observed in Infinity to the longer state wires that connect the common ring section in the center column of Figure 1(a) to the individual ring sections in the side columns. The state wires from column to column are about 5000 lambda long, whereas the state wires within each column are only about 500 lambda long. This phenomenon prompted us to study the effect of wire delays in GasP.

Our first study, published by Joshi et al. in [5], uses a lumped capacitance model to examine the impact of a long state wire. It presents a logical effort model to compute the gate and path delays in GasP operations and to analyze the relative timing constraints on which the correct operation of a GasP module depends [16, 12]. We observed that if the main effect of the long wire is to retard its driver, GasP modules should operate correctly over a large range of distances. Our analysis predicts correct operation provided the difference in the distances to predecessor and successor modules is limited. It predicts failure if these distances differ by too much. Moreover, the extra delay computed for the lumped capacitance model of a 5000 lambda long wire explains the 10% loss in throughput observed in Infinity.

However, the predicted maximum difference in module distances in the study by Joshi et al. is large enough to cast doubt on the adequacy of the lumped capacitance wire model, in particular for wires longer than 5000 lambda. Therefore, we started a second study that uses better models

¹Silicon experiments and simulations in this paper all use a 90 nm CMOS process by TSMC in which lambda (λ) is 50 nm and tau (τ) is about 8 psec. We prefer to use the process-normalized metrics lambda and tau rather than absolute metrics so as to make our results more readily applicable to other manufacturing processes.

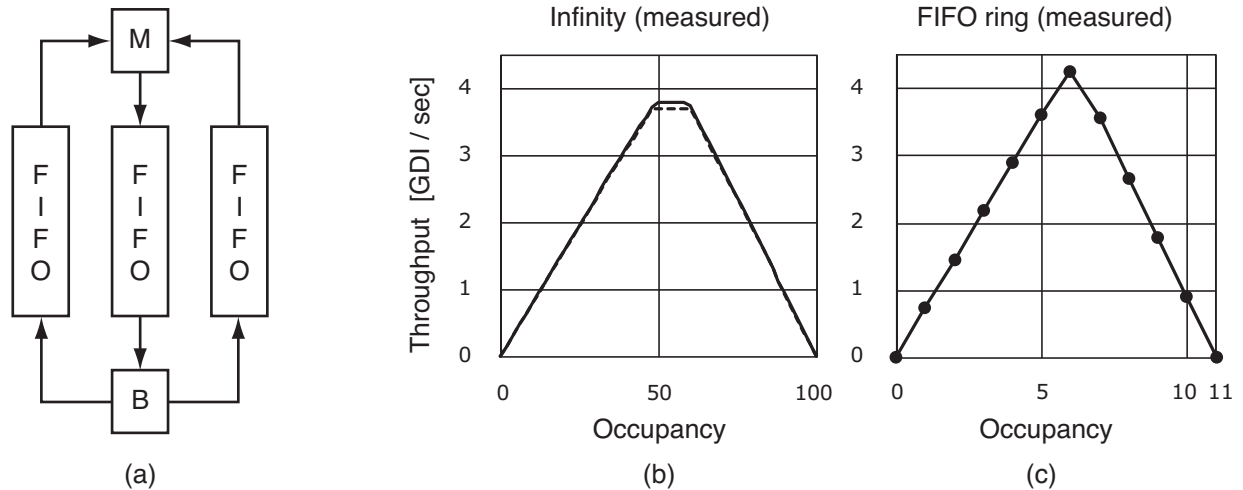


Figure 1 (a) Layout sketch of Infinity, and canopy diagrams for (b) Infinity and (c) an 11-stage FIFO ring on the same test chip. **About (a)**

The Infinity experiment has two rings of 100 GasP modules each, with a common section of 50 modules. The experiment is approximately 17500 lambda tall by 15000 lambda wide, an area of about 0.6 mm². Infinity got its name because its layout resembles the infinity symbol ∞ . The chip was named after the Infinity experiment, which is the largest experiment on it. Infinity is laid out as three columns about 17500 lambda tall of 50 modules each, as shown. Most modules are about 4608 lambda wide and 288 lambda tall, although some are slightly taller. Each module carries 37 data bits plus 15 address bits. An addressable branch module B at the bottom end of the center column uses an address bit to steer arriving data to the left or right. A demand merge module M at the top of the center column accepts data from the side columns on a first-come-first-served basis. Additional test circuitry, omitted here, enables us to load and unload the rings with data and to count the number of data elements passing through each ring. A typical experiment involves loading the rings with a chosen number of data elements, running the circuit for a known time, and then reading the counts and unloading the data for inspection. Every experiment in the chip has observed flawless retention of data counts, sequence, and content.

About (b)–(c)

The canopy diagrams in (b) and (c) show the throughput of GasP rings in relation to their occupancy i.e. the number of data elements in the ring. The throughput is the ratio of the count of data elements that passed through the ring to the run time of the experiment. We have observed maximum throughputs over four giga data items per second (4 GDI/sec). The canopy diagram for the FIFO ring in (c) shows a peak throughput of 4.2 GDI/sec. The solid- and broken-line canopy diagrams for the individual ring operations in Infinity in (b) have a flat top at about 3.8 GDI/sec. A canopy diagram with a flat top indicates the presence of a slower than normal stage. The culprits for the 10% loss of throughput are the relatively long state wires between the side columns and the center column. The column-to-column state wires are about 5000 lambda long, which is ten times longer than the 500 lambda long state wires within each column. All state wires are slightly longer than the center-to-center distance between modules in order to reach multiple connections inside each module. The measurements that identify the long wires as the culprits can be found in [5], together with a theoretical foundation based on logical effort and a lumped capacitance wire model.

to describe the properties of longer wires. Our second study uses a distributed RC model [11, 17], capable of distinguishing voltage levels at the near and far ends of the state wire. For GasP, this matters, because the two ends play different roles in the handshake signaling over the state wire.

We were particularly eager to understand not only the impact of different near and far end delays in the state wires of GasP, but also the implications for the relative timing constraints that we must impose on the GasP circuit modules to ensure their correct operation.

This paper reports the results of our second study. Section 2 introduces the 6-4 GasP family used in Infinity and in our studies, and explains its communication mechanism. Given

that neither Infinity nor the 6-4 GasP control circuits have been published before, Section 2 acts more as a tutorial on GasP than as a description of previously published work.² Section 3 discusses the key relative timing constraints for long wire communication in GasP. Section 4 describes our simulation setup to analyze the potential and limitations for long-range GasP, and presents the results. Section 5 summarizes the results of both studies. We use a two-dimensional graph, the Distance Constraint Graph [6], to help visualize how the results differ. Conclusion and work in progress follow in Section 6.

²Our previous publication [5] gives a partial description of 6-4 GasP, focusing only on key facets of its data transfer mechanism and ignoring keepers and other GasP design aspects presented here.

2 GasP Modules

Infinity uses 6-4 GasP [14]. The 6-4 GasP family gets its name from the six logic gates on the path going forward from one module to the next, path ABCDEF in Figure 2, and from the four logic gates on the path going backwards, path ABCX. The longer delay is in the forward direction because that is the direction in which the data elements are transferred. Copying data requires action on the part of the data latches, whereas moving an empty space or “bubble” backwards to declare the latches empty needs no action.

Figure 2 shows a 2-stage FIFO, with one 6-4 GasP module per stage. The two stages, M_1 and M_2 , are connected by a bidirectional state wire L2 via ports $SUCC_1$ and $PRED_2$. When state wire L2 is high it indicates that M_1 has new data for M_2 . We will refer to this as L2 being full. When state wire L2 is low it indicates that M_2 has latched the proffered data and it is safe for M_1 to change its data. We will refer to this as L2 being empty.³

When $PRED_2$ is high and $SUCC_2$ low, meaning M_1 has new data for M_2 and M_2 has space for it, M_2 raises its signal $FIRE_2$. A high $FIRE_2$ signal starts three parallel actions: (1) the data latches of stage M_2 are enabled to copy the proffered data, and (2) $SUCC_2$ is raised via gates D and E to indicate to M_2 ’s successor that new data are available, and (3) $PRED_2$ is lowered via gate X to declare L2 empty.

Note that for $FIRE_2$ to rise, gate A must synchronize a high $PRED_2$ signal with a low $SUCC_2$ signal. But to get $FIRE_2$ to fall, either a low $PRED_2$ signal or a high $SUCC_2$ signal suffices. Hence, actions (2) and (3) each have the additional side effect of resetting $FIRE_2$.

So, setting $FIRE_2$ high automatically leads to resetting $FIRE_2$ low, and there is a choice of two self-resetting loops that automate this: the forward loop DEABC, and the backward loop XFABC. Note that each loop has five gates.

The gate count matters. GasP modules are custom-designed using logical effort [16]. The transistors in each logic gate are sized such that all logic gates have approximately the same delay. Gate sizing takes into account both the transistor sizes of the driven gates and the loads of the connecting wires. As a result, we can count delay in terms of logic gate delays and we can compare path delays simply by counting and comparing the number of gates on each path.

This works as long as we can adequately predict the wire lengths. Gates A, B, D and F in Figure 2 each drive a single local gate, and so we may assume that the connecting wire lengths are known in advance.

³This paper addresses primarily the control portion of 6-4 GasP that generates the handshake operations between GasP modules. Other than giving an intuition of the data bundling relation between the handshake signaling on state wire L2 and the data being transferred, we omit any further discussion of data validity.

The situation for gate C is more complicated: C drives two local gates, D and X. In addition, C drives a number of latches and the not so local wire L1 between gates D and X and the latches. In Infinity, the data latches are placed near the 6-4 GasP module that controls them. GasP module and latches together form a macro module with a fixed wire length for L1. This solves the gate sizing problem for C.

The exceptions are gates E and X that drive the L2 state wire. The length of L2 depends on the distance between the modules, and this is unknown until the system layout has been finalized. If E and X are sized using the wire lengths for A, B, D and F, as was done in Infinity, the gate delays for E and X will vary with different module distances and the throughput will vary accordingly, as demonstrated by the canopy diagrams in Figure 1(b)–(c).

So, without further investigation into the effects of long state wires, all we can say is that, for short state wires, the maximum throughput of 6-4 GasP is the throughput that corresponds to a cycle time of 10 gate delays: 6 to forward the data and 4 to move the empty space backward. The investigation with long state wires follows in Section 4, and is easier to understand after reading the design details on state wires and keepers coming up next.

2.1 State Wires

GasP state wires use a form of forward and back handshake protocol, called “single-track” signaling [1, 15, 9, 2]. In a single-track protocol, a single wire carries both the forward and the backward handshake signals. This is attractive, not only because a single wire occupies less space, but also because the handshake consumes minimum energy per cycle. To handshake, a transition must pass in each direction. The single wire does exactly that, with automatic return to the initial state after each pair of handshake signals.

The GasP modules in this paper use single-track handshake signaling to communicate control information only. They use single-rail signaling to communicate data, as do the handshake circuits discussed by van Berkel and Bink in [1]. The single-track designs discussed by Nyström et al. in [9] combine both the data and the control into a single-track communication protocol, as do Ferretti and Beerel in [2].

There are various ways to implement single-track signaling; van Berkel and Bink differentiate between “dynamic” and “overlapping” protocols [1]. Each participant in a single-track signaling protocol must cease to drive the state wire soon enough to make room for the action of the other participant. Were one participant to drive the wire for too long, then both might end up driving it concurrently in opposite directions, consuming unnecessary energy and producing an indeterminate logic signal. How is this to be avoided?

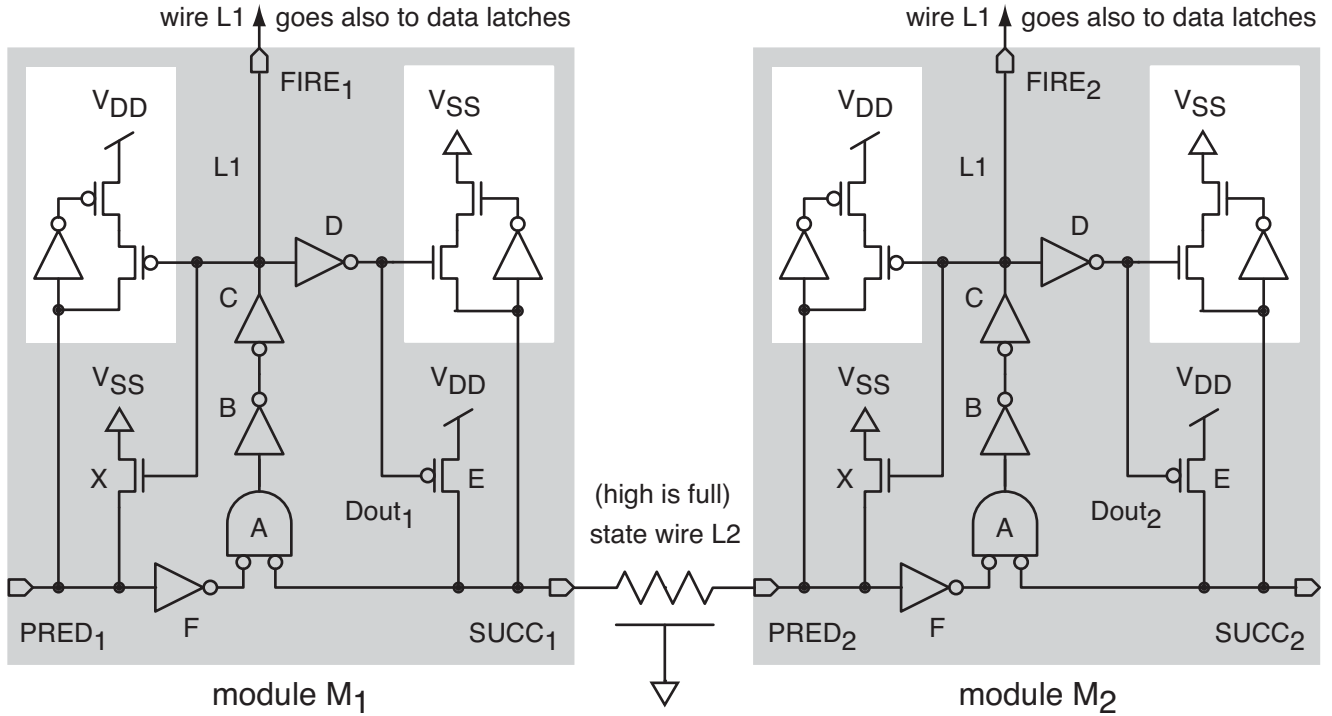


Figure 2 A two-stage FIFO with 6-4 GasP modules M_1 and M_2 connected by a state wire L2. The picture omits the additional state wires to the left and right of the modules shown. It also omits the data wires and latches. The transistors in each logic gate are sized such that all gates have approximately the same delay. The designation “6-4” refers to the forward-backward latencies. The forward latency from gate A in M_1 to gate A in M_2 is 6 gate delays, and is covered by the path ABCDEF. The backward latency from gate A in M_2 to gate A in M_1 is 4 gate delays, and is covered by the path ABCX. This gives a cycle time of 10 gate delays. In addition to this global cycle time, each 6-4 GasP module has two local self-resetting cycles of 5 gate delays: the forward loop goes through ABCDE and the backward loop through ABCXF. The AND function A acts when (a) the predecessor state wire is high, indicating a full predecessor state with data, and (b) the successor state wire is low, indicating an empty successor state with space. The action produces a high pulse on the module’s FIRE signal. The FIRE pulse renders the data latches transparent for a sufficiently long duration to capture the data. It also renders the predecessor state wire empty via transistor X, and it renders the successor state wire full via Dout and transistor E. By the latter two actions the FIRE signal shuts itself off, cutting its pulse time down to 5 gate delays. Half keepers, shown in the white insets, use small transistors to retain the voltage on the state wire when drivers X and E are off. The distributed RC symbol in the L2 state wire refers to the wire model that we use in this paper [11].

In the overlapping protocol, implemented by van Berkel and Bink, this is avoided because each participant drives the wire “long enough” for it to pass some threshold voltage that will alert the other participant who then takes over the driving role. The overlapping protocol fits in the quasi delay-insensitive design style deployed by Philips and Handshake Solutions [10].

GasP uses a dynamic implementation, which is simpler but relies on the relative timing of logic gates to avoid drive conflicts at the state wire. In GasP, the sender “briefly” drives the wire to one logic level, signaling the presence of data on adjacent data wires. The receiver, noticing the change in the wire’s state, copies the corresponding data and then briefly drives the wire to the other logic level to indicate that it has absorbed the data values.

The single-track advantages in GasP are offset by a timing requirement inherent in the word “briefly”. Because sender and receiver drive the shared state wire in opposite directions, each must take care to cease its drive promptly so that the other has free use of the wire. For a 6-4 GasP module, “briefly” means “about 5 gate delays” or half the minimum cycle time. This is where the local self-resetting loops in Figure 2 come into play, as will become clear by following one single-track handshake over state wire L2:

- Sending module M_1 drives L2 high via transistor E. After M_1 starts driving, it also promptly ceases driving via its 5 gate-delay forward self-resetting loop ABCDE. On the receiving end, it takes at least 5 gate delays for M_2 to sense a high voltage level on L2 and invert it through FABCX to drive L2 low.

- After M_2 starts driving L2 low via transistor X, it also promptly ceases driving via its 5 gate-delay backward self-resetting loop FABCX. On the reverse end, it takes at least 5 gate delays for M_1 to sense a low voltage level on L2 and invert it via ABCDE to drive L2 high.
- Except for their separation in space, the P and N type driving transistors E and X act as the two halves of an inverter. However, the crossover current that exists when one transistor's drive turns on and the other's turns off is at least as low as that of an inverter with a shared drive. The crossover current diminishes with longer state wires, as we will explain from the waveforms in Figure 5, later in this paper.

The state wires in GasP carry state: they record the full or empty state of the communication link. To retain the state for arbitrary periods of time, e.g. when there is a shortage of data or of bubbles, there are so-called keepers on the state wire. These are discussed in Section 2.2.

2.2 Keepers

GasP modules include small keepers to retain the charge on state wires in the face of noise or leakage. Each module in Figure 2 includes a half-keeper on each state wire that it can drive. Each half-keeper will keep its state wire either only high or only low. It takes two half-keepers working together to retain the state on the wire. The two half-keepers involved are always on opposite ends of the state wire. For instance, the half-keeper that is responsible for keeping L2 low resides in module M_1 which drives L2 high. And vice versa, the half-keeper that is responsible for keeping L2 high resides in module M_2 which drives L2 low.

The choice of "drive one level, keep the other" makes it easy to shut off the half-keeper right at the start of the drive. Module M_1 uses internal control signal $Dout_1$ to shut off its half-keeper for L2 while driving the state wire in the opposite direction, namely high. At the other end of the state wire, module M_2 uses control signal $FIRE_2$ to shut off its local half-keeper while driving L2 in the opposite direction, namely low. Shutting off the half-keeper right at the start of the drive avoids the energy loss that would occur were state wire L2 driven high while its half-keeper attempts to hold it low, or vice versa.

It is a happy result of the "drive high, keep low" and "drive low, keep high" arrangements in M_1 and M_2 that the half-keepers themselves help drive state wire L2 in a useful way. When module M_1 starts to drive L2 high, it immediately shuts off the half-keeper at its end of the state wire. The half-keeper in M_2 at the other end is also shut off, because that end of L2 is still low.

Only when the far end of L2 reaches the switching voltage of the receiving gate does the half-keeper at the far end turn on, and in doing so assist in driving and ultimately keeping state wire L2 high. This effect, although weak, is observable in the waveforms of Figure 5 shown later in this paper.

3 Relative Timing Constraints

The 6-4 GasP circuits in the Infinity test chip were validated using the SPICE electrical simulation tool. We simulated the behavior of each and every transistor, wire capacitance, and logic function. This was a fairly laborious and compute-intensive process. The results of these simulations showed proper operation over the range of state wire lengths actually needed in the test chip. Now that we know that the test chip works, we want to use the GasP modules in more complex designs. In particular, we want to use standard commercial static timing analysis tools and automatic placement and routing software to inspect timing margins and to drive the layout. This requires a deeper understanding of the conditions that make GasP work.

Our new approach to timing validation in GasP is based on static timing analysis of relative timing constraints [4, 13]. Intuitively, relative timing constraints describe the order in which two signals must arrive at a point of convergence.

The key relative timing constraints for short to medium to long wire communication in GasP come from the presence of the two local self-resetting loops. These loops turn off the communication drive signal to the state wire and we must avoid letting them turn the drive off prematurely.

Take for instance module M_1 in Figure 2. When gate C rises to start the $FIRE_1$ pulse, both loops DEABC and XFABC act to end the $FIRE_1$ pulse and this results in ending the drive on both M_1 's state wires. To guarantee that each state wire is driven long enough to complete its communication, we compare the communication delay to the self-resetting loop delays and require the former to beat the latter. Delays are counted from the start of the module's FIRE pulse. This produces four relative timing constraints per state wire.

For example, state wire L2 in Figure 2 produces the following four relative timing constraints:

- $RT(L2)_{ResetForward}^{TransferForward}$: The forward transfer delay to drive both ends of L2 high through DE in module M_1 is shorter than the forward self-resetting loop delay through DEABCDE in M_1 to cease the drive.
- $RT(L2)_{ResetBackward}^{TransferForward}$: The forward transfer delay to drive both ends of L2 high through DE in module M_1 is shorter than the backward self-resetting loop delay through XFABCDE in M_1 to cease the drive.

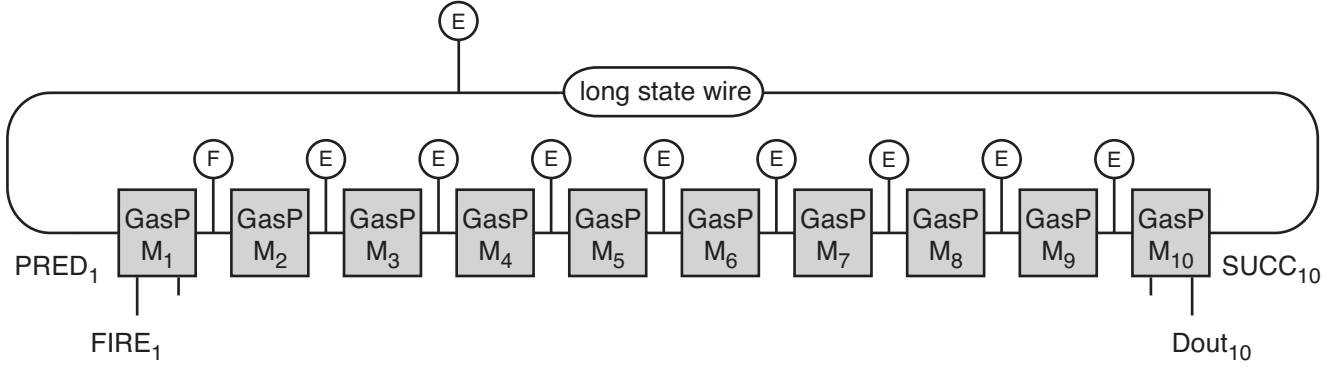


Figure 3 Our long-range GasP simulation set up includes a ring of ten 6-4 GasP modules, M_1 to M_{10} , connected by nine short state wires and one long state wire. The circles with designation E or F indicate the initial states of the state wires: E for empty, i.e. low, and F for full, i.e. high. The ring in this picture starts with one full state wire. The picture explicitly indicates the GasP signals $PRED_1$ and $FIRE_1$ of module M_1 and $SUCC_{10}$ and $Dout_{10}$ of module M_{10} . These are the signals that play a direct role in the handshake communication over the long wire. The picture omits the corresponding signals for the other 6-4 GasP modules.

- $RT(L2)_{ResetForward}^{TransferBackward}$: The backward transfer delay to drive both ends of L2 low through X in module M_2 is shorter than the forward self-resetting loop delay through DEABCX in M_2 to cease the drive.
- $RT(L2)_{ResetBackward}^{TransferBackward}$: The backward transfer delay to drive both ends of L2 low through X in module M_2 is shorter than the backward self-resetting loop delay through XFABCX in M_2 to cease the drive.

The M.Sc. thesis by Prasad Joshi [4] shows how to validate relative timing constraints for a given GasP design using PrimeTime, the Synopsys static timing analysis tool.

4 Long-Range GasP Simulations

To develop our understanding of long state wires in GasP we undertook a series of simulations.

First, we simulated the behavior of isolated long wires using a distributed RC model. We used four R-C-R sections for every 100 lambda of wire, matching the corresponding 90 nm TSMC wire resistance and capacitance numbers for state wires. We simulated the wire delays for wire lengths ranging from 100 to 60000 lambda. We observed that the delay in the wire is, as expected, roughly quadratic with its length. However, for longer wires, the far end of the wire exhibits very slow rise times which renders the precise meaning of “wire delay” suspect. We observed reasonable delays for wires up to 30000 lambda, or 1.5 mm, long.

Armed with these observations, we set up simulations of 6-4 GasP modules separated by long state wires. Each simulation involves a ring of ten 6-4 GasP modules with nine short and exactly one long state wire. Different initializations of the state wires permit us to generate a canopy diagram to exhibit the impact of the long wire on throughput. The two extreme initializations for data-limited rings, with exactly

one full state wire, and for bubble-limited rings, with exactly one empty state wire, will give us a larger time window to observe the dynamic aspects of the long wire and its keepers. Figure 3 shows the simulation set up for the data-limited case with exactly one full state wire.

We simulated each initial configuration with the long state wire set at a length of 1000, 4000, 10000, 20000, 24000, and 30000 lambda. To get reasonable simulation times, we used three R-C-R sections for every 1000 lambda of wire. This still produces a sufficiently fine-grained wire model. All simulated configurations with long state wire lengths up to and including 24000 lambda run correctly.

For the 30000 lambda long state wire we observed failures in the two extreme initialization scenarios with (a) exactly one full state wire and (b) exactly one empty state wire:

- When we start with one full state wire, initialized as in Figure 3, we lose the full state during the first handshake over the 30000 lambda long state wire, because module M_{10} shuts off its high drive before the wire has accumulated enough charge to rise from ground level to the switching voltage level of M_1 . The ring, now completely empty with nothing to work on, deadlocks.
- When we start with one empty state wire, or bubble, we see not a loss of the one and only bubble but the creation of an extra bubble. This is due to the fact that we initialized the ring with the empty state at the long state wire. During the first handshake over the 30000 lambda long state wire, module M_{10} behaves exactly like it did in the previous scenario: it shuts off its high drive before the wire has accumulated enough charge to rise from ground level to the switching level of M_1 . And so, we end up with two bubbles instead of one. With two bubbles, the round trip delay becomes just short enough to keep the ring going.

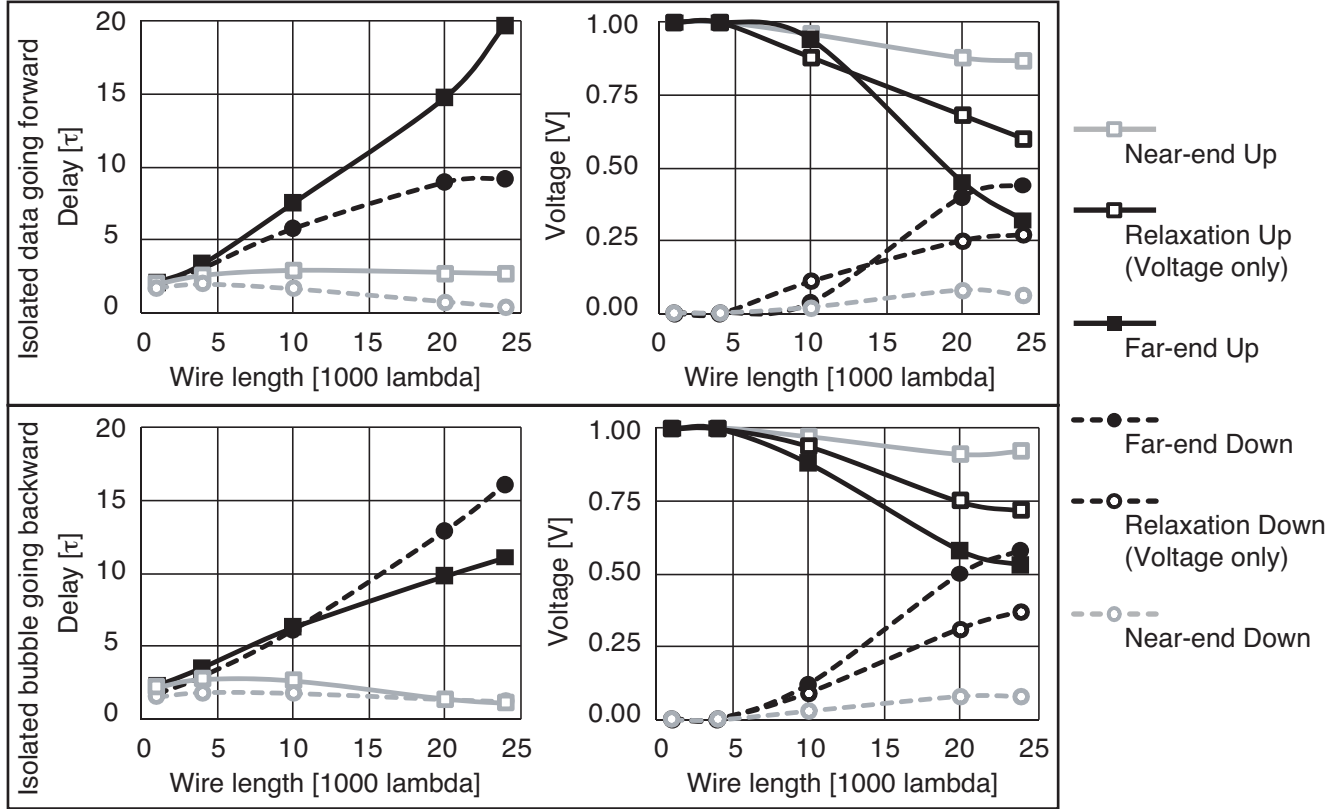


Figure 4 Simulated delays for rising and falling transitions and their voltage levels at the near and far end of the long state wire. The top window shows the results for the simulation set up in Figure 3 with one full state wire. We measured the delays and voltages during a full-to-empty single-track handshake consisting of a rising transition followed by a falling transition in the reverse direction. The bottom window contains similar measurements for the opposite scenario with one empty state wire. The bottom measurements track an empty-to-full single-track handshake, consisting of a falling transition followed by a rising transition in the reverse direction. Notice how the delays and voltage levels at the near and far ends of the wire start to deviate for state wires longer than 5000 λ . We measured the voltage levels at the end of the drive pulse, and again after charge relaxation when both ends have the same voltage. Notice how the 20000 and 24000 long state wires depend on charge relaxation to drive the far end above (top) or below (bottom) the 50% voltage level and thus complete the first part of the handshake. Longer state wires have an easier task in completing the second part of the handshake, because there is less charge to take away (top) or add (bottom).

The rest of Section 4 focuses on the correctly functioning designs with long state wires up to 24000 λ .

4.1 Simulated Wire Delays and Voltage Levels

We measured delays for rising and falling transitions in the long state wire at both the near end and the far end of the wire. The near and far end delays are measured from the time that the drive signal at Dout_{10} or FIRE_1 reaches 50% of the supply voltage level to the times that the near and far ends of the wire reach 50% of the supply voltage level.

In dealing with GasP circuits, we have become accustomed to measuring time in gate delays. With that in mind, we calibrated all delays, including the wire delays, in terms of τ , the normalized inverter delay with equal rise and fall times.

Figure 4 summarizes the measurement results that we obtained for the extreme simulation scenarios with one full state wire (top) and one empty state wire (bottom). We believe that these are the simulation scenarios that stress the GasP ring operations the most. The left-hand graphs show the normalized delays for transitions at the near and far ends of the long state wire. The right-hand graphs show the voltage levels at both ends, measured at the end of the drive pulse and again after charge relaxation.

For state wires up to 5000 λ , the delays for rising and falling transitions take 2–4 τ , which is about one FO3 or FO4 gate delay, and about one gate delay in 6–4 GasP. The voltage levels change more or less simultaneously at both wire ends, with rail-to-rail voltage swings. The delay is almost entirely in driving the wire. Beyond 5000 λ , delays and voltage levels start to deviate at the two wire ends.

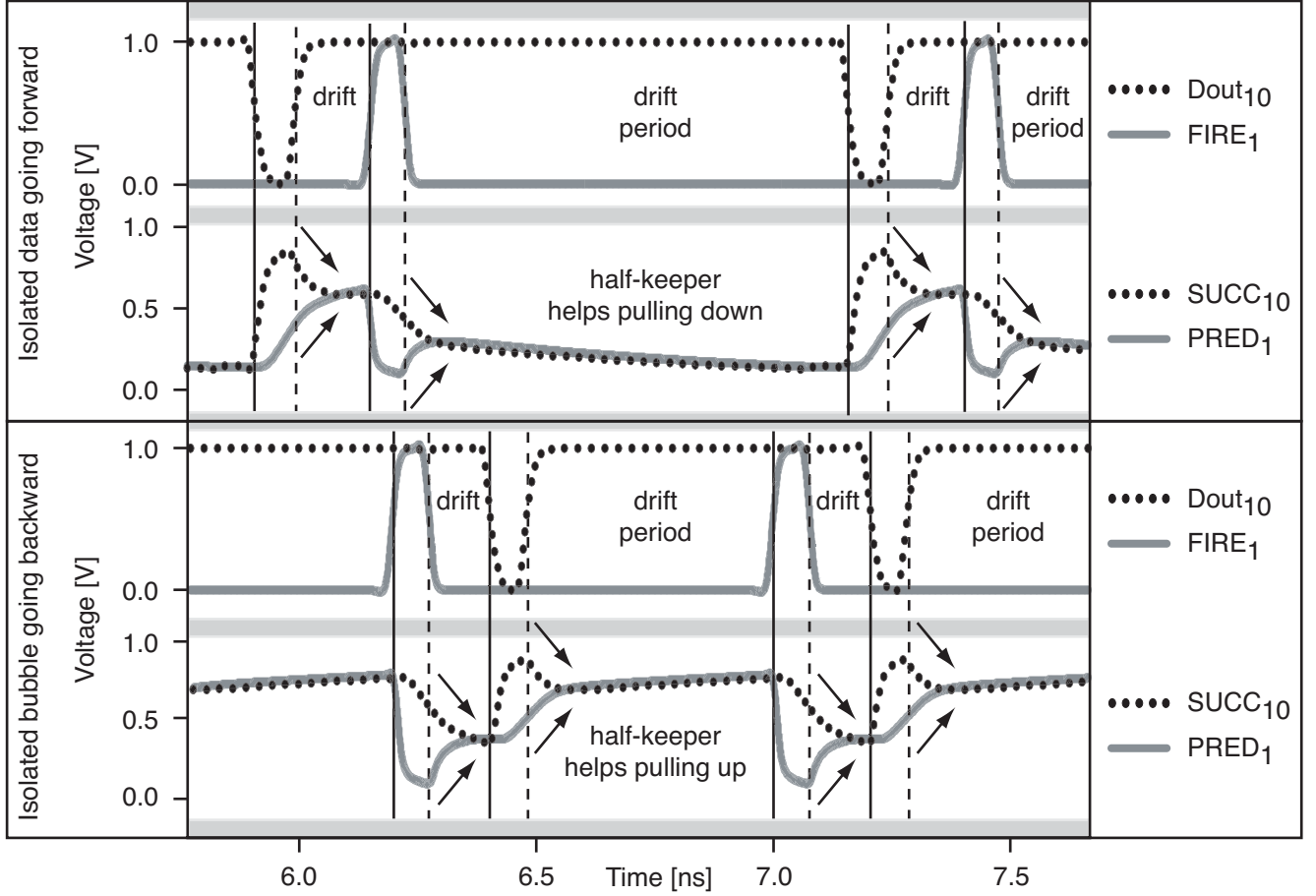


Figure 5 Voltage waveforms for the 10-stage GasP ring of Figure 3, with a long state wire of length 24000 lambda, and initialized with one full state wire (top) and with one empty state wire (bottom). The upper frames of each pair show the voltages to the P and N type drive transistors of the long state wire, viz. $Dout_{10}$ and $Fire_1$. The lower frames of each pair show the voltages at the two ends of the long state wire, viz. $SUCC_{10}$ and $PRED_1$. For very long state wires, the drive pulse shuts off before the far end of the wire reaches 50% of the supply voltage level. Thereafter, the near and far ends drift closer to the same voltage as the charge distributes along the length of the wire and relaxes into equilibrium. The half-keepers at the far-end turn on when the far end reaches its switching voltage, and in doing so assist in keeping the state wire at or closer to the chosen voltage level.

The delays and voltage levels in Figure 4 characterize the critical handshake communications over the long wire. The handshake communication in the top window starts with a low pulse on $Dout_{10}$ that drives a rising transition over the wire, and is followed by a high pulse on $FIRE_1$ that drives a falling transition back over the wire. The longest delay belongs to a rising transition over the 24000 lambda long state wire, and is about 20τ . This delay is so long because the transition does not quite make it to the far end of the wire before the drive pulse turns off. In fact, when the drive pulse turns off, the far end has risen only to about 30% of the supply voltage level. After the drive ceases, the charge in the state wire distributes itself along the length of the wire and relaxes to reach an equilibrium at about 60% of the supply voltage level. It is the charge relaxation that completes the rising transition at the far end of the wire and in doing

so enables the reverse handshake. Because the wire is only partly charged after the rising transition, the delay for the following falling transition is significantly shorter: it takes about 9τ to pull the far end back down to the 50% level.

The simulation results in the bottom simulation window of Figure 4 are similar. We expected the results in the top and bottom windows to be more symmetric. Upon inspection, we noticed that the gate delays in our 6-4 GasP modules are not as well matched as is possible with logical effort. The implications of both simulation results are similar, though.

Figure 5 shows the waveform details for the 24000 lambda simulations of Figure 4. As before, the top window gives the details for the simulation scenario with one full state wire, and the bottom window covers the simulation scenario with one empty state wire.

The upper frames in both the top and bottom windows of Figure 5 show the waveforms for the drive pulses $Dout_{10}$ and $FIRE_1$. A low pulse on $Dout_{10}$ briefly drives P type transistor E in M_{10} and pulls the long state wire high at SUC_{10} . A high pulse on $FIRE_1$ briefly drives N type transistor X in M_1 and pulls the long state wire low at $PRED_1$. Each drive pulse lasts about 5 gate delays. The waveforms at the two wire ends follow in the lower frames.

Note that $Dout_{10}$ is always higher in absolute voltage than $FIRE_1$. This indicates that transistors E and X are never both on at the same time. This behavior is characteristic of GasP circuits, even for short state wires. The result is that transistors E and X that act as the two halves of an inverter exhibit a lower crossover current than an inverter with shared drive. With longer state wires, the crossover current diminishes further and finally disappears, due to the extra wire delay that separates the drive pulses. There is no crossover current between E and X in the long state wire simulations of Figure 5.

The important message in Figure 5 is that the charge on the wire continues to distribute throughout the length of the wire even while the wire is undriven, i.e. even during the periods marked “drift”. Thus, it suffices, during the available 5 gate delay period of drive, to insert or remove enough charge in the wire to ensure that its voltage will be clearly above or below its chosen threshold after the distribution finishes. The amount of charge that a long wire can accept in 5 gate delays is limited by the resistance of the wire rather than by the size of the driver. This is what limits the wire lengths of the uniform, minimum-width metal-2 or metal-3 state wires that we used in Infinity and in the simulation studies reported in this paper. As we pointed out earlier: this limit is somewhere between 24000 and 30000 lambda.

Charge distribution without drive, or “charge relaxation” as we call it in the title of this paper, permits GasP modules to use a single long wire for bi-directional handshake signals.

4.2 Simulated Throughput

The canopy diagrams in Figure 6 show how latency and throughput are affected by the long state wire.

Latency increases by the delay of the wire. With a longer wire, data and bubbles take longer for each trip around the ring because they must pass once through the long state wire. The increased latency lowers the left and right sides of the canopy diagram. This effect appears small because the long wire delay is a relatively small fraction of the total latency around the ring.

The long wire also limits throughput. The impact on throughput is pronounced because each and every element passing through the long wire must use it twice to execute

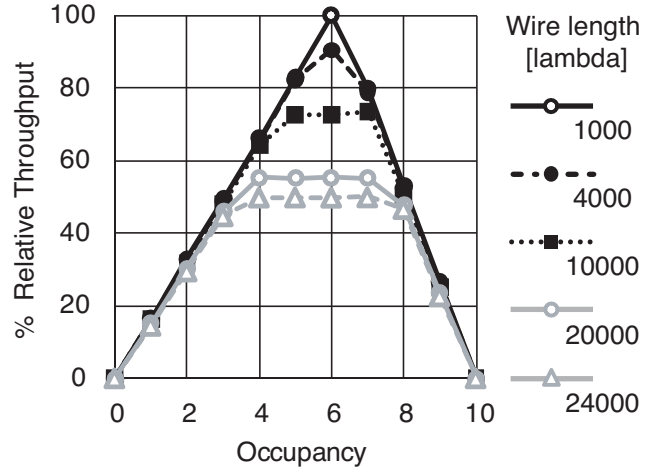


Figure 6 Simulated canopy diagrams for the 10-stage GasP ring of Figure 3 for various lengths of the long state wire. The lower sides show how longer wires slightly increase latency. The flat and lower tops show how longer wires dramatically decrease throughput.

a single-track handshake. Not only is the impact of the long state wire felt twice, but its effect accumulates because queued-up data elements and bubbles must wait for earlier handshakes to complete. This effect shows up dramatically as the flat and lower tops in the canopy diagrams for the longer state wires in Figure 6.

Note that the maximum throughput with the long state wire set to 24000 lambda is about half that of the maximum throughput with the long state wire set to 1000 lambda.

5 Summary: Distance Constraint Graph

Qualitatively, our simulations show that for sufficiently short state wires, the main impact of the wire is to retard the action of its driver. The capacitance of short wires dominates their resistance and so the driver “sees all of the wire”. For such state wires, a larger driver can succeed in driving the wire faster. We can use logical effort to size the driver to provide an acceptable communication delay.

At some medium length of the state wire, however, increasing the wire length fails to retard further the action of its driver. In this regime, the driver saturates, pinning the near end of the wire to the power or ground rail. But the resistance of the wire itself limits the speed with which the far end of the wire can respond. In the 90 nm TSMC CMOS technology that we use, this regime sets in at 5000 lambda. In effect, the driver can no longer see the total capacitance of the wire, but only the capacitance of its near end.

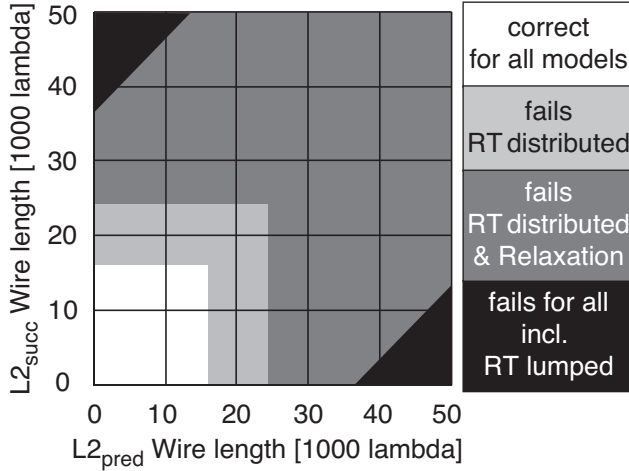


Figure 7 Distance Constraint Graph (90 nm CMOS), showing how the distance $L2_{pred}$ from a given 6-4 GasP module to its predecessor, and likewise $L2_{succ}$, from the module to its successor, are constrained according to:

- (white area)
Relative timing under the distributed RC model, which predicts functionality for distances below 16000 lambda.
- (light-grey area)
Charge relaxation, which predicts functionality for distances up to 24000 lambda.
- (everywhere except black area)
Relative timing under the lumped capacitance model, which predicts functionality, i.e. correct operation, when $L2_{pred}$ and $L2_{succ}$ differ by less than 37000 lambda.

At about 16000 lambda, the delays for rising and falling transitions to the far end of the wire begin to exceed the 5 gate delay self-resetting loops in the 6-4 GasP module. This is the point where the relative timing constraints of Section 3 begin to fail. The wire length is now long enough that the driver shuts off before the far end of the wire has risen or fallen to the 50% voltage level. We now enter the domain of long-range GasP and charge relaxation.

For long state wires, it is the charge on the wire that represents its state, not the voltage. Though the wire is a passive component, it acts through its charge, even when undriven. The charge in the wire distributes itself and relaxes to occupy the full length of the wire. The driver must put enough charge into it so that the wire relaxes within the switching voltage range of the receiver. Our simulations show that the driver can do this for state wires up to about 24000 lambda.

The above summary captures the results of our second study of wire delays in GasP. The distributed RC model that we use for the state wires in our second study significantly reduces the maximum module distance suggested by the simple lumped capacitance model used in our first study [5].⁴

⁴The goal of our first study in [5] was to understand the effects of the

In the lumped capacitance model, the delay of a self-resetting loop depends on the delay of the state wire attached to it. With a lumped capacitance model, the forward self-resetting loop delay always exceeds the forward transfer delay and the backward self-resetting loop delay always exceeds the backward transfer delay. Thus, in this model, a self-resetting loop can never prematurely terminate the drive of “its own” state wire, but the self-resetting loop’s partner still can. Therefore, the worst that can happen is that the delays of the two self-resetting loops in a GasP module differ enough so that the faster loop terminates prematurely the drive of the slower state wire. The lumped capacitance model predicts correct operation provided the *difference in the lengths of the two state wires* is below some maximum, which is 37000 lambda in our technology.

The Distance Constraint Graph [6] in Figure 7 gives a graphical overview of the results of both studies.

6 Conclusion and Future Work

The need to drive a state wire for “long enough” to deliver adequate charge presents a new kind of timing constraint. Unlike most constraints that consider the time of arrival of two signals, this one considers the shut off time of a drive signal. Moreover, it is couched not in terms of something that has happened, but in terms of something that will happen after the signal ceases.

Knowing the importance of charge gives us a better understanding of single-track communication. It is comforting to know that GasP modules can function, albeit with reduced throughput, with state wires so long that their internal delay is nearly a full GasP cycle. Such long wires deliver signals after their drive has ceased.

One might accommodate longer state wires by extending the drive period for a longer time. However, because GasP circuits obtain logical simplicity by representing a send or receive transaction as a pulse of standard duration, we are reluctant to make some GasP modules slower than others.

Exploring wire lengths beyond what might be considered reasonable, and still observing correct GasP functionality, has taught us that GasP designs offer quite a bit of freedom to trade off distance and cycle time.

These results apply equally to other single-track handshake protocols, including [1, 15, 9, 2]. All single-track circuits, even those that are based on the conservative overlapping protocol by van Berkel and Bink in [1], cease to charge or

500 versus 5000 lambda long state wires in Infinity. According to the wire classification in Section 5 these are short wires. The lumped capacitance model was O.K. to use in the context of our first study. But it is not the correct model to understand the delay effects of medium and long wires in GasP, which is why we use a distributed RC model in our second study.

discharge the wire, or wires in case of [9, 2], based only on information available at the near end. The present study extends the operational analysis available for single-track signaling beyond the lumped capacitance model.

We are currently investigating how to add long wires into the logical effort models for logic with interconnect [7], how to support them in our timing analysis tools [4], and how to engineer them to minimize the loss in latency and throughput [8].

Perhaps the most important implication of this study is that *one can engineer the wires!* Indeed, in the current era of circuit technology where the properties of wires dominate delay as well as power and area, one *should* engineer the wires as well as the transistors. For example, a 50% increase in the width and spacing allocated for a 6-4 GasP state wire can improve its delay by a factor of two at an area cost of less than 1% for a single-rail 64-bit datapath. It is wise to allocate more space to a single-track state wire that makes multiple transitions per transaction than to a bundled data wire that makes only one. Improvement of selected state wires may further optimize the design at locations where the bi-directional single-track signaling would otherwise fail to keep up with the uni-directional data, as for example shown in [3]. Understanding how to use wire engineering to improve distance and cycle time permits us to reason about and optimize the insertion and placement of GasP repeaters.

We hope soon to build a next test chip to confirm that real GasP circuits can indeed operate with long state wires. This chip will also explore how engineering the wires can improve performance. We hope that measurements from such a chip will shed light on the speed, the power demand, and the reliability of long-range GasP circuits.

Acknowledgments We gratefully acknowledge Jon Lexau for (*running*)* our 90 nm simulations. Sun Microsystems paid for the design, fabrication and testing of Infinity, and also sponsored the present work through a generous grant to the Portland State University Foundation.

References

- [1] K. v. Berkel and A. Bink. Single-Track Handshake Signaling with Application to Micropipelines and Handshake Circuits. In *Proc. Advanced Research in Asynchronous Circuits and Systems (ASYNC)*, pages 122–133, 1996.
- [2] M. Ferretti and P. Beerel. High Performance Asynchronous Design Using Single-Track Full-Buffer Standard Cells. *IEEE Journal of Solid-State Circuits*, 41(6):1444–1454, 2006.
- [3] R. Ho, J. Gainsley, and R. Drost. Long Wires and Asynchronous Control. In *Proc. IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 240–249, 2004.
- [4] P. Joshi. Static Timing Analysis of GasP. Master of Science thesis, Electrical Engineering, Faculty of the USC Viterbi School of Engineering, University of Southern California, December 2008.
- [5] P. Joshi, P. Beerel, M. Roncken, and I. Sutherland. Timing Verification of GasP Asynchronous Circuits: Predicted Delay Variations Observed by Experiment. In *D. Dams, U. Hannemann, M. Steffen (Eds.): Willem-Paul de Roever Festschrift, LNCS 5930*, pages 260–276. Springer-Verlag Berlin Heidelberg, 2010.
- [6] S. Mettala Gilla. Distance Constraint Graph: A Graphical Representation for 6-4 GasP showing how Relative Timings constrain the Module Distances. Technical Report, ARC2009-smg01, Asynchronous Research Center, Portland State University, September 2009.
- [7] A. Morgenshtein, E. Friedman, R. Ginosar, and A. Kolodny. Timing Optimization in Logic with Interconnect. In *Proc. International Workshop on System Level Interconnect Prediction (SLIP)*, pages 19–26, 2008.
- [8] F. Mu and C. Svensson. Analysis and Optimization of a Uniform Long Wire and Driver. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 46(9):1086–1100, 1999.
- [9] M. Nyström, E. Ou, and A. Martin. An Eight-Bit Divider Implemented with Asynchronous Pulse Logic. In *Proc. IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 229–239, 2004.
- [10] A. Peeters and K. van Berkel. Single-rail Handshake Circuits. In *Proc. Asynchronous Design Methodologies*, pages 53–62, 1995.
- [11] J. Rabaey, A. Chandrakasan, and B. Nikolić. *Digital Integrated Circuits: A Design Perspective*. Prentice-Hall, 2003.
- [12] K. Stevens, R. Ginosar, and S. Rotem. Relative Timing. In *Proc. IEEE International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC)*, pages 208–218, 1999.
- [13] K. Stevens, R. Ginosar, and S. Rotem. Relative Timing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(1):129–140, 2003.
- [14] I. Sutherland. A Six-Four GasP Tutorial. Technical Report, UCIES2007-is49 at <http://fleet.cs.berkeley.edu/docs>, 2007.
- [15] I. Sutherland and S. Fairbanks. GasP: A Minimal FIFO Control. In *Proc. IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 46–53, 2001.
- [16] I. Sutherland, B. Sproull, and D. Harris. *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann, San Francisco, 1999.
- [17] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley, 2005.