Temporal Codes in On-Chip Interconnects

Michael Mishkin*, Nam Sung Kim† and Mikko Lipasti*
*University of Wisconsin Madison, †University of Illinois Urbana Champaign
mmishkin@wisc.edu, nskim@illinois.edu, mikko@engr.wisc.edu

***Abstract*—Dynamic power consumption associated with signal toggles over long distance wires accounts for a significant portion of on-chip interconnect power. Improving dynamic energy efficiency in highly capacitive interconnects can be achieved by reducing the toggle rates associated with data communication. Temporal coding schemes facilitate bounded activity factors by encoding information as placement of signal toggles in time and can thereby improve the energy efficiency of data communication by encoding multiple bits per toggle.**

We introduce two temporal protocol variants designed for traversal of the crossbars in on-chip networks. These protocols reduce peak power without loss of bandwidth and achieve energy efficient on-chip communication in high capacitance long distance interconnects. Extending these energy savings to a multi-hop mesh topology is achieved by router implementations equipped with bypassing mechanisms that elide per hop reencoding overheads. We demonstrate a four bit per transition temporal protocol with up to 75% communication energy reduction that can be achieved over a baseline serial bit stream protocol.

I. Introduction

As processors have scaled to increasingly many cores per chip, packet switched networks have emerged as the preferred on-chip interconnect technology [1], [2]. These on-chip networks support the high bandwidth communication requirements of data movement between processor cores and memory. On-chip data movement is associated with high dynamic power [3], [4], [5] due to bit toggles over long distance interconnects [6]. Temporal coding aims to reduce this energy component by reducing the number of bit toggles involved in on-chip communication [7]. Temporal codes reduce toggle rates by representing multi-bit sequences as signal toggles placed in time. This reduces the peak dynamic power associated with movement of bits across the interconnect and under certain circumstances improves energy efficiency.

Figure 1 illustrates transmission of a signal toggle placed within a symbol window that corresponds to an encoded four bit sequence. Each transmission window contains a sequence of symbol windows, and each symbol window represents a distinct value. Protocols that encode more bits per toggle are more energy efficient. Encoding more bits per toggle involves including more symbol windows per transmission window. As the temporal window size increases, the latency of the temporal window can increase exponentially with the number of bits encoded. Operating at higher temporal resolution counteracts the latency degradation and is possible because the energy efficiency of the protocol enables its transmitters and receivers to operate at higher clock frequencies without exceeding the power budget. However, this temporal resolution is constrained by the transmitter and receiver circuit designs and the precision with which the signal propagation delay jitter can be bounded. In fully synchronous designs, the temporal resolution is the same as the transmitter and receiver clock frequency.

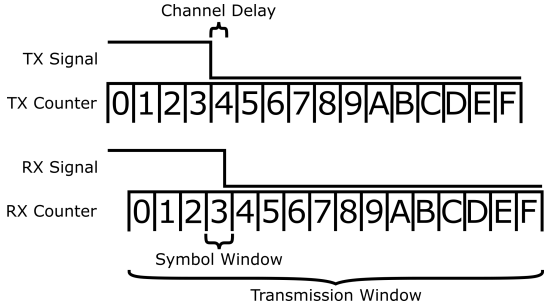


Figure 1: A toggle placed within a window containing 16 symbols.

978-1-5090-6023-8/17/$31.00

At a fixed temporal resolution, an increase in transmission window size is associated with a decrease in per wire bandwidth. Meeting a target channel bandwidth thus requires increasing the number of wires per port as the number of bits per transition increases. Within point-to-point links, there is no increase in per wire capacitance associated with a widening channel. However, for crossbars the increase in channel width increases the area of the crossbar, thus increasing the capacitance associated with crossbar traversal. In systems containing both long distance wires and crossbars, the optimal protocol is a function of both the link lengths and the crossbar radix. However, for networks with sufficiently long links and low radix, the crossbar traversal energy can become insignificant compared to link traversal.

Temporal protocols are efficacious in interconnects with long distance links because the high capacitance of the links is associated with high enough dynamic energy to outweigh the transmitter and receiver energy overheads. Similar wire distances may be traversed in high diameter topologies, like meshes, as in high radix topologies, like flattened butterflies [8], however similar energy savings are not achieved due to per hop transmission overheads. We will discuss bypass mechanisms that allow temporal signals to continue propagation through routers without incurring these overheads.

Section II discusses the design of temporal protocols built for crossbar traversal. Section III considers router design for temporally encoded transmissions in NoCs including bypass mechanisms for reducing per-hop energy and latency overheads. In section IV, an energy model based on post-synthesis measurements is developed and used to model the NoC crossbars and links. Section V presents performance and energy results for temporal coded transmission within the on-chip network of a tiled 64 core processor.

II. Protocol Variants

The design of temporal codes for on-chip networks must consider both crossbar traversal and communication over point-to-point links. For point-to-point links, the voltage levels at the beginning of transmission are the same as the levels at the end of the previous transmission. In packet switched networks each router output port can be connected to one of several crossbar inputs. When a crossbar switches between two input ports with different signal levels it results in bit flips at the output ports that do not encode any information. Preventing misinterpretation of these bit toggles can be accomplished either by protocol design or by circuit design. In this section, we consider the circuit level implications for two protocol variants that we refer to as return-to-zero (RZ) and non-return-to-zero (NRZ).

The RZ protocol always includes a negative signal transition for every positive signal transition such that at the end of transmission all signal levels have returned to zero. The RZ protocol divides the transmission window in half such that each toggle encodes one fewer bit than would be encoded by a protocol that placed just one toggle within a transmission window of the same size. Consequently, the number of symbols that can be represented per toggle is reduced. This difference in toggle efficiency represents a lower dynamic energy efficiency for RZ protocols. However, the per wire bandwidth is increased and fewer wires are required to achieve an equivalent bandwidth, since more bits are encoded per transmission window containing two toggles. The RZ protocols always contain both a positive and negative toggle for every transmitted symbol and therefore consume the same amount of energy for each transmission.

The NRZ protocols require additional logic to prevent propagation of false toggles caused by input selection adjustments between signals that can begin transmission as either a one or a zero. The NRZ protocols also include a symbol that is represented by no signal transition at all. The presence of a "no transition" symbol in the NRZ protocols provides an opportunity for reduction of channel energy consumption beyond what is achieved by transmitting fewer bits per toggle alone. The best choice of value for the "no transition" symbol is the bit sequence that appears most frequently, which for many applications is a sequence of all zeros.

The energy efficiency of temporal protocols over long links is plotted in Figure 2. The activity unit is the probability of zero bits. This is because the toggle activity factors for serial bit streams and temporal protocols are not equivalent, however each can be calculated based on a Bernoulli distribution of bit levels. This way of representing the bit-stream variability does not represent the full range of activity factors for the baseline because bit streams with very high activity factors cannot be represented as streams of independent bit values. The maximum expected activity factor of a stream of independent bits is only 50%, while the peak baseline power actually occurs at 100% activity and is roughly double what is shown in Figure 2. For temporal protocols, the peak power occurs when there are no transmissions of "no transition" symbols. The peak power for the temporal protocol can be seen on the right side of Figure 2, where temporal protocols with toggle efficiency greater than 2 bits per toggle are observed to use significantly less energy than the baseline with 50% activity over long links.

At zero link length, the crossbar traversal and transmission overhead energies for temporal codes is greater than the baseline for all zero bit probabilities. As link length increases, the channel energy dominates the transmission overhead and the temporal protocols are more energy efficient than the baseline for a range of activity factors that increases with the link length. For link lengths in excess of 2mm, the 4bNRZ protocol has the lowest peak power of all of the temporal protocols considered, and continues to be the lower power protocol as the link length increases. If protocols with greater toggle efficiency were considered then they would demonstrate further peak power reductions. However such protocols are not considered as their minimum transmission latency would exceed the latency constraint of the test-bench used to generate the data for these graphs.

A crossbar scaling study is shown in Figure 3, where the energy per bit is shown across zero bit probabilities for a set of channel widths, with zero additional link length. For large crossbars, temporal coding exhibits lower energy crossbar traversal than the baseline. Where this differs from link length scaling is that the minimum peak power for temporal protocols traversing larger crossbars is achieved at 3 bits per transition. For higher probability of zero bits, the 2 bits per transition protocol becomes slightly more energy efficient than 3 bits per transition because of its higher likelihood of transmitting "no transition", but the peak exceeds that of 3 bits per transition. The greater energy consumption of the 4 bit per transition protocol's crossbar traversal is due to the wider channel widths associated with low per wire bandwidths, that increase the size of the crossbar. Protocols that encode more than four bits per transition are expected to exhibit even higher energy crossbar traversals due to the increasing crossbar area associated with even lower per wire bandwidths.

In any system where the dominant component of transmission energy is either the transmission overhead or the crossbar traversal, a protocol with fewer bits per transition tends to be optimal. Protocols with many bits per transition are only the optimal low energy protocols within systems where either the dominant component of transmission energy is a point-to-point link or where the latency constraint is sufficiently relaxed that increasing the bits per transmission does not require increasing the channel width. The following section addresses design considerations associated with temporal codes in on-chip networks in latency constrained systems, whose long links account for the dominant dynamic energy component of data transmission.

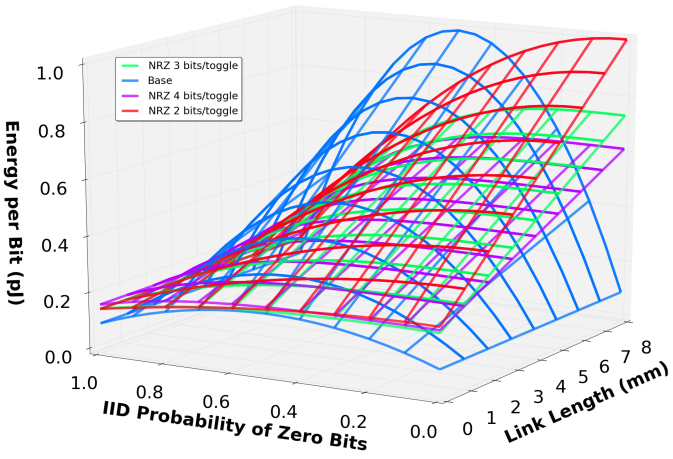


Figure 2: Link Length Scaling of a 128 bit per cycle link: Energy per bit as a function of link length and probability of zero bits.

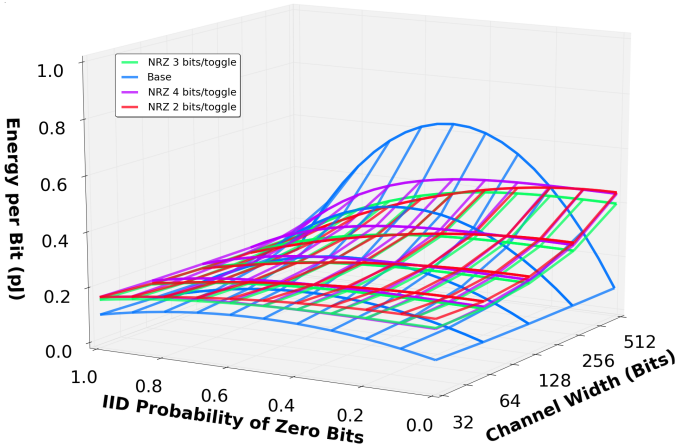


Figure 3: Crossbar Scaling: Energy per bit for traversal of a 16x16 crossbar as a function of bit width and probability of zero bits.

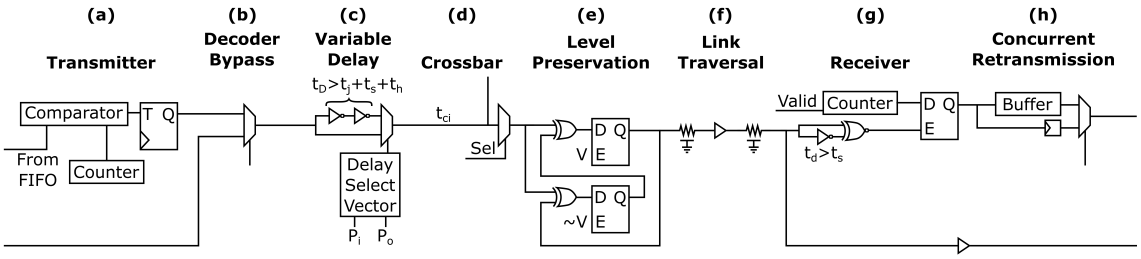


Figure 4: Temporally Encoded Communication Pipeline

III. Router Design

A schematic of one hop in the temporally encoded NoC is shown in Figure 4. Each labeled component is part of the router data path designed for transmission of temporal codes. The data plane operates at a higher frequency than the control plane for high temporal resolution. Minimizing the clock signal fan-out to the transmitters and receivers saves energy so the only clocked elements that operate at this higher frequency are the counters and transmission flip-flops. The control plane contains low frequency sequential elements pertaining to crossbar allocation, signal validity, and data plane configuration.

Transmitter: The transmitter converts a sequence of bits from a binary encoding to a temporal encoding by placing a toggle within the transmission window. A counter iterates through symbol windows in a designated order while a comparator watches for a match between the output of the counter and an outgoing bit sequence. Using a grey code counter minimizes toggles on the counter output. When a transmitter's comparator indicates a match, a T-flip-flop will toggle its output at the next clock tick, as is shown in Figure 4a. Multi-bit transmissions are divided into bit sequences that are transmitted by a set of parallel transmitters, comprising an input port of the router. The counter output can be shared among all transmitters of the same input port. When a port is inactive, the port's counter is clock gated. Counters are not shared between adjacent ports to prevent active counter fanouts to inactive ports.

Crossbar: Crossbars designed for temporal codes require mechanisms to prevent the signal toggles resulting from crossbar connection updates from reaching the receiver and being misinterpreted as data. While handling this at the receiver would be functionally correct, preventing propagation of these toggles across links saves energy. We have considered two temporal protocol variants that handle the inconsistent connection. The RZ protocol uses an even number of transitions per wire and can be implemented using standard crossbars but is less energy efficient than NRZ protocols. The NRZ protocols may incur an odd number of transitions within a given transmission but require level preservation logic.

A latch based implementation of the level preservation logic is shown in Figure 4e. When multiplexer control updates occur, the valid signal is deasserted, the output of the circuit is held constant, and any toggles on the input are stored in a secondary latch. When the valid signal is reasserted, the level preservation circuit will once again allow toggles at its input to propagate to the receiver as data.

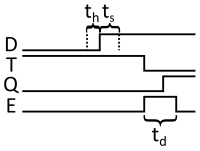


Figure 5: The transmitted signal toggle (T) must arrive outside of the setup and hold window of the receiver, illustrated in Figure 4g.

Receiver: The temporal decoders are storage elements with two inputs: one input is provided by the fully synchronous counter and the other input is the temporally encoded signal. The counter sequence begins after a valid signal is asserted at the input port. The output of the receiver is the value from the counter that was stored upon the arrival of a signal toggle. The decoder is a latch based circuit so there are restrictions on when transitions can arrive at the input terminals. Signal toggles must occur within a window of counter stability in which the counter value does not change. This window of stability excludes the setup and hold time of duration $t_s + t_h$ in Figure 5.

The receivers are based on pulsed latches where a pulse generated by a transition detector at the time of an encoded signal toggle triggers storage of the counter values at that moment. In order for the receiver to operate properly, the transition must arrive while the counter value is stable, and the duration of the pulse output by the transition detector of Figure 4g must exceed the setup time of the latch to store the counter value while the pulse is asserted. Therefore, timing closure must guarantee that the transition arrival time satisfies the following inequality in order for the the counter value to be properly stored within the latches.

$$t_D^N + t_s < t_T + t_d < t_D^{N+1} - t_h$$

Router: The temporal receivers and transmitters are both located at the input ports of the router such that the transmitter for a subsequent hop can interact directly with the colocated receiver of the previous hop. During periods of low contention, incoming flits can be passed directly from the receiver to the transmitter without utilizing the input buffers. This is analogous to the bypass paths which are a common NoC router optimization that lower per-hop latency under low contention and save power by eliminating extraneous buffer accesses [9], [10]. In addition to reduction of latency and buffer utilization, bypassing can be used to forward a temporally encoded signal directly to the next hop without decode and retransmission, thus eliminating the transmission overhead at intermediate hops when router contention is low. Temporal decoder bypassing cannot be interrupted by any pipeline stalls due to the temporal nature of the encoding. If a stall occurs, then the flit needs to be decoded and retransmitted after the stall has been resolved.

The decoder bypassing mechanism's schematic is illustrated in Figure 4b. To facilitate a low latency pipeline with minimal stalling, a small header flit is transmitted ahead of the temporally encoded payload. In the absence of contention the router can be speculatively configured one cycle ahead of the temporal signal transmission. Routers using a static dimension order routing algorithm simplify lookahead route computation (RC). If speculative virtual channel allocation (VA), switch allocation (SA), and switch traversal (ST) [11] succeed then the header flit is routed in a single cycle to the next hop and the incoming temporal transmission that follows can bypass decoding, as is shown in Figure 6a.

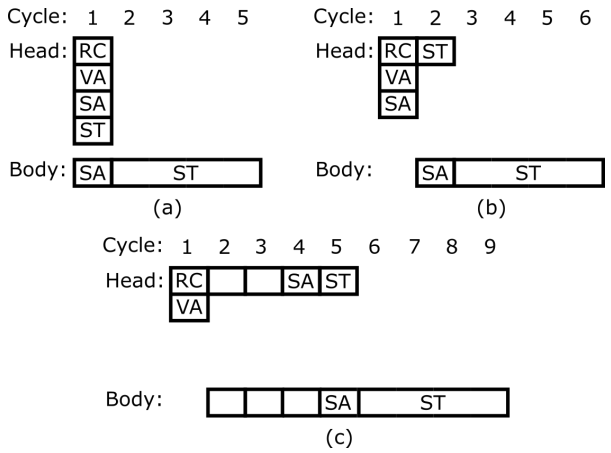
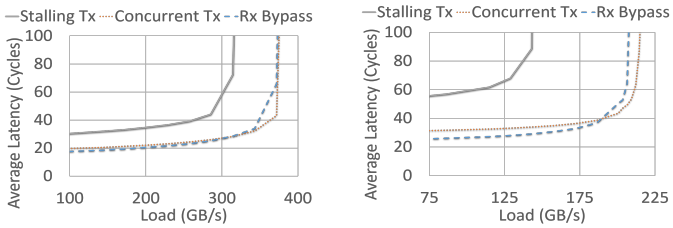


Figure 6: Gantt charts describing minimum latency for a) decoder bypassing, b) early retransmission, and c) no bypassing.



(a) Flattened Butterfly (b) Mesh

	Mesh	FBFly
Ports per router	5	10
Concentration	1 core/router	4 cores/router
Virtual Channels	2 per port	4 per port
Routing Algorithm	XY	XY

(c) Booksim Configuration

Figure 7: Throughput vs Latency with uniform random traffic in 64 node (a) flattened butterfly and (b) mesh topologies.

If the header flit stalls, then the decoder bypass path cannot be used and the temporally transmitted signal must be decoded at an intermediate receiver. If the header flit is then granted switch allocation before the multi-cycle temporal transmission has been fully decoded, a different bypass mechanism can be used to begin retransmission before decoding has completed. This bypass mechanism is depicted in Figure 4h. The bypass path links the output of the decoder to the input of the encoder through an initially transparent latch. At the end of the receiver's transmission window, the latch holds the decoded value at the input of the encoder for the duration of the transmitter's transmission window.

During concurrent retransmission, the transmitter sequence for the subsequent hop always starts once the receiver sequence is already in progress. When the transmitter's counter reaches the symbol to be transmitted, the received symbol is guaranteed to be ready. Preventing interpretation of incorrect values requires that either the receiver be initialized to the value of the "no transition" symbol or an additional ready bit should be provided by the receiver to prevent interpretation of an invalid value prior to receiving a toggle. As is shown in Figure 6b, the minimal latency when early retransmission is invoked is one cycle more than decoder bypassing because the receiver must begin decoding ahead of the retransmission.

When next hop allocation does not take place before the decoding completes, the temporally encoded flit is inserted into the input buffer to be retrieved once allocation of the flit is successful. Without active decoder bypassing, each flit needs to be fully decoded before being retransmitted. Therefore, the minimal router latency without bypassing shown in Figure 6c depends upon the size of the transmission window. Early retransmission bypassing lowers the router latency and slightly lowers the energy by avoiding the input buffers. Decoder bypassing further lowers router latency by allowing single cycle routing and also lowers energy by eliminating the retransmission overhead during periods of low router contention.

Figure 7 is a comparison of the latency and throughput responses of two 64 node on-chip network topologies. Figure 7a shows the results of Booksim [11] experiments on a flattened butterfly [8] topology and Figure 7b shows the results of experiments on a mesh topology. The number of virtual channels is selected for equivalent aggregate input buffer capacity. The latency reduction achieved with decoder bypassing is more significant in the mesh topology than the flattened butterfly. Due to the long distance links in the flattened butterfly topology, the intended benefits of decoder bypassing are mostly already achieved by having few hops between distant nodes. For the mesh, however, there are many hops between nodes and each hop potentially requires decoding and reencoding. Under low load conditions transmission latency comparable to the flattened butterfly can be achieved by a mesh topology with decoder bypassing.

Multicycle Paths: Temporally encoded transmissions originate at a router input port's transmitter, then traverse the router's crossbar and the link before arriving at the input ports of the next hop's router, where the signals are either decoded or bypass to the next hop. The path from a transmitter to a receiver can be multiple clock cycles of propagation delay. The timing requirements for these paths are dictated by the setup and hold times of the receivers. Typical single cycle path timing requirements constrain propagation delays to be less than a maximum delay that is slightly less than the clock period. Multicycle path timing requirements are different in that the multicycle timing constraint allows the signal arrival time to exceed the clock period. The timing constraint requires that a signal transition does not fall within the setup and hold window. If a path would violate this setup and hold window constraint, then the violation can be resolved by inserting a delay along the path that pushes the arrival time into the next clock cycle.

Selecting an appropriate delay along multiplexed paths can pose a challenge when different delay requirements must be satisfied along the same path. In a sufficiently large crossbar, different paths through the crossbar can have different delays. For example, the delay difference between adjacent input ports to a given receiver may be some δt such that the set of delays for all ports spans a range that could intersect the receiver setup and hold window for some subset of input ports. Some additional delay needs to be added to the paths that would otherwise violate the timing constraint to push the arrival time of the signal toggle to be within the period of stability for the receiver. However, the presence of shared paths through the crossbar restricts the locations at which delay elements can be placed without affecting multiple paths through the crossbar.

In a multiplexer based crossbar, each multiplexer input corresponds to only one transmitter and receiver pair, so fine tuning of the path delays could be implemented at these locations. However, since there are as many multiplexer inputs as the product of the number of input and output ports of the crossbar (NxM), it is possible that fewer, slightly larger, variable delay elements would be required if they were placed external to the crossbar.

Figure 4c depicts a delay tuning approach that places a variable delay element at each crossbar input port. The variable delay setting needs to be a function of the input port and output port to account for all delays through the crossbar. Placing the variable delay element before link traversal, at the crossbar input port itself directly maps each delay element to an input port, thereby simplifying the variable delay selection to a function of only the output port, where a separate function is used for each input port. This port to delay mapping can be stored concisely as a bit vector. The variable delay elements only require two possible delay settings, where one delay is minimal and the other exceeds the setup and hold time of the receiver. So each delay setting can be expressed as a single bit. If the routers are small enough then the output port to delay mappings could even be shared among the inputs of a given router since the delay differences from the input ports to the next hop would be primarily a result of link length differences.

The topologies that benefit most from decoder bypass paths have low radix routers and therefore have low arrival time variability among the input ports of the router. With arrival times at the receiver bounded within an acceptable window, the delay of the decoder bypass paths can be set such that the same variable delay setting would be appropriate for both a bypassed signal and a locally generated transmission. Properly bounding the arrival time window to align a fixed delay bypass path may require variable delay elements with more than two delay settings.

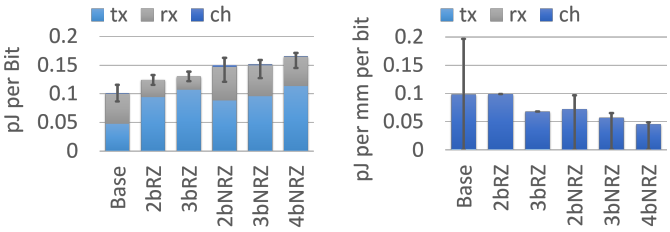
The variable delay elements may also be used to enable multicycle paths in DVFS enabled NoCs [12], [13]. Due to the differences in propagation times that result from scaled voltage levels and the different relative clock phases associated with scaled clock frequencies, different delay settings may be required for different DVFS levels. The variable delay elements provide the adaptability necessary to accommodate DVFS in the interconnect. Adjustments to the DVFS settings require an update to the delay select vectors. The proper delay configurations for each voltage/frequency level can be determined during timing closure and simply loaded into the delay select vectors whenever a change to DVFS is made at run time.

IV. Experiments

The energy characterizations used to generate Figure 2 and Figure 3 were derived from a set of experiments conducted using Synopsys Design Compiler based on descriptions of a point-to-point link between representative implementations of a transmitter and a receiver. The link is represented as a sequence of wire segments and repeaters, annotated with wire loads according to parameters obtained from cacti [14] pertaining to global wires at the 40nm technology node with a pitch of 320nm, resistance of 254.5m$\Omega/\mu m$, and capacitance of 0.283 $fF/\mu m$. The experimental configuration assumes a system clock frequency of 700MHz. The temporal protocols operate at a temporal resolution that is 4x the system clock frequency. The models are synthesized in Synopsys Design Compiler using a 40nm TSMC technology library.

The power measurements obtained from several experimental configurations form the basis for a parameterized model of energy characteristics for each protocol. The energy characterization involves solving for coefficients of the proportionality between energy and the product of capacitance and activity, $E \propto AC$, using linear regression. Each protocol's energy profile can then be expressed in terms of parameters such as transmission overhead and channel traversal energy based on this characterization, shown in Figure 8.

The energy characterization is the basis for a power model used to analyze the energy associated with interconnect traffic in a simulated



(a) Overhead Energy per Bit independent of link length.

(b) Channel Traversal Energy per bit per mm of link length.

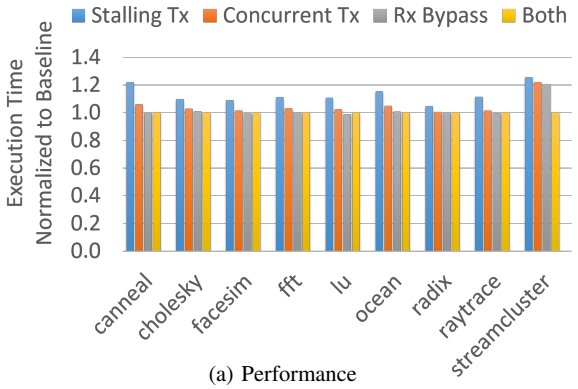
Figure 8: The energy characterization of temporal protocols includes overhead and channel length dependent components. Error bars represent the range of possible activity factors and the main bar represents 50% probability of zero bits.

64 core processor with an 8x8 mesh topology. The simulator used for this study is sniper [15] with an interconnect model based on router port contention that assumes serialization of all packets arriving at a router destined for the same output port. Structural hazards such as input buffer capacity and virtual channels are not modeled. Whenever a flit arrives at a router its wait time is calculated based on the other flits waiting at the router and the available bypass mechanisms. The wait time is used to determine which bypassing mechanism is used. The possible responses to flit arrival include decoder bypassing, concurrent retransmission, and buffering. Each has its own associated hop traversal energy calculated based on the energy characterizations of each protocol. Input buffer write and read energies were obtained using FabMem [16]. We assume that only data packets are transmitted as temporal codes and that packets with smaller payloads, like coherence messages, are communicated as binary voltage levels. The results of experiments pertaining to the impact of using temporal coding for data messages is described in the following section.

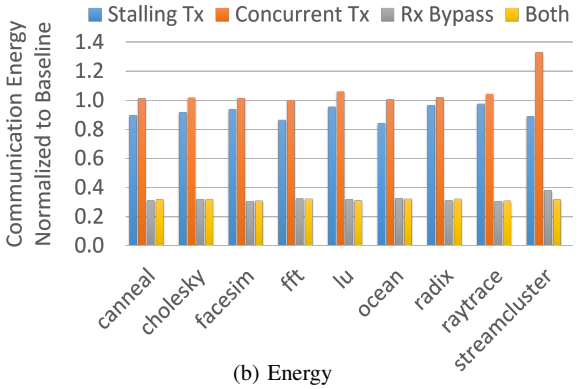
V. Results

The experiments consider four different bypassing capabilities for the 4bNRZ protocol, one without bypassing (Stalling Tx), one with only concurrent retransmission (Concurrent Tx), one with only decoder bypassing (Rx Bypass), and one with both of the bypassing mechanisms active (Both). The baseline must be latched at each hop but is assumed to use the same speculation techniqeus as for temporal protocols with both types of bypassing mechanisms active. The reported energy components pertain to interconnect data path traversal including links, crossbars, transmitters, receivers and buffers. 75% probability of zero bits is assumed for all energy calculations, which is consistent with the observation made in [7] that roughly 30% of four bit sequences in L2 cache data are all zeros. Links between nodes are assumed to have a length of 1mm.

The results in Figure 9a demonstrate speedups for many of the benchmarks when bypassing mechanisms are active. Figure 9b shows the dynamic energy estimates associated with communication over the interconnect. The substantial energy savings observed for configurations with decoder bypassing demonstrates that network contention levels are low enough to benefit from decoder bypassing in an 8x8 mesh topology, even for these high communication workloads. The effectiveness of bypassing would be degraded by resource limitations of the router implementation, but this data demonstrates the opportunity for energy savings within router designs similar to Figure 4.



(a) Performance



(b) Energy

Figure 9: Data collected using Sniper for 64 core processor with a mesh interconnect topology running communication heavy benchmarks selected from the Parsec [17] and Splash [18] suites.

VI. Conclusion

Temporal coding schemes are an energy efficient means of on-chip communication for systems where high activity factors of binary bit streams along highly capacitive interconnects results in high dynamic power consumption. Temporal codes that reduce the activity factors associated with data communication reduce peak power and improve energy efficiency. Designing an NoC to support temporal codes requires protocols capable of traversing crossbars without risking propagation of false toggles during crossbar control updates, which can require increased router complexity. Within on-chip networks, a protocol that requires more complex routers but improves energy efficiency is preferable because the highly capacitive links of the NoCs represent a more significant energy component than the overhead of the router modifications. In high diameter networks, where the accumulated per hop overheads can be significant, the latency and energy overheads are counteracted by using bypass paths. The long multicycle paths that decoder bypassing creates have timing constraints that are resolved at run-time using variable delay elements within the data path. Based on the energy characterizations of representative transmitter and receiver implementations, our energy models of on-chip networks demonstrate that temporal codes can reduce peak power and improve energy efficiency for on-chip data communication in NoCs, without sacrificing link bandwidth or latency. The four bit per toggle NRZ temporal protocol is shown to have the lowest peak power of the protocols considered and to exhibit up to 75% energy savings over a baseline serial bit stream protocol for communication heavy workloads on a 64 core processor with an 8x8 mesh topology.

Acknowledgements

This work was supported in part by NSF (CCF-1615014, CCF-1628384, CCF-1600896 and CNS-1600669).

References

[1] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, 2001, pp. 684–689.

[2] S. Heo and K. Asanovic, "Replacing global wires with an on-chip network: a power analysis," in *ISLPED '05. Proceedings of the 2005 International Symposium on Low Power Electronics and Design*, Aug 2005, pp. 369–374.

[3] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," in *SLIP '04. Proceedings of the 2004 International Workshop on System Level Interconnect Prediction*, 2004, pp. 7–13.

[4] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff, "Energy characterization of a tiled architecture processor with on-chip networks," in *ISLPED '03. Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, Aug 2003, pp. 424–427.

[5] M. Buckler, W. Burleson, and G. Sadowski, "Low-power networks-on-chip: Progress and remaining challenges," in *ISLPED '13. Proceedings of the 2013 International Symposium on Low Power Electronics and Design*, 2013, pp. 132–134.

[6] G. Pekhimenko, E. Bolotin, N. Vijaykumar, O. Mutlu, T. C. Mowry, and S. W. Keckler, "A case for toggle-aware compression for gpu systems," in *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA)*, March 2016, pp. 188–200.

[7] M. N. Bojnordi and E. Ipek, "Desc: Energy-efficient data exchange using synchronized counters," in *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2013, pp. 234–246.

[8] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2007, pp. 172–182.

[9] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *MICRO-36. Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec 2003, pp. 105–116.

[10] A. Kumary, P. Kunduz, A. P. Singhx, L. S. Pehy, and N. K. Jhay, "A 4.6tbits/s 3.6ghz single-cycle noc router with a novel switch allocator in 65nm cmos," in *Proceedings of the 25th International Conference on Computer Design*, Oct 2007, pp. 63–70.

[11] N. Jiang, J. Balfour, D. U. Becker, B. Towles, W. J. Dally, G. Michelogiannakis, and J. Kim, "A detailed and flexible cycle-accurate network-on-chip simulator," in *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, April 2013, pp. 86–96.

[12] A. K. Mishra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan, and C. R. Das, "A case for dynamic frequency tuning in on-chip networks," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2009, pp. 292–303.

[13] X. Chen, Z. Xu, H. Kim, P. Gratz, J. Hu, M. Kishinevsky, and U. Ogras, "In-network monitoring and control policy for dvfs of cmp networks-on-chip and last level caches," in *Proceedings of the IEEE/ACM Sixth International Symposium on Networks-on-Chip*, May 2012, pp. 43–50.

[14] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Cacti 6.0: A tool to model large caches," *HP Laboratories*, pp. 22–31, 2009.

[15] T. E. Carlson, W. Heirman, S. Eyerman, I. Hur, and L. Eeckhout, "An evaluation of high-level mechanistic core models," *ACM Transactions on Architecture and Code Optimization (TACO)*, 2014.

[16] T. Shah, "Fabmem: A multiported ram and cam compiler for superscalar design space exploration," Master's thesis, Department of Electrical and Computer Engineering, North Carolina State University, May 2010.

[17] C. Bienia, "Benchmarking modern multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[18] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The splash-2 programs: characterization and methodological considerations," in *Proceedings of the 22nd Annual International Symposium on Computer Architecture*, June 1995, pp. 24–36.