This is the author's version of the work. The definitive work was published in *Proceedings of the IEEE International Field-Programmable Custom Computing Machines Symposium (FCCM'12), pp. 101-108, Toronto, Canada, April 29th - May 1st, 2012.*

Power Management Strategies for Serial RapidIO Endpoints in FPGAs

Moritz Schmid, Frank Hannig, and Jürgen Teich
Hardware/Software Co-Design, Department of Computer Science,
Friedrich-Alexander University Erlangen-Nürnberg (FAU), Germany
Email: {moritz.schmid, hannig, teich}@cs.fau.de

***Abstract*—We propose a novel data budget-based approach to dynamically control the average power consumption of Serial RapidIO endpoint controllers in FPGAs. The key concept of the approach is to not only perform clock-gating on the FPGA-internal components of the communication controller, but to disable the multi-gigabit transceivers during idle periods. The clock synchronization, inherent to serial interfaces, enables us to omit the often needed periodic link sensing, and only enable the controller according to a predefined schedule to transmit the allocated amount of data during a specific interval. Following this approach we are able to reduce the dynamic power consumption by up to 77 % on average.**

I. Introduction

The growing desire for high performance computing has encouraged a huge demand for very fast interconnects in embedded systems. Especially, the area of digital signal processing requires computing platforms that deliver high performance but also predictable system behavior to guarantee certain service requirements. It is meanwhile common to design custom computing platforms, composed of heterogeneous hardware components, connected by a serial interconnect. Examples for such computing platforms can be found in medical image processing or in surface radar processing stations [2]. A very popular interconnect capable to fulfill the requirements of such systems is the Serial RapidIO (SRIO) communication standard [8].

Although some research has been conducted on the abilities of SRIO as a high-speed interconnect, to the best of our knowledge, none of these studies have focused on the power requirements of the protocol. As opposed to large scale packet-switched networks, data-networks in embedded systems are commonly over provisioned and provide much higher data rates than it is actually necessary for the operation of the system. As a result, the network controller is alternating between periods of data transmission and idle listening. Such periods of idle listening are the main source for energy waste in communication controllers and can be exploited to save energy by disabling the controller for as long as possible.

In this work, we propose a novel budget-based power-management scheme for SRIO endpoints in FPGAs to exploit the clock synchronizing characteristics of serial interconnects and available a priori knowledge of the expected communication rates to schedule active and idle periods. We define a data rate budget to dynamically control the activity schedule of SRIO endpoints to manage the average power consumption of the interconnect.

To summarize our contributions:

1) We present an analysis of the power requirements in Serial RapidIO endpoints.
2) We propose a novel budget-based power management scheme for SRIO FPGA endpoints.

The rest of this paper is organized as follows. Section II discusses related research and publications on Serial RapidIO, low power communication protocols and low power design techniques. Section III introduces the fundamentals of the SRIO protocol and identifies possible starting points for SRIO power management. Section IV elaborates on power management strategies for SRIO FPGA endpoint controllers. To support the proposed strategies, Section V presents evaluation results which are discussed in the conclusion in Section VI.

II. Related Work

The RapidIO (RIO) Interconnect Specification, released by the RapidIO Trade Association, is an open standard, developed to achieve high-performance, low-cost, as well as reliable and scalable system connectivity in embedded systems, networking applications and communication devices. Several works are concerned about the applicability of SRIO for high performance embedded systems, such as novel imaging systems [9] or spatial radar applications [2]. However, to the best of our knowledge, no research has been conducted to involve SRIO in a low-power communication scheme, yet.

In contrast, most research on low power communication schemes and protocols was conducted in the area of wireless sensor networks, mobile and automotive communication, since these environments must cope with a limited energy supply. A key concept in power-aware communication is the introduction of idle periods in which no communication happens, and thus, the transceiver can be turned off or at least be transferred into a low power state [7]. Unterassinger et al., for example, design a power management unit for ultra-low power wireless sensor networks, which is based on multiple power modes for the transceivers in [11]. El-Hoiydi et al. design a novel medium access protocol based on polling for downlink data streaming and compare it to other protocols in [3]. Their key findings include that the minimization of

the idle listening period, as well as overhearing may become the main source for energy reduction in communication. In the area of wire-bound communication, a more complex approach based on PCI-Express is presented in [5]. The authors use appropriate line speeds, lane configurations and moreover different power modes for the physical layer.

In CMOS-based architectures, the power consumption is obtained according to $P = C_L \cdot \alpha \cdot f \cdot V_{dd}^2$, where we denote the capacitative load by C_L, the switching activity by α, the clock frequency by f and the supply voltage by V_{dd}. However, the most promising factor, the supply voltage, as well as the capacitative load are given and cannot be altered at will, as, for example, the operation of the MGTs depend on a certain guaranteed voltage level. The toggle activity and the clock rate, fortunately, can be influenced, which is readily exploited in FPGA designs. Techniques to reduce the switching activity of certain links, such as clock gating, have already been introduced over one and a half decades ago [1] and are meanwhile integrated in CAD tools to perform automatic clock-gating on a very fined-grained level [4]. Although this is a very effective technique for ASIC design, clock-gating is not so efficient in FPGAs due to the high static power dissipation, so that it may deliver only between 50 and 80 % of its ASIC counterpart [16]. Furthermore, techniques have been proposed to reduce the clock-spine placement and clock-gating in Xilinx Virtex-5 FPGAs, which may reduce dynamic power consumption of up to 28% [12]. In addition, Huda et al. consider gating and placement of clock nets on Xilinx Virtex FPGAs to lower the power requirements [6].

III. Background

A. Serial RapidIO

The RapidIO specification defines a layered architecture consisting of the logical, transport and physical layer. The logical layer is responsible for the implementation of the transaction concept, providing support for memory operations, atomic operations, unaligned memory transfers, globally shared memory and message passing. Routes for traversal of the nodes of the fabric are provided by the transport layer. The physical layer is specified for both, a parallel and a serial standard. The serial standard, also referred to as Serial RapidIO, can operate at up to 16 serial duplex links with a maximum line rate of 6.25 Gbaud. Furthermore, the physical layer is responsible for receiving and transmitting packets and for providing transmitter or receiver based flow control. The implementation assures packet delivery, since packets remain the responsibility of the transmitter until the receiver has acknowledged the acceptance. In case of insufficient resources at the receiver, the transmitter must retry the packet. After an implementation dependent amount of retries, the protocol defines the recovery from erroneous transmissions. To ensure that latency sensitive traffic is not delayed by larger packets, the maximum payload for SRIO packets is 256 byte at an overhead of about 20 bytes, depending on the transaction type. To conduct our studies, we have used the Xilinx SRIO IP core, which will be introduced next.

B. Xilinx SerialRapidIO FPGA IP Core

For use with their range of FPGAs, Xilinx offers two IP cores to implement an SRIO FPGA endpoint. The logical and transport layer, which comprise the logical layer core (LOGIO), are separated from the physical layer core (PHY). In this work, we have used the IP cores in version 5.6 which implements version 2.1 of RIO specification [13]. On top of the Xilinx SRIO IP core, we have implemented a power management unit (PMU), to selectively disable the SRIO endpoint. The individual components of the SRIO architecture are traversed by data in sequence, according to the direction of the transmission. For example, in case of an egress transmission, a packet must first traverse the logical layer, before it is being held in the buffer between the LOGIO and the PHY until the PHY was able to successfully transmit it to the next SRIO device. This is crucial for the implementation of the PMU, which includes clock gating of the IP Core components in the order of the traversal. Another part of the SRIO, that can be disabled in idle periods are the multi-gigabit transceivers (MGTs). On a Virtex-6 LXT, which was used for this study, the GTX MGTs [14] are used as serial transceivers, which are organized in clusters of four transceivers, sharing two differential reference clocks. Two analog supply powers are used, MGTAVCC and MGTAVTT. MGTAVCC is responsible for the internal analog circuits, including the phased locked loop (PLL) to synthesize a clock that matches the frequency of the clock that generates the incoming serial data stream, the transmitters and the receivers. MGTAVTT powers the termination circuits of the transmitters and the receivers. The GTX supports several power-down modes to facilitate the implementation of a generic power control. For both, the transmitter and the receiver lane, it is possible to put the actual transceiver as well as the PLL into a low power mode. However, it is only supported to power down the receiver in conjunction with the transmitter. At start-up or after a power-down of the GTX transceivers, the serial interfaces and the physical layer loose synchronization and must follow a protocol defined initialization routine to resynchronize. Important indicators are the link status, as well as the receive and transmit port of the module. The endpoint is functional, if all three indicators are asserted.

IV. Power Management Strategies for SRIO

A. Motivation

In Figure 1, we depict the communication channel utilization over time in a fixed bandwidth communication scenario, where the data rate requirements are lower than the actual available data rate. The channel alternates between active

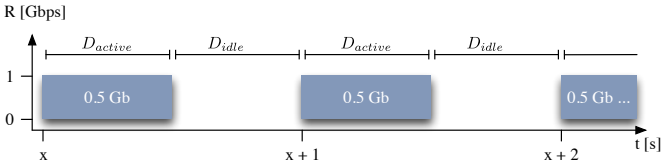

Figure 1. Channel activity at a transmission rate of 1 Gbps at an available data rate of 1 Gbps and 0.5 Gbit of data.

periods (D_{active}) of data transmission and idle periods (D_{idle}), where the network adapter does not transmit any user data. Although there is no payload being transmitted during the idle periods, the link is highly active to keep the communication partners synchronized. Hence, such idle periods are prime candidates for optimization of power consumption in serial interconnects. If the link is deactivated during idle periods and reactivated for transmission or reception of new data, the link is not ready right away, but must resynchronize before data can be transmitted. This link synchronization is referred to as the link training delay (D_{lt}) and, unfortunately, may take up between 100 and 250 ns, depending on the line rate of the interface.

Several possibilities exist for power-management of idle-periods. A straight-forward means of reducing the power consumption of the SRIO endpoint is to increase link utilization by reducing the amount of lanes and the maximum lane rate, as depicted in Figure 2. Such an approach is very

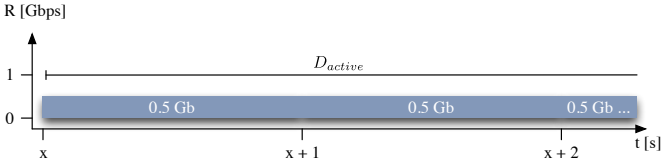

Figure 2. Channel activity at a transmission rate of 0.5 Gbps at an available data rate of 2 Gbps and 0.5 Gbit of data.

effective, since does not suffer from idle periods in which the link is only active for synchronization. However, it imposes a hard limit on the available data rate, so that the link is suitable only for evenly distributed traffic and cannot handle bursty traffic sources.

In contrast a rather complex strategy is to keep the transceiver initially in the disabled state and only activate it in case of outgoing transmissions or link sensing for incoming communication. For egress data, the situation is very convenient, since the transceiver can be activated as soon as data is available for transmission. Data can then be transferred and the transceiver can be disabled afterwards, which is shown in Figure 3. Such an implementation is especially useful for signal processing applications, where data suffers a high latency due to processing but is delivered at a high throughput. Unfortunately, there exist several problems with such an approach. In Figure 4, we consider the link training delay after enabling the transceiver, which causes additional latency and reduces the link capacity. Powering down the transceiver during idle periods can reduce the power consumption, however, incoming transmissions cannot be received. If the arrival times of ingress transmissions are not known a priori, the transceiver must be periodically enabled to sense the link for prospective incoming data. Deciding on when to wake up and how long to listen is a complex optimization problem, since not only the power consumption of the transceiver but also that of excess memory necessary to store egress data at the sender until the link partner becomes available must be taken into account.

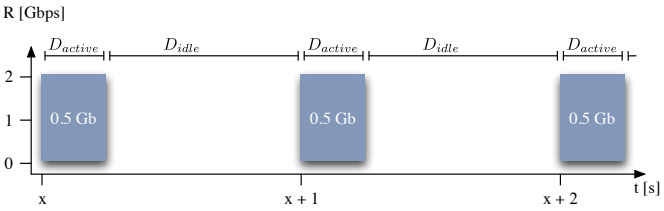

Figure 3. Channel activity at a transmission rate of 2 Gbps at an available data rate of 2 Gbps and 0.5 Gbit of data.

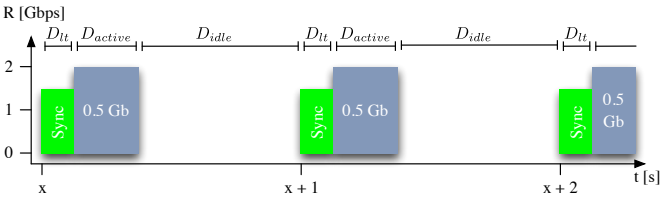

Figure 4. Channel activity at a transmission rate of 2 Gbps at an available data rate of 2 Gbps and 0.5 Gbit of data. Addtionally, the link training delay D_{lt} is depicted.

Embedded systems applications that make use of complex serial interconnects often posses the advantage that data streams are known a priori and can therefore be scheduled. Especially if the system must fulfill real-time constraints, the exact scheduling of data streams is a common practice. In this work we propose a novel budget-based power-management for a priori known data streams over SRIO communication links. The advantage of such system behavior over the previously introduced power management strategies is that the transceivers can be activated in fixed intervals, which minimizes idle as well as link sensing periods and can therefore reach a very high level of energy efficiency.

B. Methodology

The process of activating an SRIO communication controller can be subdivided into 5 steps, as depicted in Figure 5.

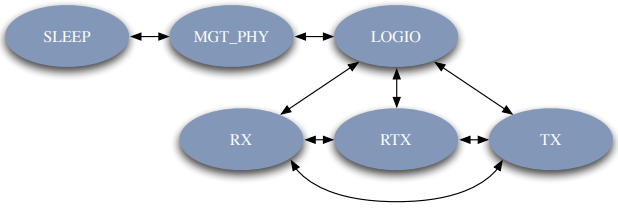

Figure 5. Transceiver hardware enable states and transitions for SRIO communication events.

These are:

- SLEEP – The transceiver lanes as well as the PLL are powered down, the IP core and the user application are clock-gated.
- MGT_PHY – The transceiver are powered up and the clock to the physical layer is enabled.
- LOGIO – The logical layer and is enabled in addition to the MGT and the PHY.
- TX/RX – Either the transmitter or the receiver logic is activated in addition to the MGT, the PHY and the LOGIO.
- TRX – The complete transceiver logic is activated.

The initial state is SLEEP, in which the complete transceiver hardware is disabled. From here, to wake up the transceiver, the next state is MGT_PHY, where the MGTs and the PHY are enabled to start link training. The transceiver will remain in this state for the duration of D_{lt} until the link is synchronized, after which the logical layer can be enabled in state LOGIO. Depending on whether only the transmitter, the receiver, or both components will be needed, the transceiver is then put into one of the corresponding states TX, RX, or TRX, respectively. The logical layer is enabled in all of theses states, however, the logic for transmitting and receiving can be selectively disabled, according to the task to fulfill.
Each of the states is associated with a power consumption P_x, where x denotes the current state of the hardware:

P_S	Power in state SLEEP
P_M	Power in state MGT_PHY
P_L	Power in state LOGIO
P_T	Power in state TX
P_R	Power in state RX
P_{TR}	Power in state TRX

We define the power level P_S during the state SLEEP as the *basic power reference level*. The change to other states by activating the communication controller elevates the power consumption. We denote $P'_x = P_x - P_S$ as the increase in power consumption in state x compared to the basic level of power consumption. For example, $P'_M = P_M - P_S$ expresses the increase in power consumption, if the MGTs and the PHY are activated. In contrast, if no power management is implemented, the communication controller is constantly in the state TRX and consumes an amount of energy equal to $P'_{TR} = P_{TR} - P_S$. We can achieve an improvement in power consumption if the mean value of power consumption over a certain period of time t, consisting of the sum of the power levels in the different states is smaller than the power consumption P'_{TR} over the same period of time. To elaborate on this, we examine the power management strategies presented in the previous section.
Consider the case when there is no a priori knowledge of the duration of idle periods and data may arrive or must be transmitted at arbitrary points in time. We will first analyze the transmission process, which is depicted in Figure 6. Packets traverse several stages of the endpoint

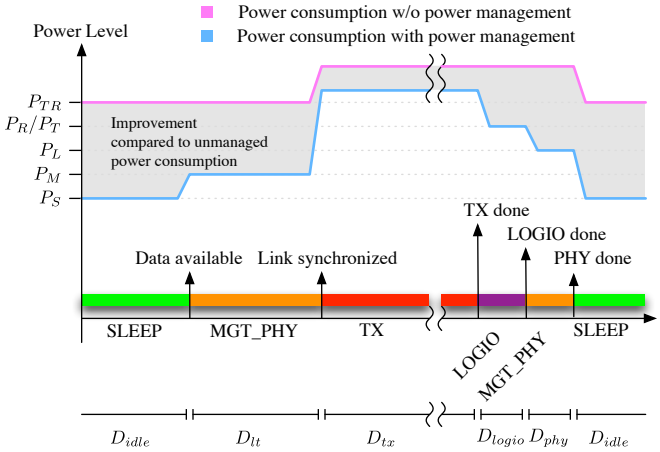

Figure 6. Transmission process and power levels of managed and unmanaged SRIO endpoints. Higher power consumption is due to increased toggle activity during TX state.

controller, each of which may selectively be enabled or disabled. The first step is to enable the MGTs and the PHY. Until the link is synchronized, the remaining hardware can stay in the disabled state. After synchronization, both, the logical layer and the transmit logic are activated in a single step to start the transmission. When the last data packet was processed by the transmit component, it still has to traverse the logical layer and the PHY, so we sequentially disable the components, once activity has ceased. During transmission of the data, we experience an overall higher power consumption, which is due to the higher toggle rate in the individual components. We also depict the power consumption of an unmanaged endpoint controller in Figure 6 for comparison. Here, the power consumption is also elevated during the actual transmission. We can now evaluate the energy savings of this approach as follows: The energy $E_{unmanaged}$, consumed by the unmanaged endpoint controller is

$$E_{unmanaged} = P_{TR} \cdot (D_{idle} + D_{tl}) + P_{TX} \cdot D_{tx}, \qquad (1)$$

whereas, the energy $E_{managed}$ consumed by the managed endpoint is

$$\begin{aligned} E_{managed} \quad = \quad & P_S \cdot D_{idle} + P_M \cdot D_{lt} + P_T \cdot D_{tx} \\ & + P_L \cdot D_{logio} + PM \cdot D_{phy}. \end{aligned}$$

The difference between both is displayed in Figure 6. Obviously, the longer the idle periods can be kept, the more effectively the energy can be reduced.
An advantage of this approach is that it only increases the delay by D_{tl} of a data stream, but does not decrease the maximum available line rate. The same observation holds for receiving packets, where the transceiver must be activated periodically in intervals D_{int} to probe the link for incoming transactions, as it is illustrated in Figure 7. However, this is

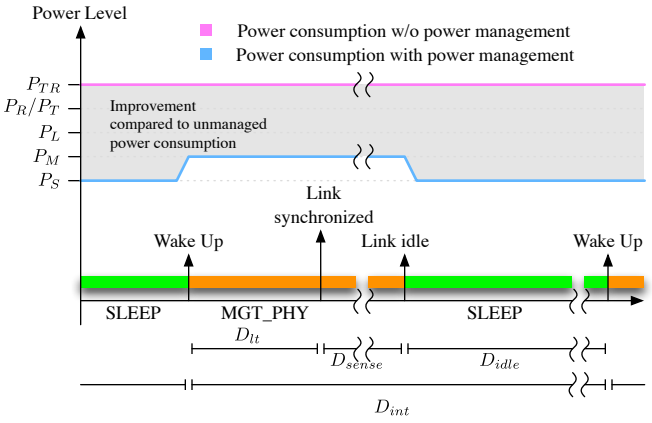

Figure 7. Periodic link sensing process and power levels of managed and unmanaged SRIO endpoints.

only the case if the transceiver is deactivated. During active transmission periods, no extra hardware must be activated for link sensing and receiving. The worst case scenario happens if there are no active periods due to transmissions and new ingress data becomes available right after the last sensing period D_{sense} has ended. The periodical interval comprises the idle time D_{idle}, the link training time D_{lt} and the sensing time D_{sense}. The maximum delay an ingress

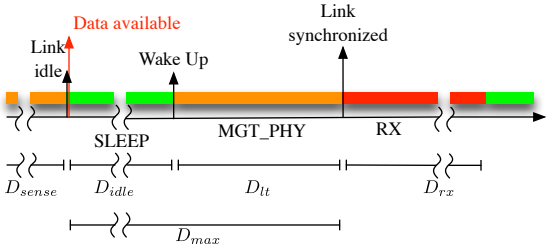

Figure 8. Maximum delay D_{max} at periodic link sensing scheme.

packet can experience is depicted in Figure 8 and therefore $D_{max} = D_{idle} + D_{lt}$. The energy that can potentially be saved E_{save} in comparison to an unmanaged controller during such a cycle in the best case is the difference between the power consumed in state TRX during the time D_{max} and the power consumed in state SLEEP during D_{idle} and in state MGT_PHY during D_{lt}:

$$E_{save} = P_{TR} \cdot D_{max} - (P_S \cdot D_{idle} + P_M \cdot D_{lt}) \qquad (2)$$

During idle periods without any incoming or outgoing transmissions, the potential saving evaluates to:

$$E_{save} = P_{TR} \cdot D_{int} - (P_S \cdot D_{idle} + P_M \cdot (D_{lt} + D_{sense})) \qquad (3)$$

Furthermore, the sender must provide enough memory to buffer egress data while waiting for a sleeping link partner to wake up. Depending on the application, the worst case requires a memory of size $K = D_{max} \cdot R$, where R is equal to the line rate. Providing such a memory will cause additional power consumption, which must be subtracted from the potential savings. Of course, if there are no transmissions, the memory can be disabled by clock gating, as well.

In case of a priori knowledge of the amount of data to transmit between link partners within a given time interval, we can circumvent the necessity to periodically activate the receiver for link sensing. According to a predefined schedule, the transceivers of both link partners are enabled, so that a common activity period D_{active} within an interval D_{budget} can be used to transmit and receive data between idle intervals D_{idle}. The duration of such an activity period depends on a specific data-budget between the link-partners that is defined at design time, but may also be adapted during run time. The amount of data to transmit and receive may not be equal. Thus, the active time equals to the already known delays for link training as well as IP core traversal times and maximum of the transmit time D_{tx} and the receive time D_{rx}, $D_{active} = D_{lt} + max(D_{tx}, D_{rx}) + D_{logio} + D_{phi}$. For example, in case of more data to be transmitted than received, the consumed energy during D_{active} evaluates to

$$\begin{aligned} E_{active} \quad = \quad & P_M \cdot D_{lt} + P_{TRX} \cdot D_{rx} + P_{TX} \cdot (D_{tx} - D_{rx}) \\ & + P_L \cdot D_{logio} + PM \cdot D_{phy}. \end{aligned}$$

Apart from active periods, the controller can remain in the low-power sleep state during the idle period D_{idle}. An illustration of this case is depicted in Figure 9. The

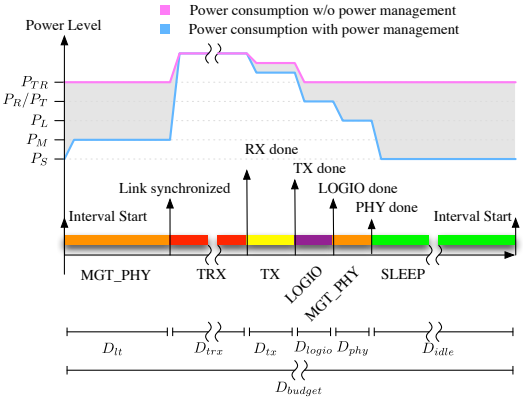

Figure 9. Transmission process and power levels of budget-based power management and unmanaged SRIO endpoints. Higher power consumption is due to the increased toggle activity during TRX and TX state.

energy saved if the endpoint is managed by the budget-based protocol evaluates to

$$E_{save} = P_{TR} \cdot D_{budget} - (E_{active} + P_S \cdot D_{idle}). \qquad (4)$$

V. Experimental Results

In order to evaluate the proposed power management strategies, we have analyzed the PMU for the Xilinx SRIO IP Core in version 5.6 using the Xilinx Virtex-6 LXT 240 FPGA on the ML605 evaluation board, which supports single lane SRIO architectures. Since the ML605 does not include an oscillator to drive the system clock, we have used an ML505 board as external source at 125 MHz. At 125 MHz we can generate SRIO endpoints at line rates of 1.25, 2.5, 3.125 and 5 GBaud. Before measuring the power consumption of the proposed power optimization strategy

on actual hardware, we have performed an analysis by simulation.

A. Simulation-based Analysis

We have created two different versions of the SRIO endpoint controller, one with the PMU and without power management. Without the PMU, we have synthesized the endpoint for two different optimization goals, minimum period (Speed) and minimum power (Power). The PMU-based endpoint design was optimized for minimum period (PMU). To estimate the improvement possible due to power management, the designs were analyzed using the Xilinx XPower Analyzer tool [15]. In comparison to the measurement on actual hardware, XPower offers the advantage, that it can exactly determine the power consumption of each individual component of the FPGA. The results of the XPower Analysis are listed in Table I and were made under *commercial* settings for the temperature grade, as well as *typical* process settings. As expected, the largest part of the power consumption is due to leakage, which is an FPGA inherent problem. To visualize the remaining proportions, we have omitted the leakage part in Figure 10. A very large

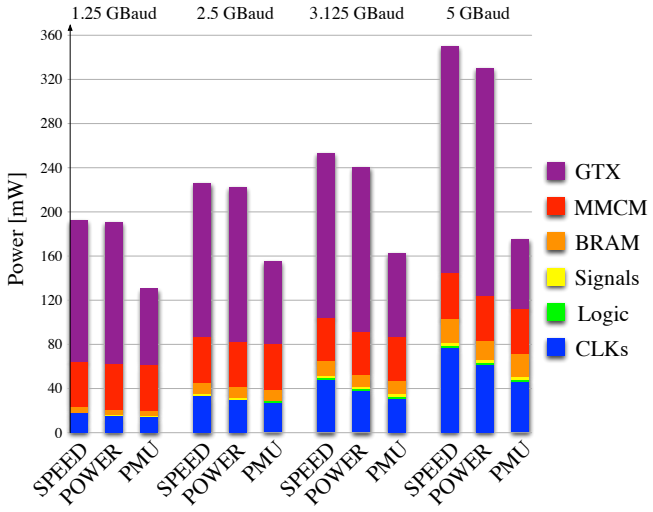

Figure 10. Power requirements of individual FPGA components of the SRIO endpoint designs for all possible line rates.

proportion of the remaining power consumption is due to the GTX transceivers, and deactivation of the transceivers in the PMU design can reduce the required energy by up to 75 %. Moreover, it can be seen that the PMU designs can also reduce the energy required by the clock tree due to clock-gating, which outperforms the automatic approach of up to 35 % for the 5 GBaud design. Moreover, we observe that the PMU requires extra logic, especially for the faster designs, which will also be noticeable in the hardware measurements. The downside of the XPower analysis is that the results do not very well reflect the dynamic behavior of the interconnect at different data budgets, which is why we cannot abstain from performing power measurements on the actual hardware.

B. Hardware-based Measurements

The power supply on the Virtex-6 FPGA is controlled by a Texas Instruments (TI) UCD9240 controller. Measuring of the power consumption of the FPGA and the MGTs can be easily accomplished on the ML605 using the PMBus interface to the TI controller and the TI Fusion Digital Power Designer software package [10]. For this study it is interesting to measure the power consumption on the VCCINT power rail, which drives the internal components of the FPGA, as well as the MGTAVCC and MGTAVTT power rails, as described in Section III-B. The SRIO endpoint was implemented with and without the PMU. The design without the PMU was used to generate a bit file with minimum period as design goal (Speed) and furthermore implemented with the Xilinx ISE internal power reduction option enabled (Power). The design with the PMU was optimized for minimum period (PMU). All three designs were implemented for all possible line rates at a 125 MHz system clock speed. Due to space restrictions, we only list the measurement results for 5 GBaud in Table II.

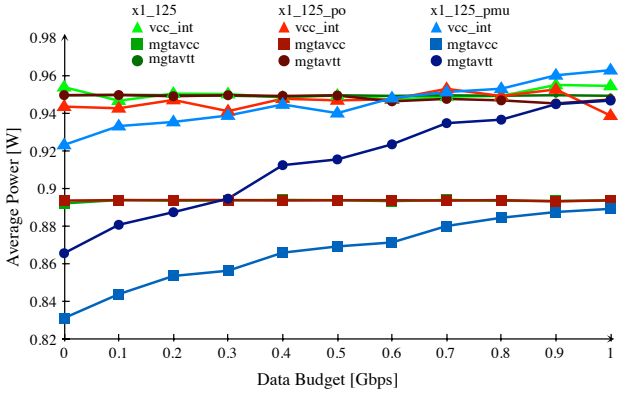

Figure 11. Average power consumption of SRIO endpoint designs for 125 MHz system clock and 1.25 GBaud line rate.

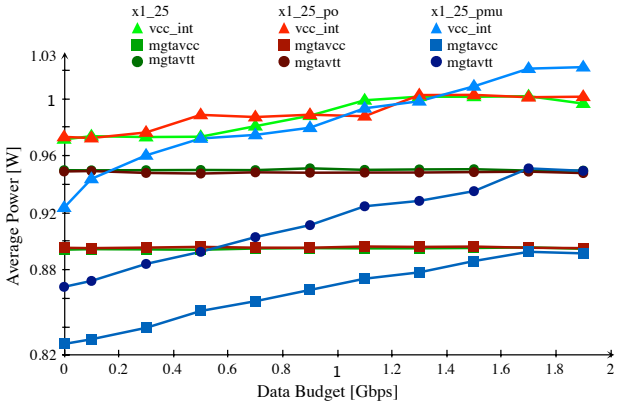

Figure 12. Average power consumption of the SRIO endpoint for 125 MHz system clock and 2.5 GBaud line rate.

A plot of the measurements for the 1.25 GBaud implementation is shown in Figure 11. The measurements were conducted up to the maximum data budget of 1 Gbps. To keep the rest of the measurements comparable, individual

Line Rate	1.25 GBaud			2.5 GBaud			3.125 GBaud			5.0 GBaud		
Design	Speed	Power	PMU	Speed	Power	PMU	Speed	Power	PMU	Speed	Power	PMU
Resources (Slices)												
Slices	6352	6415	6412	6341	6434	6340	6144	6147	6270	6018	6107	6310
Power [W] - XPower Analysis												
CLKs	0.017	0.015	0.014	0.033	0.030	0.027	0.048	0.038	0.031	0.077	0.071	0.046
Logic	0.000	0.000	0.000	0.000	0.000	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Signals	0.001	0.001	0.001	0.001	0.001	0.002	0.002	0.002	0.002	0.003	0.003	0.003
BRAM	0.005	0.005	0.005	0.011	0.010	0.010	0.014	0.011	0.013	0.022	0.018	0.021
MMCM	0.041	0.041	0.041	0.041	0.041	0.041	0.039	0.039	0.039	0.041	0.041	0.041
GTX	0.128	0.128	0.072	0.140	0.140	0.075	0.157	0.157	0.076	0.206	0.206	0.071
Leakage	3.360	3.36	3.358	3.361	3.361	3.358	3.362	3.361	3.359	3.365	3.364	3.359
Total	3.552	3.550	3.492	3.588	3.584	3.514	3.622	3.609	3.521	3.715	3.705	3.542

Table I
RESOURCE REQUIREMENTS AND POWER ANALYSIS RESULTS FOR THE SRIO ENDPOINT ENHANCED BY POWER MANAGEMENT(PMU) AND UNMANAGED ENDPOINT DESIGNS OPTIMIZED FOR MINIMUM PERIOD (SPEED), AS WELL AS MINIMUM POWER CONSUMPTION(POWER), LISTED ACCORDING TO LINE RATE. DESIGN NAMES REFLECT OPTIMIZATION GOAL AND PMU USAGE.

Design	Speed			Power			PMU		
Power Rail	VCCINT	MGTAVCC	MGTAVTT	VCCINT	MGTAVCC	MGTAVTT	VCCINT	MGTAVCC	MGTAVTT
Data Budget	**Power [W]**								
0.0	1.0313	0.9421	0.9513	1.0274	0.9421	0.9513	0.9697	0.8317	0.8616
0.1	1.0350	0.9415	0.9517	1.0274	0.9415	0.9517	0.9777	0.8364	0.8681
0.3	1.0364	0.9419	0.9516	1.0289	0.9419	0.9516	0.9779	0.8446	0.8696
0.5	1.0439	0.9418	0.9515	1.0381	0.9418	0.9515	0.9997	0.8445	0.8744
0.7	1.0416	0.9418	0.9507	1.0370	0.9418	0.9507	0.9954	0.8542	0.8779
0.9	1.0472	0.9419	0.9515	1.0411	0.9419	0.9515	1.0105	0.8562	0.8836
1.1	1.0465	0.9414	0.9517	1.0470	0.9414	0.9517	1.0090	0.8612	0.8876
1.3	1.0508	0.9420	0.9512	1.0496	0.9420	0.9512	1.0237	0.8682	0.8903
1.5	1.0560	0.9417	0.9509	1.0497	0.9417	0.9509	1.0293	0.8746	0.8961
1.7	1.0574	0.9427	0.9511	1.0567	0.9427	0.9511	1.0448	0.8723	0.8940
1.9	1.0589	0.9416	0.9510	1.0573	0.9416	0.9510	1.0506	0.8850	0.8978

Table II
POWER RAIL MEASUREMENTS OF THE 5 GBAUD DESIGN WITH AND WITHOUT PMU. THE DESIGN WITHOUT PMU WAS OPTIMIZED FOR MINIMUM PERIOD (SPEED) AND POWER REDUCTION (POWER). THE PMU DESIGN (PMU) WAS OPTIMIZED FOR MINIMUM PERIOD.

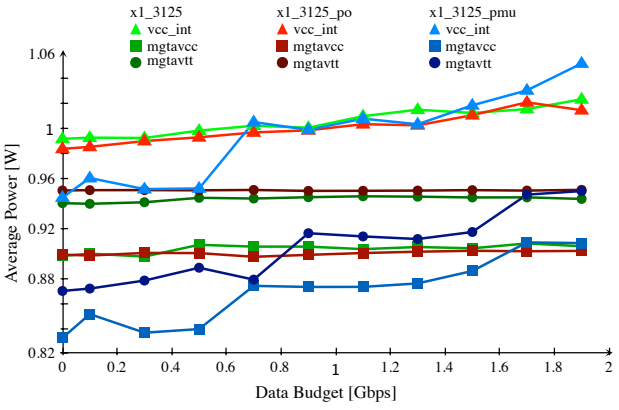

Figure 13. Average power consumption of the SRIO endpoint for 125 MHz system clock and 3.125 GBaud line rate.

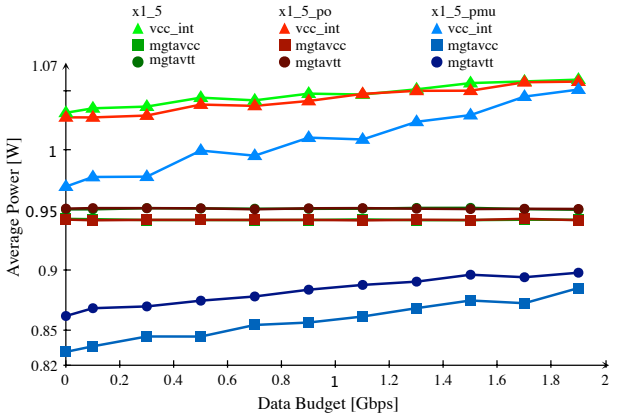

Figure 14. Average power consumption of the SRIO endpoint for 125 MHz system clock and 5 GBaud line rate.

designs in Figures 12, 13, and 14 were measured up to 1.9 Gbps, although, the 3.125 and 5 GBaud versions are capable of transmitting data at higher rates. The graphs show that disabling the GTX transceivers, affecting MGTAVCC and MGTAVTT, during idle periods is very effective, however, as the data budget approaches the maximum supported line rate, the measurements converge with those without the power management implemented. FPGA-internal clock-gating of the communication controller, which only affects VCCINT, is only marginally effective. On the contrary, once the data budget approaches the maximum achievable line rate, the power consumption is actually higher due to the extra logic of the PMU. Furthermore, the results show that although very fine-grained power reduction techniques prove to be effective, manual optimizations, such as disabling the GTX transceivers during idle periods, can outperform such techniques. The best results can of course be observed for idle periods, in which we can lower the combined

power consumption of the GTX transceiver on power rails MGTAVCC and MGTAVTT by up to 200 mW. As specified earlier, we take the values at idle operation in the lowest power state as the basic reference value. For the 5 GBaud design, which represents the best case, we can achieve a reduction of the power consumption for the MGTs by 77% on average, and for the internal FPGA design by 58 %. For the 1.25 GBaud design, we can still achieve a reduction by 44 % on average for the GTX transceivers and by 18 % for VCCINT. Another observation is the comparison between lowering the maximum line rate for an underused link to avoid idle listening and raising the link rate to the maximum possible speed and use power management to increase idle periods, in which the controller can be deactivated by a PMU. A comparison for the measurements up to a data budget of 0.9 Gbps is depicted in Figure 15. For communication controllers not involving power management, reducing the line rate is very effective. However, using power management instead of lowering the maximum line rate can reduce the power consumption to a much higher degree.

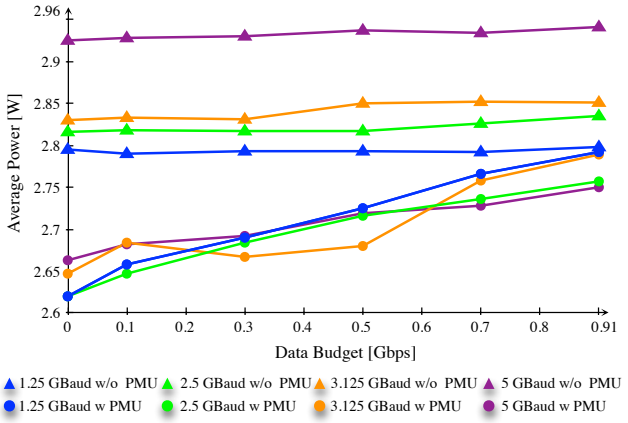

Figure 15. Comparison of the average power consumption of SRIO endpoints for 125 MHz system clock and all possible line rates.

VI. Conclusion

We have proposed a novel data budget-based approach to dynamically control the power consumption of SRIO endpoint controllers in FPGAs. The key concept of the approach is to not only perform clock-gating on the FPGA-internal components of the communication controller, but to disable the MGT transceivers during idle periods. The clock synchronization, inherent to serial interfaces, enables us to omit the often needed periodic link sensing, and only enable the controller according to a predefined schedule to transmit the allocated data budget for the budget interval. Following this approach we are able to reduce the dynamic power consumption by up to 77 % on average. Moreover, we have shown that lowering the line rate on underused links is an effective technique to reduce the power consumption, however, transmitting the data at maximum speed to maximize idle periods in which the controller can be deactivated, can reduce the power consumption even more.

References

[1] L. Benini, P. Siegel, and G. D. Micheli. Saving power by synthesizing gated clocks for sequential circuits. *IEEE Design and Test of Computers*, 11:32–41, 1994.

[2] D. Bueno, C. Conger, A. D. George, I. Troxel, and A. Leko. RapidIO for radar processing in advanced space systems. *ACM Transactions on Embedded Computing Systems*, 7(1):1–38, 2007.

[3] A. El-Hoiydi and J.-D. Decotignie. Low power downlink mac protocols for infrastructure wireless sensor networks. *Mobile Networks and Applications*, 10:675–690, October 2005.

[4] F. Rivoallon, Xilinx Inc. Reducing Switching Power with Intelligent Clock Gating, March 2011. *http://www.xilinx.com/support/documentation/white_papers /wp370_Intelligent_Clock_Gating.pdf*, accessed December 22nd, 2011.

[5] T. Hanawa, T. Boku, S. Miura, M. Sato, and K. Arimoto. Power-aware, dependable, and high-performance communication link using PCI Express: PEARL. In *IEEE International Conference on Cluster Computing Workshops and Posters (CLUSTER WORKSHOPS), 2010*, pages 1–4, September 2010.

[6] S. Huda, M. Mallick, and J. H. Anderson. Clock gating architectures for FPGA power reduction. In *IEEE 19th International Conference on Field Programmable Logic and Applications*, pages 112–118, September 2009.

[7] T.-J. Lee, M.-S. Han, J.-W. Han, S.-J. Shin, K.-M. Kim, S.-Y. Chun, and K. Son. A study on the low power communication for underwater sensor network. In *IFIP 9th International Conference on Embedded and Ubiquitous Computing*, pages 420–423, October 2011.

[8] RapidIO Trade Association. RapidIO specification, 2009. *http://www.rapidio.org/spec/current*, accessed January 7, 2010.

[9] C. Sulzbachner, J. Kogler, and W. Kubinger. An embedded high performance data acquisition and pre-processing interface for asynchronous event-based silicon retina data. In *2010 IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications*, pages 313–318, July 2010.

[10] Texas Instruments Inc. Using the UCD92xx Digital Point-of-Load Controller, April 2011. *http://www.ti.com/lit/ug/sluu490/sluu490.pdf*, accessed December 22nd, 2011.

[11] H. Unterassinger, M. Dielacher, M. Flatscher, S. Gruber, G. Kowalczyk, J. Prainsack, T. Herndl, J. Schweighofer, and W. Pribyl. A power management unit for ultra-low power wireless sensor networks. In *IEEE AFRICON*, pages 1–6, September 2011.

[12] Q. Wang, S. Gupta, and J. H. Anderson. Clock power reduction for virtex-5 fpgas. In *Seventeenth ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 13–22, February 2009.

[13] Xilinx Inc. LogiCORE IP Serial RapidIO v5.6 User Guide, July 2011. *http://www.xilinx.com/support/documentation/ ip_documentation/srio_ug503.pdf*, accessed December 22nd, 2011.

[14] Xilinx Inc. Virtex-6 Family Overview, March 2011. *http://www.xilinx.com/support/documentation/ user_guides/ug366.pdf*, accessed December 22nd, 2011.

[15] Xilinx Inc. XPower Estimator User Guide, October 2011. *http://www.xilinx.com/support/documentation/ user_guides/ug440.pdf*, accessed December 22nd, 2011.

[16] Y. Zhang, J. Roivainen, and A. Mämmelä. Clock-gating in fpgas: A novel and comparative evaluation. *Euromicro Symposium on Digital Systems Design*, pages 584–590, 2006.