Mesochronous clocking and communication in on-chip networks

Daniel Wiklund

Dept. of Electrical Engineering
Linköping University
S-581 83 Linköping, Sweden
danwi@isy.liu.se

ABSTRACT

On-chip networks are becoming a popular research topic, both in industry and universities. Many researchers assume a fully synchronous or globally asynchronous, locally synchronous model of operation for the network. We have previously proposed the use of mesochronous communication within the network as a simple and robust way to get around the problems of fully synchronous operation.

The mesochronous communication technique is simple both in understanding and implementation and allows for significant freedom in design, placement, and inter-subsystem delays.

This paper presents the concept of mesochronous communication and clocking. It further introduces a possible implementation for the network component interconnections using mesochronous communication with an integrated clock distribution mechanism. This implementation is then analyzed from a functional perspective.

1. INTRODUCTION

During the last few years there has been a significant increase in popularity of research on on-chip networks. With many architectural proposals there are also the need to understand the method of communication within the network on lower levels. A common assumption is that the network is run either globally synchronously over the chip or that it is run completely asynchronously [1, 2].

In principle there are three methods to use for synchronization of a system. The most commonly used today is the synchronous system where a global clock is distributed over the system with low skew. This clock is then used to time all the events and transactions in the system.

Another method that is popular in research and extreme low power products is to run the system completely asynchronous. Completely asynchronous systems need to use handshaking or special timing circuitry for both computations and communications in order to keep synchronization within the system.

The third method is to use blocks that are synchronous but communicate asynchronously, better known as the globally asynchronous, locally synchronous (GALS) methodology. With the current increase in the number of different clock domains used on a single chip this is a very promising overall technique to use for IP block integration.

If a subsystem is distributed over a large area within a chip there is also the possibility to use a method that is somewhat in between the previously mentioned methods. Here there is a common clock that is distributed to the system without concern about the phase difference in different parts of the system. Parts of the system may run as synchronous subsystems and can be designed using the accepted standard methods of today.

The communication between the subsystems is then done in much the same way as for a fully synchronous system. The primary difference is that the incoming data to a subsystem has to be aligned to the local clock phase. Since the clock rate is the same for the entire system this can be performed with a simple retiming circuitry.

We have previously proposed an on-chip network [3] and suggested it should use a mesochronous communication and clocking scheme for internal communication [4]. This paper presents a possible solution for the mesochronous communication and clock distribution between the network components.

Daniel Wiklund is a Ph.D. student at the division of Computer Engineering, Dept of Electrical Engineering, Linköping University. This work is funded by the STRINGENT electronics center. This work is based on an idea by Prof. Christer Svensson, Electronic Devices, Dept of Electrical Engineering, Linköping University.

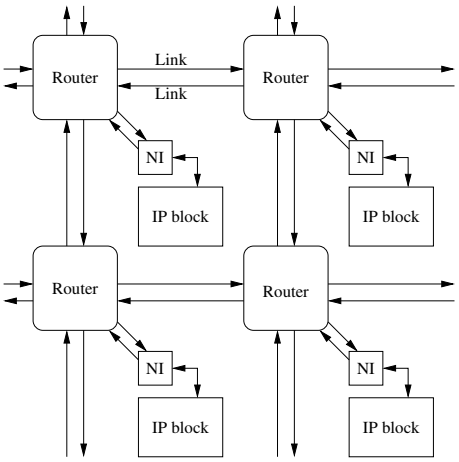

Fig. 1. Section of a mesh style on-chip network.

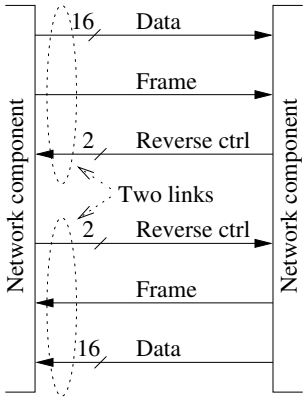

Fig. 2. Basic signals for one undirectional internal network link.

2. CLOCK DISTRIBUTION IN ON-CHIP NETWORKS

Assuming that the components of the on-chip network is implemented using a method that is not fully asynchronous, a clock is needed in each router and network interface. If the system is to use a globally synchronous model the distributed clock has to have low skew between the connected neighbors in the network but it is possible to allow significantly more clock skew across longer distances in the network. Since there are only fairly local links in the network, see figure 1, this is a feasible solution for networks with low clock rate (up to a few hundred MHz).

A fully asynchronous solution of course do not need a clock distribution at all. There is also the possibility to use a mixed system based on the globally asynchronous and locally synchronous (GALS) scheme, i.e. the routers and network interfaces are run synchronous internally and asynchronous externally. This method requires the distribution of a clock to every network component but there is no specific demands on clock skew or even difference in clock frequency between the components.

3. MESOCHRONOUS CLOCKING

Our proposed solution is to use mesochronous clocking, i.e. using the same frequency but with unknown phase. There are two basic methods for communication between subsystems when using mesochronous clocking. The first is to recognize and handle the situation when metastable conditions may occur [5]. The detection of this kind of possibly metastable condition gives rise to a 180 degree shift in the local clock phase used for synchronizing the incoming data. This method has the advantage of not needing any extra wiring for the synchronization but is bad since the clock phase difference may be very close to where the metastable conditions will occur.

The second approach is to keep as far away from the metastable case by analyzing the incoming data phase or by providing a timing signal along with the data signals that can be used to estimate the phase difference.

One of the most basic requirements for unconstrained mesochronous clocking and communication is that there must be no special timing requirements on the communication. This means that things like low level flow control for the communication may be impossible to implement. The subsystems then have to be designed in such a way that this low level flow control is not necessary.

We propose to use the second approach where every network component is built using the well-known methodology for fully synchronous systems but with special interface cells for synchronization at each input.

4. A POSSIBLE IMPLEMENTATION OF MESOCHRONOUS CLOCKING

In the SoCBUS project every (unidirectional) link from a network component to another consists of a data bus that is typically 16 bits wide, one forward framing signal, and two reverse control signals, as shown in figure 2. To enable the use of mesochronous clocking this link is then supported by a single strobe signal going in the forward direction. This strobe carries a sig-

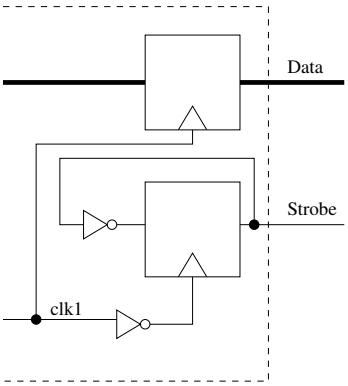

Fig. 3. Sender for strobed mesochronous clocking.

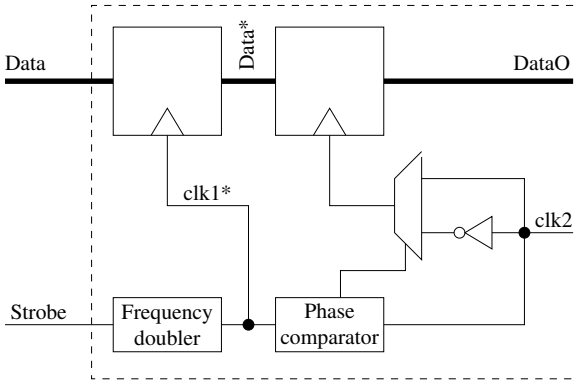

Fig. 4. Receiver for strobed mesochronous clocking.

nal that is simply the clock frequency divided by two. This signal will have the same characteristics when it comes to spreading and distortion as the other signals transferred on the data and control wires. This strobe can then comfortably be used to time the transfers. As can be seen in figure 2, there are two signals, the reverse control, going in the opposite direction. Whenever there are two opposing unidirectional links between the components as in the figure there is no extra cost for these reverse control signals. They can simply be timed to the opposing forward signals. This situation is probably the by far most common but if there is only one unidirectional link between two components the reverse control signals have to be accompanied by their own strobe signal.

As long as the data signal paths and the strobe signal path is fairly well balanced in each link the strobe signal will convey all necessary information for the synchronization to the local clock phase.

Figure 3 shows the sender side design for a mesochronous link. The sender creates the strobe signal associated with the link. The receiver is somewhat more complex, see figure 4. A frequency doubler is used to reconstruct the clock signal from the incoming strobe. This clock signal is a replica of the sender clock which is skewed to match the incoming data and is used for the latching of incoming data into the first stage flip-flops. The second flip-flop stage is used to time the data to the local clock phase. A phase detector and phase comparator is used to select whether the second stage will use the normal phase or the 180 degree shifted version of the local clock.

Without an extra flip-flop stage to synchronize the incoming data to the in-phase clock the signals from this second stage may have only one half clock cycle until the next rising edge in the receiving network component. This half period can be used as a timing

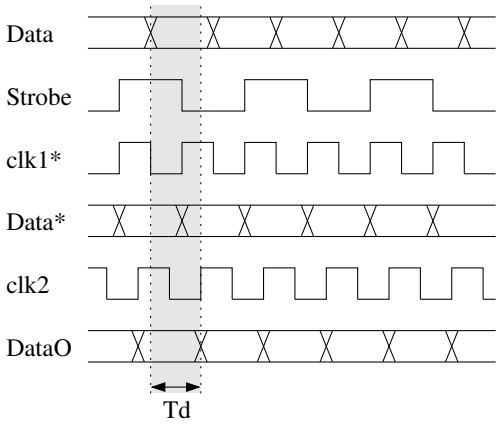

Fig. 5. Receiver waveform for mesochronous transmission. T_d denotes the receiver synchronization delay.

constraint for the first stage in a pipelined implementation of the network component. If this is not acceptable an additional flip-flop stage may be introduced to synchronize the signals fully to the local clock phase.

Since the strobe signal is transmitted along with the communication this can also be used to distribute the clock to the network components. This is achieved by using the doubled frequency from one of the input links as the network component clock. Thus there is no need for an additional clock distribution network and the specific link interface where the clock is taken do not need any more stages for the synchronization than the first one.

5. MESOCHRONOUS VS. ASYNCHRONOUS COMMUNICATION

The simplest way of designing the network components is to use a fully synchronous design methodology for the on-chip network. This is possible for the time being but will eventually (within a few years) become impossible for high end chips. When this time is reached there is basically two ways to go for the communication internal to the network, either to partly/fully asynchronous or a mesochronous solution.

Asynchronous solutions have some advantages over mesochronous solutions since the handshaking will directly allow for link-level flow control that is generally needed for networks using buffering within the network components. Another advantage is that a good implementation of the asynchronous interfaces will provide an extremely reliable system.

The mesochronous scheme in turn has some advantages over the asynchronous solution. A mesochronous system can, from a high perspective, be viewed as a fully synchronous system with pipeline delays on the interconnect links. The throughput on every link can also be higher since there is no need for handshaking of every transfer. Thus the wires may carry several sequential waves of data at the same time. The strobe signal used for synchronization can be used to distribute the clock as well as conveying timing information. Also the components for implementing the mesochronous communication is simple, compact, and robust.

In an on-chip network the mesochronous communication must be complemented by asynchronous bridging between the network clock domain and the IP block clock domains. This asynchronous communication is then done within the network interfaces. Thus the complete on-chip network system is an implementation of a highly flexible GALS style interface structure.

6. CONCLUSIONS

This paper shows the applicability of mesochronous communication and clocking to on-chip networks. A possible implementation for synchronization and retiming is presented together with a basic comparison of the mesochronous and asynchronous styles of communication.

7. REFERENCES

[1] K. Goossens, J. van Meerbergen, A. Peeters, and P. Wielage, "Networks on silicon: Combining best-effort and guaranteed services," in *Proceedings of the Design and Test in Europe conference (DATE)*, 2002.

[2] Sune F. Nielsen and Jens Sparsø, "Analysis of low-power SoC interconnection networks," in *Proc of the Norchip conference*, 2001.

[3] Daniel Wiklund and Dake Liu, "Switched interconnect for system-on-a-chip designs," in *Proc of the IP2000 Europe conference*, 2000.

[4] Daniel Wiklund and Dake Liu, "SoCBUS: Switched network on chip for hard real time embedded systems," in *Proc of the International Parallel and Distributed Processing Symposium (IPDPS)*, April 2003.

[5] Fenghao Mu and Christer Svensson, "Self-tested self-synchronization circuit for mesochronous clocking," *IEEE Transactions on Circuits and Systems*, vol. 48, no. 2, pp. 129–140, 2001.