Asynchronous Microprocessor

Sanskriti Pahinkar, Harshita Negandhi

Electronics Department, D.J Sanghvi College of Engineering

***Abstract*-** In today's fast growing world, efficiency and speed plays an important role. Asynchronous design has been proposed to overcome various problems incurred due to synchronous architectures. This thesis starts with comparing synchronous and asynchronous designs and further describes different asynchronous process designs.

***Index Terms*-** asynchronous design, clock skew, handshaking, micro-pipeline, synchronous processor.

I. Introduction

As technology evolves into submicron level, synchronous circuits which are based on a single global clock have incurred problems of clock skew, design complexity and electromagnetic compatibility. This little device-a clock, can increase the circuit silicon and power dissipation, which can lead to overheating and in turn affect power supplies. In an attempt to overcome these limitations, researchers are actively considering asynchronous processor design. Globally Asynchronous Locally Synchronous (GALS) scheme, in which all communications between clock domains are handled using dedicated communication channels, is widely used. These communication channels use asynchronous handshaking protocols to transfer information between clock domains. Thus instead of a global clock when data can be moved from one unit to another, asynchronous units employ local handshake over asynchronous channels [Hau95, Sei80].

II. Synchronous v/s Asynchronous Processors

In this fast moving technological world, synchronous processors are made of hundreds of millions of transistors with clock rates up to several giga hertz. These high clock rates only imply that millions of transistors switch several billion times a second, regardless of the work to be done or not. This leads to tremendous power loss which can be eliminated with the help of an asynchronous processor.

Another major problem revolving around a synchronous process is the clock skew. A lot of efforts are required in distributing clock pulses throughout the chip in a manner such that it reaches all sinks at exactly the same point in time. This can sometimes be eliminated using additional wires at the cost of chip size and energy distribution.

Speed of a synchronous processor can be improved by optimizing the critical path. This is done either by optimizing the elements or more than one clock cycle need to be accounted, which is definitely not desired.

On the flip side, in asynchronous processors, there are several starting points. Each optimized element can speed up the processor. Thus, unlike synchronous architectures, only the frequently used elements are optimized, thus improving the average speed of the processor.

III. Asynchronous Processor Design

In a synchronous system, a clock is used to define points in time where all elements will have valid and stable data at their interfaces. The output signal may make several transitions between two clock pulses and none of them has to be valid and it has no successor. To overcome this, a technique called handshaking is used in asynchronous processors.

Signalling Protocol:

Handshaking is the one of the most important concepts of an asynchronous design. It transfers data quickly without taking into account the slowest link in the chain. Thus, there is no critical path which will determine the minimum cycle time.

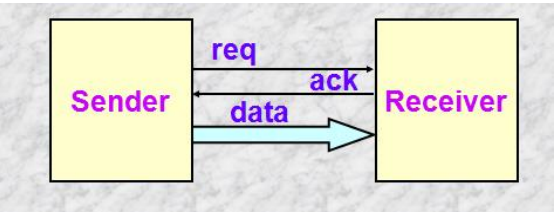


Figure 1: req: initiate an action, ack: signal completion of that action

There are different types of signalling protocols:

a) Control Signalling
   - Two Phase Handshaking Protocol
   - Four Phase Handshaking Protocol

b) Data Signalling
   - Bundled data with
     - Two-phase HP's
     - Four-phase HP's
   - Dual Rail Data with
     - Two-phase HP's
     - Four-phase HP's

III.a.1 Four Phased Handshaking Protocol

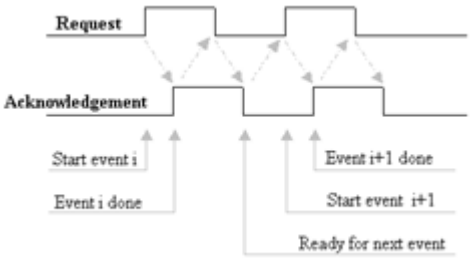


Figure 2: 4-phase Asynchronous Signalling Protocol

Four phased handshaking protocol, also known as 4-cycle, RZ (return to zero), and level signalling protocol, is one the most widely used protocols because of its ease of implementation. 4-phase signalling requires 4 control signal transitions (request rising, acknowledgement rising, request falling, and acknowledgement falling) per data transfer. In this protocol there are typically 4 transitions, 2 on the request and 2 on acknowledge, which are required to complete a particular event transaction. However, this sequence can cause a degradation in the overall system performance in a global interconnect assuming relatively long distance communications.

III.a.2 Two Phase Handshaking Protocol

This is one of the pervasive choices for asynchronous design. This protocol is also named as 2-cycle and NRZ (non-return to zero) protocol.

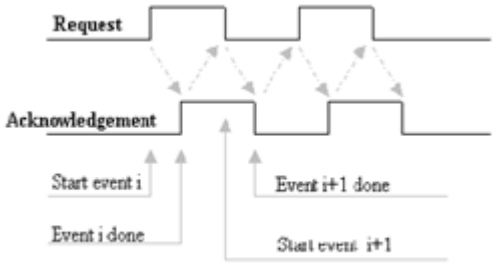


Figure 3: Two phase Asynchronous Signalling Protocol

Every transition on the request line falling and rising indicates a start of new event (request). The same is true for the transitions on the request on acknowledge line. Unlike four-phase signalling protocol, two-phase signalling reduces the transitions by half, it is more effective in terms of performance and power consumption. For this reason, despite its design complexity, two-phase signalling is recommended as an implementation method for asynchronous global interconnects.

III.b.1 Bundled Data Handshaking Protocols

These protocols are called as single-rail though bundled-data is used to describe the

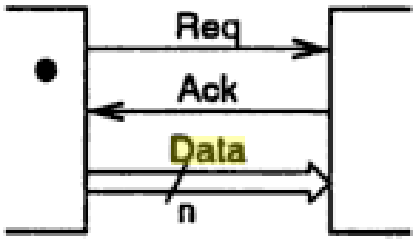


Figure 4: Bundled data (push) channel

simultaneous transmission of control and data signals, whereas single rail describes usage of one wire for each data bit. Bundled-data refers to a situation where the data signals use normal Boolean levels to encode information, and where separate request and acknowledge wires are bundled with data signals.

III.b.1.i Four phase bundled-data protocol

The term four-phase refers to the number of communication actions. First, the sender issues data and sets request high, followed by the receiver absorbing the data and setting acknowledge high. Next, the sender responds by taking request low and the receiver acknowledges this by taking acknowledge low. At this point, the sender may initiate the next communication cycle.

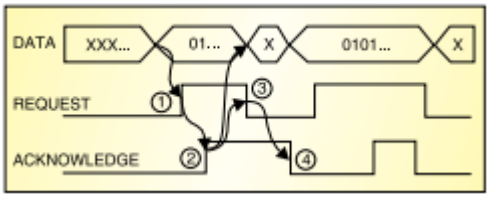
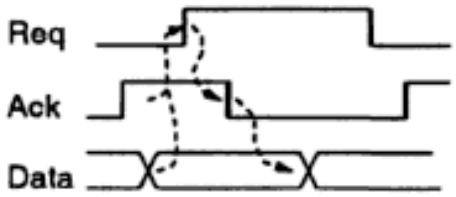


Figure 5: 4-phase Bundled-data Protocol

This type of protocol has relatively more switching activity, which may lead to slower and more energy consuming circuits.

III.b.1.ii Two phase bundled-data protocol

A four-phase bundled data protocol has a disadvantage in superfluous return-to-zero transitions that cost unnecessary time and energy. This is overcome by a two-phase bundled data protocol.



Figure

Figure 6: 2-phase Bundled-data Protocol

The information on request and acknowledge wires is now encoded as signal transitions on the wires and there is no difference between a 0 to 1 or a 1 to 0 transition as they both represent a "signal event". Ideally, a two-phase bundled data protocol is faster than a four-phase protocol.

This protocol is efficient in both time and energy because of its very little switching activity. Components sensitive on transitions are more complex than elements, which just react to signal levels.

Two-phase protocol is widely used in AMULET3 and is often referred to as 'Micro-pipeline'.

When the sender is the active party that initiates the data transfer over the channel, it is known as push-channel. On the other hand, the receiver asking for new data is also possible and is known as pull-channel. In the latter case, directions of request and acknowledge are reversed and validity of data is indicated in the acknowledge signal going from sender to the receiver.

III.b.2 Dual-Rail Protocol

A dual-rail protocol is a more sophisticated protocol that is robust to wire delays.

III.b.2.i Four-phase Dual-rail Protocol

A four-phase dual-rail protocol encodes the request signal into data signals using two wires per bit of information that has to be communicated.

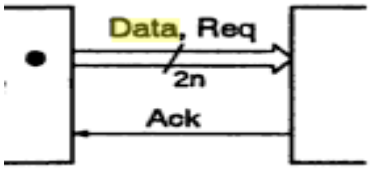


Figure 7.a

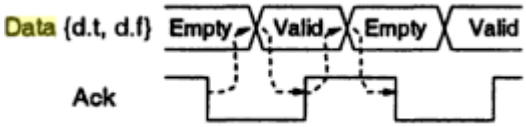


Figure 7.b

In actuality, it is a four-phase protocol using two request wires per bit of information *d*; one wire *d.t* is used for signalling a logic 1 i.e true, and another wire *d.f* is used for signalling logic 0 i.e false.

	d.t	d.f
Empty ("E")	0	0
Valid "0"	0	1
Valid "1"	1	0
Not used	1	1

Figure 7.c

Figure 7: 4-phase dual rail (push) channel

Thus, a sequence of four-phase handshakes are seen, where the request signal in any handshake signal can be either *d.t* or *d.f*. In this type of protocol, two parties can communicate reliably regardless of the delay in the wires connecting the two parties, which makes this protocol delay insensitive.

IV. PIPELINING

A technique used in advanced microprocessors where the microprocessor begins executing a second instruction before the first has been completed.

Pipelining is an implementation technique that exploits parallelism among instructions in sequential instruction stream.
A conventional computer pipeline is a synchronous pipeline which is controlled by a global clock. In synchronous system each operation of an arithmetic has to be finished within a given time slot by overall clock signal. Data signals have to be stable at latching time. The clock period of synchronous pipeline is limited to a minimum of time taken for slowest pipeline stage to complete its processing.

By contrast an asynchronous pipeline does not have any global clock, hence every stage can take a variable time to finish and can work independently. Therefore the next stage can begin after the previous stage has finished which theoretically makes asynchronous pipeline faster than synchronous pipeline.

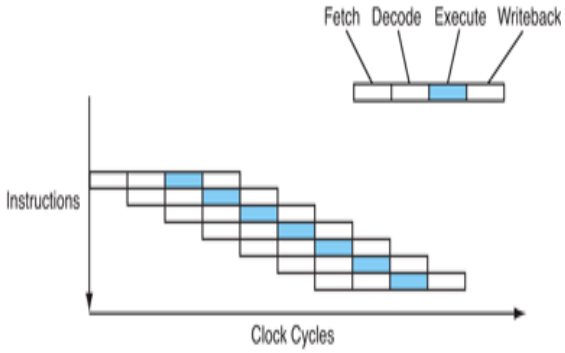


Figure 8: 4 stage pipeline

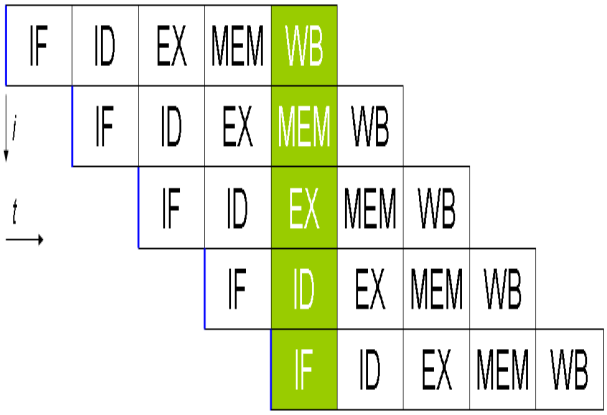


Figure 9: 5 stage pipeline

V. Drawbacks

The handshaking mechanism, which introduces overhead may slow down the circuit or even cost more energy than it saves.

Secondly, there is lack of a proper computer-aided design (CAD) tool.

Sometimes, it is difficult to exchange information between pipeline stages, which results in redundant data storage.

Furthermore, the control logic for asynchronous processors is more complex than synchronous processors, which leads to comparatively high energy consumption.

VI. Conclusion

On a whole, asynchronous processors have specific advantages like low power consumption and good electromagnetic compatibility. Asynchronous methodology can exploit the simplicity provided by sequential computation while attaining performance benefits by beginning the next computation as soon as the previous one is completed, instead of having to wait for the next clock pulse. In the future, asynchronous processors may also benefit from techniques developed for synchronous architectures, like lowering the supply voltage, which is easy to implement in an asynchronous system due to handshaking which provides the necessary flexibility.

References

[1] Principles of Asynchronous Circuit Design: A Systems Perspective edited by Jens Sparsø, Steve Furber

[2] SCALP: A Superscalar Asynchronous Low-Power Processor, Philip Brian Endecott, Ph.D thesis of University of Manchester, UK, 1996

[3] AMULET1: An Asynchronous ARM Microprocessor, J. V. Woods et al, IEEE Trans. on Computers, Vol.46.4, p.385-398, Apr 1997

[4] Automating the Design of an Asynchronous DLX Microprocessor, Manish Amde et al, DAC'03, p.502-507, June 2003

[5] Research feature on 'Asynchronous Processor Survey' by Tony Werne Venkatesh Akella, University of California, Davis

[6] Electronics for you

Authors

First Author – Sanskriti Pahinkar, pursuing B.E in Electronics, Dwarkadas J. Sanghvi College of Engineering, email id: sanskriti.pahinkar@gmail.com

Second Author – Harshita Negandhi, pursuing B.E in Electronics, Dwarkadas J. Sanghvi College of Engineering, email id: hrn0502@gmail.com