MODULAR DESIGN OF DIGITAL CIRCUITS

Loek Thijssen

Delft University of Technology, Faculty ITS,
PO Box 5031, 2600 GA Delft, The Netherlands
E-mail: A.P.Thijssen@its.tudelft.nl

Abstract

Since the beginning of modular design of digital circuits with the 7400 series of standard integrated circuits in 1970 the complexity of standard digital integrated circuits has grown rapidly. With this growth the number of problems coupled with the application of standard library modules in ASICs or on printed circuit boards has grown as well. A good estimate is that more than 40 % of the design effort in PCB design is spent for solving problems in interfacing these standard type of circuits. The reason for this amount of overhead is the lack of standardization in interfacing. The only way out is a strict standardization of the interface hardware and software. This paper presents an overview of the different design levels to be standardized and concentrates on some practical solutions for frequently occurring interface problems.

Keywords: Modular design, interface standardization, shorter time-to-market.

1 Introduction

Reuse of previously designed modules is generally assumed to be cost effective and can lead to a shorter time-to-market. In practice this is not always true. Problems may arise from the physical interface level during data transfer resulting in so-called hazards, soft errors that are very hard to detect. They sometimes originate from a critical signal timing, clock skew, insufficient noise margins, susceptibility for electromagnetic induction in low-voltage circuits, etc. The designer has to verify that timing and other margins are sufficient to compensate for these phenomena.

In practice this verification process is difficult because of a lack of details about the range of timing and other parameters. Especially temperature and technology-dependency of the parameters are seldomly documented in a sufficient way. For a designer it is nearly impossible to make an accurate estimate for worst case and best case values of the parameters. The parameters as specified in data books are not reliable. Their measuring procedure is not always specified. So the designer cannot verify if the parameters are correct or not.

Take for example a two-input CMOS NOR gate. The minimum propagation time must be measured while the two parallel nMOS transistors are switched simultaneously. The maximum propagation time must be measured while the

pMOS transistors switch one by one. Also, the previous set-up of the gate has some influence on the propagation time. In any case, when the manufacturer specifies the minimum propagation time as a fixed percentage of the maximum you know the timing parameters as specified in the databook are not reliable.
Another example has been described by [Samsom, 1993]. The data for a flip-flop must be available in the so-called setup and hold time interval at the D input. This interval is usually measured by shifting a rising or a falling *data edge* around the clock edge. When the data is accepted by the flip-flop or not is used as an indication of where the set-up interval begins and hold interval ends. This procedure is not correct for two reasons. Samsom demonstrated that the setup and hold interval must be measured by shifting a *pulse* around the clock edge, according to the shape of the interval to be measured. Significant parameter deviations were found. Second, the increase of the output propagation time is a better criteria than the criteria of data acceptance.
We conclude from the above that a logic designer sometimes has to go into all physical details of the modules to be used and their compatibility. A way out is to increase all margins.

2 The voltage interface

The voltage level is the lowest interface level. In Figure 1 the so-called AC and DC noise margins have been defined. These noise levels are essential for reliable data transfer between sender and receiver. They create margins for electro-magnetic interference from other electrical equipment's.

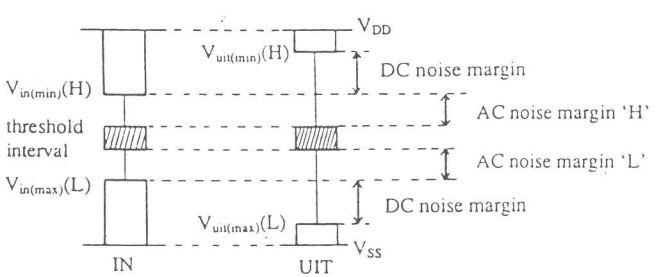

Figure 1 *AC and DC noise levels.*

Recently we see a design trend to use lower power supply voltages, because of power reduction in large integrated circuits. This trend automatically implies lower noise margins. As a result, noise margins have become very critical. Every user of a portable phone in a plane or hospital knows this effect. There is an easy way out of this power reduction versus reliability conflict. Two DC supply voltage levels should be used, a 5 Volts external data interface and an internal 2

Volts power supply for the internal data processing. A standard *internal* 5 V $\leftrightarrow$ 2 V data interface solves all level conversions. This interface should be subject to a world wide standardization.

3 The timing interface for clocked data transfer

When two modules are exchanging clocked data it is necessary that the data output of the sender (FF1) remains stable during the set-up and hold interval of the receiver (FF2) and reverse. When both sender and receiver have the same external clock pulse there are two margins for clock skew [Thijssen, 2000]

$$t_{skew}(FF1 \rightarrow FF2) = t_{P(min)}(FF1) - t_h(FF2) \quad (2.1)$$

$$t_{skew}(FF1 \rightarrow FF2) = T_{clock} - t_{P(max)}(FF1) - t_{su}(FF2) \quad (2.2)$$

The first margin depends only on the flip-flop parameters and the second margin includes a part of the clock period T_{clock} as well. The first margin frequently becomes critical when modules are used, that have been realized in different processes or when different flip-flop structures are used internally in the ICs. We find this situation while testing printed circuit boards. Boards are usually tested by a Boundary Scan [JTAG, 1988] facility. In Boundary Scan the output of a scan line of a circuit sends on one edge of the clock and the scan input of the next circuit reads data on the other clock edge. This makes all margins for clock skew dependent from the clock period, as has been shown in Figure 2. In general, the value of the minimal available margin for clock skew becomes significantly higher. This timing system is called a *data lockout timing*.

The conclusion is that with a data lockout timing on *all* data outputs and inputs clock skew between different circuits or modules will generally be no problem at all. Only on very high-frequent boards a data lockout timing does not solve all skew problems. In this case we must use different clock frequencies for internal data transport within modules and data transport in between modules.

Modern digital circuit processing allows very high clock frequencies. That means that the built-in margins for clock skew in flip-flops easily become critical. Timing verification programs may indicate the critical spots for clock skew. The daily practice is that with some additional combinational logic gates data or clock signals are delayed in such a way that clock skew is no longer critical. This is a bad approach. When using predesigned modules in a greater design the timing parameters seldomly are documented in a sufficient and relevant way. That means that a system designer has no idea if the circuit timing becomes critical or not. Another point is that during the life time of a system parameters may change slightly during the life cycle. This phenomenon asks for predefined timing and other margins, as much as possible independent of technological parameters.

A data lockout timing provides, as we have seen In Figure 2, margins for clock skew that not only depend on the technological timing of the flip-flops of a circuit, but also on its clock period. In a data lockout timing the clock period has

been split up between both margins for clock skew, creating predefined and guaranteed margins for clock skew. So the preferred approach in the design of larger circuits is to define so-called clock macros, parts of a circuit that are small

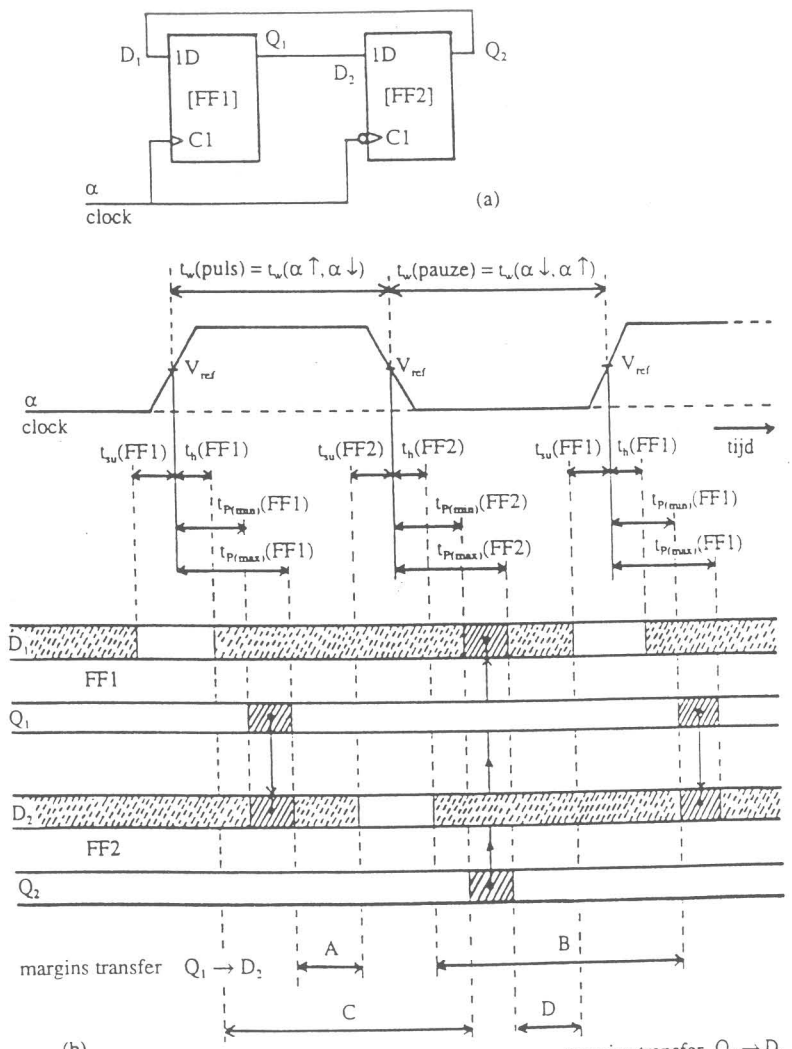

Figure 2. Clocked data transfer

enough to cope with the current clock skew problems related to flip-flop parameters and built in margins for clock skew. In between clock macros a data lockout timing is used, solving all external timing hazards. A databus is a good example of such a predefined clock macro.
An implementation of a data lockout timing in between two *existing* positive edge-triggered clock macros can easily be done by adding a negative edge-triggered register in between the positive edge-triggered macros. As a matter of fact, the data timing in between the edge-triggered macros is completely transparent for this extra register. This solution applies for internal as well as for external clock macros like integrated circuits on a printed circuit board. For *new designs* a negative output timing and a positive input timing, or reverse, should be a world-wide standard on data inputs and outputs. In HF logic design there is no choice at all!

4 A standard RESET interface

On the system or board level there is an urgent need for standardization as well. In a long period of teaching digital design courses the common RESET function have appeared to be a constant source of problems. Most circuits have an asynchronous reset. Such a reset input in an antenna for EMI. Measurements have shown this several times. Error probabilities of lower that 1 at 10^5 clock pulses have been found under noisy conditions. A disaster for reliability! But synchronous resets frequently give problems as well. The following list is far from complete.

- Activating an external reset button produces an asynchronous reset signal. This signal should be synchronized before using it in a synchronous environment.
- Deactivating the reset button is asynchronous as well and should also be synchronized.
- The minimum duration of the reset pulse may vary over the various types of circuits. Sometimes the presence of the reset signal during one active clock edge is sufficient to reset the circuit. In other cases a synchronous reset must be present during several clock pulses. Which unit takes care for this extension of the reset command?
- The encoding of the reset command is usually done by activating a reset signal. Sometimes the reset command is encoded with several setup signals from the control unit. The Boundary Scan interface is a good example of an encoded reset command.
- Problems may arise when one or more modules have been reset, while others are still being resetted. There is no central 'reset done' message.
- After a reset usually a setup or initialization procedure starts in several modules. Other modules are ready to begin their normal operation cycle. Sometimes a hardware dead lock situation is the unwanted result.

So not only the end of the reset procedure of all modules must be detected or synchronized, but also the end of the initialization procedures. The device is ready to begin its normal activities if and only if all reset and initialization procedures of all modules have been finished and synchronized.

In the past most standard digital modules have been designed with a reset input and a manual how to use this input. There are as many manuals as there are standard modules. This means that the designer of a printed circuit board has to design extra circuitry in order to adapt an external reset signal for use on the board under design. Simply interconnecting all reset signals does not solve the reset interface!

In order to make reuse of digital modules in greater designs possible and cost-effective it is necessary that this reset problem (and other problems as well) are solved on a worldwide scale. If not, redo a complete design is the only correct conclusion.

Part of this standard reset interface may be an open-collector or an open-drain bus line. The external reset makes the bus LOW. This active level LOW must be recognized by a kind of bus master. The output of this bus master takes over the reset bus and makes the bus LOW as long as necessary for all units being reset. Every unit in turn makes the bus LOW as long as the unit is being reset. When the reset bus becomes HIGH all units have been reset. Normal control may start the function of the device now

Conclusions

Reuse of standard modules in greater designs is only profitable if any module preserves its functionality in the new environment. That means that the hardware interface of modules must be designed for reuse, and not only for the original functionality. Worldwide standardization saves a lot of design time. The costs, some hardware overhead, are a peanut when compared to the present-day external hardware and design time overhead.

Literature

1. JTAG 1988: Boundary Scan Standard
2. C. Maunder and R.E. Tullos, The Test Access Port and Boundary Scan Architecture, IEEE Computer Society Press Tutorial, Washington 1990.
3. S. Samsom, Clocking of Digital Systems, Master's Thesis TU Delft no 1-68340-28(1993)24
4. A.P. Thijssen, Digitale Techniek, Delft University Press, Delft 2000.