Asynchronous FPGA Risks

Ken Erickson
California Institute of Technology
Jet Propulsion Laboratory, Pasadena, CA 91109

Abstract

A logic design can be a synchronous or asynchronous structure. The worst case timing margin of a synchronous design implemented with a field-programmable gate array (FPGA) is easy to perform using available FPGA design tools. However, it may be difficult to impossible to verify that worst case timing requirements are met for a complex asynchronous logic design. As the result, there is a risk that a timing problem can occur as the result of temperature variations, aging and radiation. Such a timing problem could have a serious impact on a space mission or other applications. For this reason, it is recommended that the risks of asynchronous FPGAs be assessed and managed. Project management can decide whether each risk is acceptable or whether action should be taken to mitigate the risk. The preferred mitigation action is to redesign the logic to make it a synchronous design.

I. Synchronous versus Asynchronous Logic

A logic designer often has the choice of implementing FPGA logic using synchronous or asynchronous logic structures. The terms "synchronous" and "asynchronous" have different meanings to different people. For the purpose of this paper, a logic circuit is synchronous when all flip-flops are clocked on the same edge of the same global, low-skew clock. When this is not the case, the design is asynchronous. The following are examples of asynchronous logic:

1. All flip-flops are not clocked using the same edge of the same global, low-skew clock.
2. A flip-flop is clocked by the output of a gate, another flip-flop or another logic element.
3. The asynchronous set and reset inputs of a flip-flop are used to control the state of the flip-flop for purposes other than initialization (e.g. power on reset).

A synchronous design can be easily analyzed using available FPGA design tools, such as Actel Timer, to verify that the logic performs the intended function and that flip-flop setup and hold times are met under worst case conditions.

However, it may be very difficult to verify that a complex asynchronous logic design meets performance requirements under worst case conditions. The reason is that FPGA design analysis tools are tailored to efficiently perform timing analyses of synchronous logic, not asynchronous logic. As the result, the designer will be required to manually check the worst case timing of each clock and data path to verify that worst case setup and hold time requirements are met for all flip-flops. For complex asynchronous logic structures, this process may be extremely tedious or virtually impossible. The end result may be a design having unknown setup and hold time margins as well as a design that has the risk of failing due to timing race conditions. If a timing problem is found and corrected, the process of performing timing analyses of all possible delay paths should be repeated for the simple reason that a change in one part of the logic may affect another part of the logic due to signal rerouting within the FPGA.

If a worst case timing analysis of an asynchronous logic design is not performed, then the worst case timing margins of the design are unknown. In this case, the design may be a ticking time bomb.

Several examples of asynchronous logic structures and potential problems associated with them are presented in [1].

However, there are design applications where asynchronous logic is justified. The following are examples of such applications:

1. Speed requirements cannot be met with a synchronous design.
2. Power requirements cannot be met with a synchronous design.
3. When dictated by requirements.
4. To solve special problems

Shown in figure 1 is an example of the use of asynchronous logic to solve a special problem. A divide-by-16 frequency divider is required to generate a 1 MHz clock from a 16 MHz clock signal. However, the low-skew buffers provided by the FPGA are all being used. If a synchronous counter were to be used for the frequency divider that is clocked using a non low-skew buffer, potential timing problems could occur as the result of clock skew. The solution adopted is to use a four stage ripple counter for the divider function. This circuit is simple to analyze and performs the desired function with no potential timing problems.
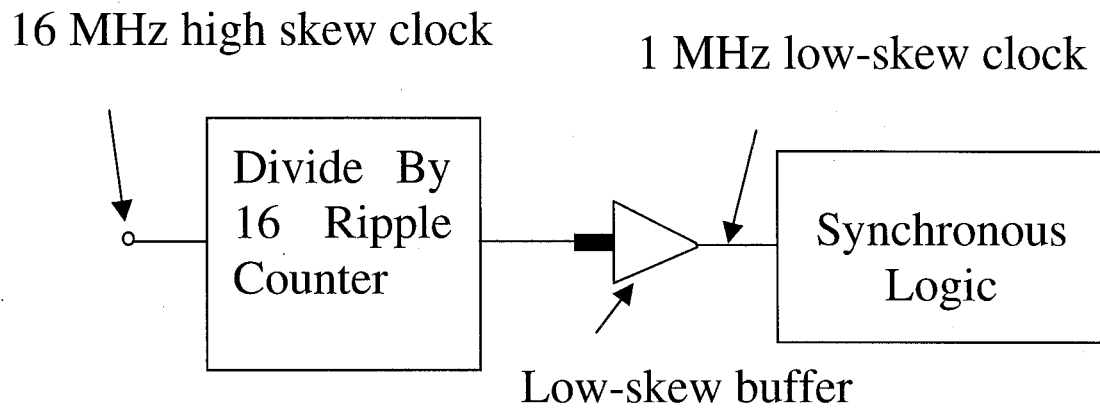
16 MHz high skew clock

1 MHz low-skew clock

Divide By 16 Ripple Counter

Synchronous Logic

Low-skew buffer

Figure 1. Ripple Counter Application

II. Dealing with Asynchronous Inputs

When asynchronous data is clocked into a flip-flop, there is a small probability that the output of the flip-flop will exhibit an unpredictable delay. This occurs when the input transition not only violates the setup and hold-time specifications, but also occurs during the tiny timing window where the flip-flop accepts new input. Under these conditions, the flip-flop can enter a symmetrically balanced transitory state called a metastable state. The effect is called metastability. This effect and its likelihood of occurring are discussed in detail in [1].

While in the metastable state, the flip-flop logic level is indeterminate. When metastability occurs, the problem is not just the illegal logic level in the balanced state, but the unpredictable time for a valid, stable logic state to be achieved. Problems that can result are as follows:

1. Multiple transitions could be detected from a single transition of the asynchronous input.
2. If the asynchronous input is clocked into multiple flip-flops, the flip-flops may not have the same output states.

A logic circuit commonly used to interface an asynchronous input signal to synchronous logic is shown in figure 2. When metastability occurs, a stable logic state will be clocked into flip-flop FF2 provided that the output of flip-flop FF1 stabilizes to a valid logic state within one clock period.

It is best if the logic circuit is designed in such a way that a metastabiltiy condition will, at most, result in a self-clearing intermittent condition and that the consequences will not be serious. For this reason, the designer should consider these affects when designing the logic.

III. Why Marginal Designs Go Undetected

The following are reasons why marginal designs go undetected and find there way into spacecraft:

1. A detailed peer review of the design is not performed. Often the assembly containing FPGAs is reviewed at a high level, but a detailed review of the FPGA logic is not performed. For example, the detailed logic design may not get reviewed at the critical design review. This writer has experienced this first hand. A design review is held where an overview of the design is presented using viewgraphs and the board schematics are available in the room for those interested in looking at them. However, these schematics show the FPGA symbols, but do not provide the design documentation for the logic contained in the FPGAs. In some cases, the logic designs had not even been completed as of the date of the design review. Approving a design on this basis is like giving someone a signed blank check.
2. Complete worst case timing analyses are not performed for the design. The result is unknown timing margins.

3. The risks of asynchronous designs are not being assessed and mitigated.

IV. Asynchronous Design Risks

Assume that an asynchronous FPGA having unknown timing margins is used in a spacecraft application. During the mission, there is the possibility that the assembly using the FPGA will not perform its intended function as the result of logic timing variations due to aging, radiation and temperature. The result could be an intermittent problem or a permanent failure. If the FPGA is used in a critical assembly, a failure of the FPGA to perform its intended function could result in the end of the mission. Even if the FPGA is used in an assembly that has a redundant counterpart, the same problem or similar problems can result in both units.

Another potential problem of an asynchronous design having unknown timing margins is the possibility of failure of the design late in the test program due to timing problems. The later the failure occurs, the higher the impact due to the project cost and schedule delays. For example, a failure that occurs just prior to launch could have a large impact on both cost and schedule.

Unfortunately when timing problems occur in asynchronous designs, the cause may be difficult to determine. As a result, it may take a long time to determine what the cause of the problem and the solution.

V. Risk Assessment

Many space projects use risk management systems to identify and manage risks. Using risk information, project management can weigh risks against the cost and schedule impacts of mitigating the risks. NASA requirements for establishing effective risk management are given in [2].

In order to assess the mission risk caused by asynchronous FPGAs, there are two factors to consider. These factors are (1) the likelihood of a failure and (2) the impact of a failure. The risk is determined by the combination of the two factors.
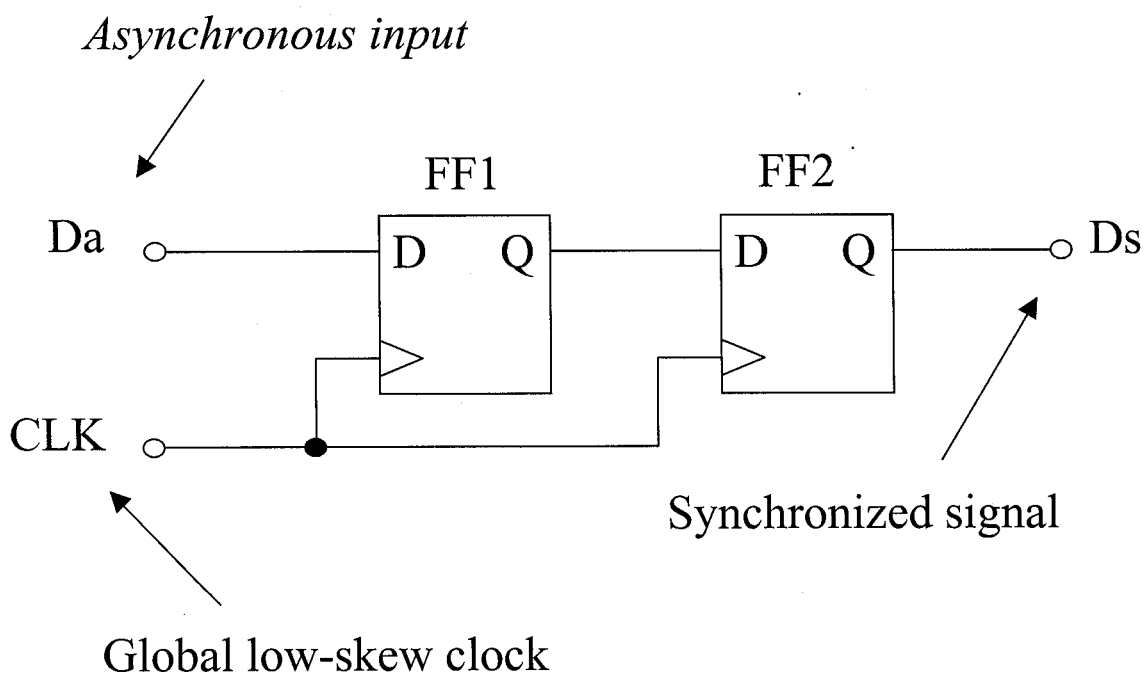
Asynchronous input

Da

CLK

FF1

FF2

D   Q

D   Q

Ds

Synchronized signal

Global low-skew clock

Figure 2. Synchronizer

The following methods can be used to assess the risks of asynchronous designs:

1. Detailed peer review by a logic design expert.
2. Complete or partial worst case timing analysis
3. Voltage/temperature margin testing

If a peer review is performed, the reviewer should look for potential problems. This review may include performing worst case timing analyses of suspect areas of the design. Some potential problems to look for are as follows:

1. Potential race conditions
2. Potential clock skew problems
3. Gated clocks
4. Decoder outputs used as clock or reset input
5. Potential metastability problems

Based on the peer review, the reviewer's assessment of the design should be reported. This assessment should identify potential problem areas, the impact on the mission if failures occur, the likelihood of failure and recommendations for mitigating the problems. The likelihood of the failure is a judgement call by the reviewer that considers the following information:

1. The nature of the designs
2. The design complexity
3. Design concerns
4. Mission aging and radiation affects

Another method of determining the risk of the design is to perform a partial or complete worst case timing analysis.

Voltage temperature margin testing can be performed on the design. During this test both the supply voltage and temperature are varied in order to determine the susceptibility of the design to delay variations. If the design being tested includes a regulated power supply, the FPGA supply voltage cannot be varied unless there is a way to temporarily disconnect the regulator from the FPGA and use an external power supply. It is recommended that small temperature steps (such as 1 degree centigrade) be used during this testing since timing problems may occur over only a narrow temperature range. When the supply voltage is increased, the delays decrease. When temperature is increased, delays are increased. Even though successful testing is a good indication, it does not prove that sufficient timing margin exists. One reason is that some delays can increase and other delays in the same FPGA can increase at the due to radiation or aging. Voltage/temperature margin testing does not simulate these affects. However, if the design fails during testing, this is in indication of design problems.

The risk of the design can be assessed based on any combination of the above risk assessment methods.

VI. Risk Mitigation

Once the risk is assessed, the decision must be made on whether or not the risk should be partially or completely mitigated. Possible risk mitigation options are as follows:

1. Redesign logic to make it a synchronous design. This is the preferred option if it not precluded by budget or schedule constraints or other factors since asynchronous risks are completely mitigated. A timing analysis of the new design can then easily be performed using available FPGA design tools to verify that worst case timing requirements are met. This action completely mitigates the risk.
2. Perform complete timing analyses of the design and correct problems found. This may be next to impossible if the design is complex and may take more time and be more costly than a complete redesign. For a complex asynchronous design, it may be difficult to know if all possible signal paths have been analyzed. When this is the case, you cannot be certain that the risk has been completely mitigated. However, if complete timing analyses are performed and it is verified that adequate timing margin exists, the risk is completely mitigated.
3. Modify selected portions of the design that have been identified as having the highest risks. A worst case timing of analysis of the modified logic should then be performed to verify that worst case timing requirements are met. This action only partially mitigates the risk. However, it may reduce the risk sufficiently to convince the project manager that it is flight worthy.
4. Perform voltage/temperature margin testing. In this case the assembly using the FPGA is tested over temperature at different supply voltages (if permitted by design). This testing, if successful, only partially mitigates the risk.
5. Prayer. If the risk of asynchronous logic is not completely mitigated, luck or divine intervention may be needed to ensure that problems will not occur during the mission.

Different combinations of the above mitigation actions can be performed to mitigate the risk. For example, a partial redesign and voltage/temperature margin testing can both be used to partially mitigate the risk.

VII. Space Hardware Async FPGA Problems

The following of flight problems have occurred on flight designs due to marginal asynchronous FPGA logic:

1. A spacecraft instrument had many problems during system testing due to marginal asynchronous logic in FPGAs. The corrective action was to redesign all FPGAs to make them synchronous. As the result, significant additional costs and schedule delays resulted.
2. A spacecraft experienced inadvertent resets during system testing. The problem occurred over the 17 to 20

degree centigrade temperature range. The problem was solved by redesigning FPGAs in the flight computer.

3. Problems were found by performing breadboard testing of a mission critical spacecraft assembly. The following potential problems were found as the result of a peer review of the FPGA logic:
   - The FPGA contained a complex asynchronous logic design having unknown timing margin
   - A potential race condition was identified. The setup time for a particular flip-flop could not be guaranteed
   - A decoded counter output was used as a clock. Decoding glitches could result in advertent clocking.

The findings of the review were presented to project management. Project management requested that the asynchronous design be redesigned to make it synchronous. The redesign was completed and the worst case timing margin was verified to mitigate the risk.

VIII. Recommendations

The following actions can be used to virtually eliminate future asynchronous design risks:

1. Prepare a design requirements document for each FPGA-based logic design. It is important to clearly define the design requirements and the interface signals. This document can be reviewed and approved by all affected parties. It will not only help nail down the requirements, but it will help others, including a peer reviewer, understand the design requirements.
2. Design synchronous logic where possible.
3. Perform <u>complete</u> worst case timing analyses of <u>all</u> designs.
4. Conduct a peer review of each logic design by an expert logic designer. Potential problems can be identified for the reviewer. Even synchronous designs may have potential problems such as:
   - Potential problems while powering up FPGA
   - Critical functions affected by single event upsets
   - Potential initialization problems
   - Potential lockup problems
   - Sneak paths through unpowered logic
   - Potential timing problems at FPGA to FPGA interfaces
5. Assess the risk of all asynchronous logic designs that have not been verified to meet worst case requirements and submit the assessment to project management.
6. Mitigate asynchronous logic design risks, if feasible

IX. Conclusions

- Asynchronous logic designs can result in risks to space missions
- Use asynchronous logic only when required, but perform complete worst case analyses

- Assess the risks of all asynchronous logic designs which have not been verified to worst case timing requirements

REFERENCES

1. R. Katz, R. Barto, and K. Erickson, "Logic Design Pathology and Space Flight Electronics," MAPLD 1999, Laurel, MD.
2. NASA AE/Office of Chief Engineer, NPG 7120.5A, **NASA Program and Project Management Processes and Requirements**, paragraph 4.2, April 3, 1998.

ACKNOWLEDGEMENT

Asynchronous FPGA Risks

Ken Erickson

California Institute of Technology

Jet Propulsion Laboratory

Pasadena, CA 91109

(818) 354-9459

kerickson@jpl.nasa.gov

Agenda

- Synchronous versus asynchronous logic designs

- Asynchronous designs

  - When to use
  - Risks
  - Risk Assessment Methods
  - Risk Mitigation Options

- Examples of Space Hardware FPGA Problems

- Recommendations

- Summary

Synchronous versus Asynchronous Logic

- Synchronous logic

  - All flip-flops are clocked using a single global low-skew clock

  - FPGA design tools can be used to easily verify that setup and hold time margin exists under worst case conditions for flip-flops

- Asynchronous logic

  - All flip-flops are not clocked using a single global low-skew clock

  - An example is a flip-flop clocked by the output of a gate or another flip-flop

  - Another example is when asynchronous set and reset inputs of a flip-flop are used for purposes other than initialization.

  - Worst Timing analyses

    ◆ Requires separate timing analyses for each flip-flop

Synchronous versus Asynchronous Logic

♦ May require many analyses to analyze all possible clock and data paths for a single flip-flop

♦ For a complex design, complete worst case analysis may not be feasible

– FPGA design tools are tailored for easily performing timing analyses of synchronous logic, not complex asynchronous logic

– An asynchronous design may be justified in special instances such as when speed and power dissipation requirements cannot be met using synchronous logic

Dealing with Asynchronous Inputs

- When an asynchronous input is clocked into a flip-flop, the there is a possibility that the flip-flop can enter into the metastable state

  - The output state is indeterminate for an unpredictable period of time

  - The time period is probabilistic

- Possible consequences

  - Multiple transitions could be detected from a single transition of the asynchronous input

  - If the asynchronous input is clocked into multiple flip-flops, the flip-flop output states may not all be the same

  - Intermittent functional problems could occur

- Synchronizer

  - Two cascaded flip-flops clocked by the logic circuit clock

Dealing with Asynchronous Inputs

- Most of the time, the metastable state time duration of flip-flop FF1 is less than one clock period and a stable logic stable logic state is clocked into flip-flop FF2

- Design considerations

- Determine the affects of metastability on logic designs

- Attempt to design logic in such a way that the metastability conditions will, at most, result in a self-clearing intermittent condition with no serious consequences

Synchronizer

Asynchronous input

FF1   FF2

Da   D   Q   D   Q   Ds

CLK

Synchronized signal

Global low-skew clock
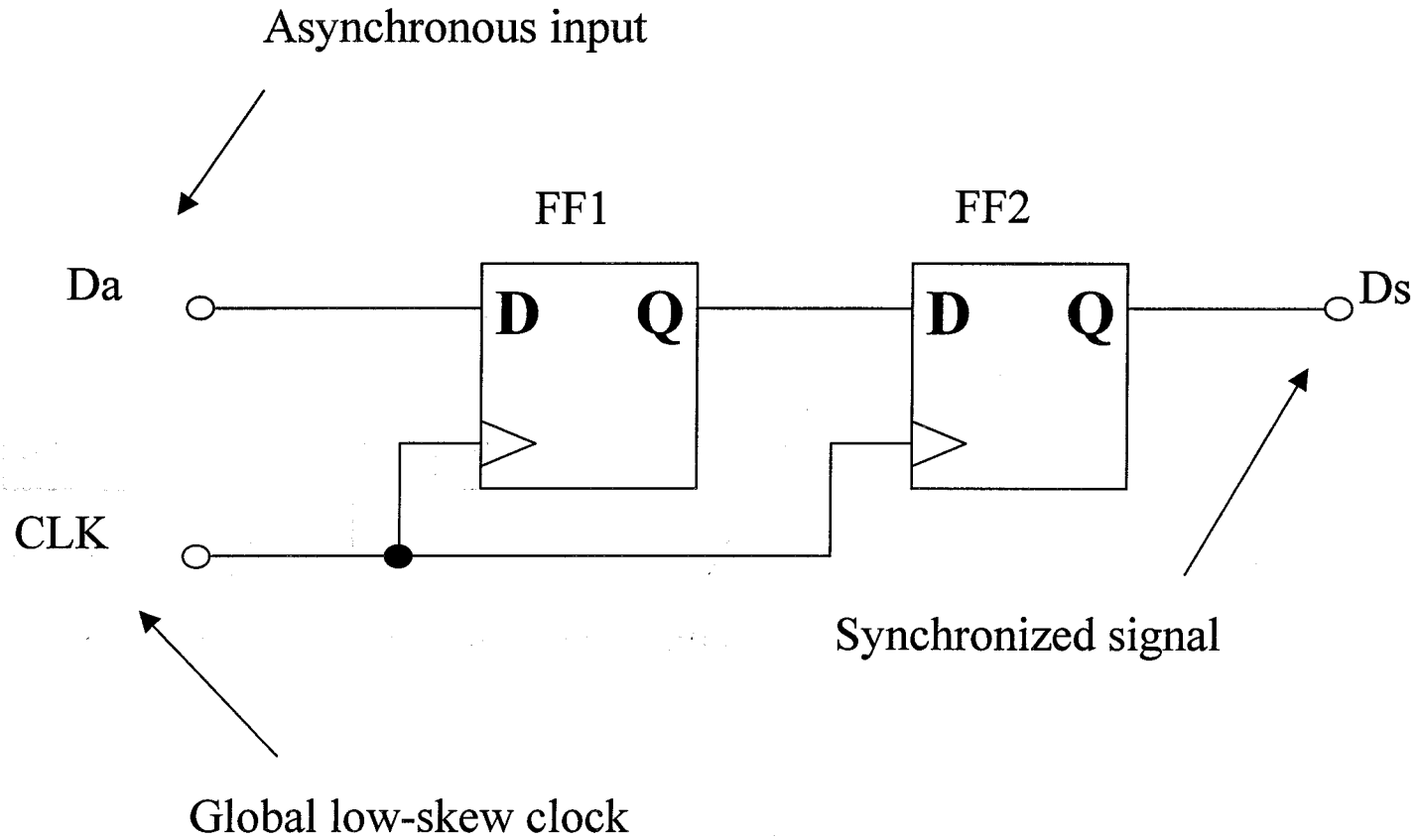
Why Marginal Designs Go Undetected

- Reasons why marginal designs go undetected and find there way into spacecraft

  - A detailed peer review of the design is not performed

  - Complete worst case analyses are not performed

  - The risks of asynchronous designs are not assessed and mitigated

- Complete worst case analyses of the

When to Use Asynchronous Logic

- When dictated by requirements

    - Asynchronous bus protocol, such as VME, for example

    - Can use synchronous logic if asynchronous input signals are synchronized to the clock of the synchronous logic

- May be the only way to meet speed requirement

- May be the only way to meet power requirement

- To solve special problems

Frequency Divider

16 MHz high skew clock

1 MHz low-skew clock

Divide By 16 Ripple Counter

Synchronous Logic

Low-skew buffer
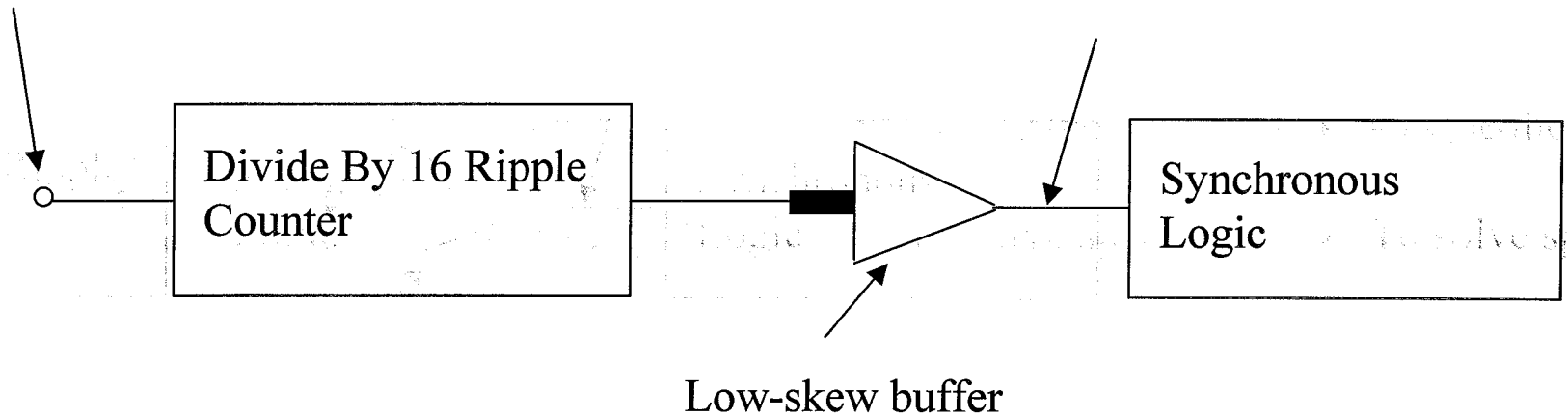
Asynchronous Design Risks

- Design may be marginal

- Design fail during the mission due to aging and radiation affects

- Impact of failure may be have a significant affect on mission or may result in the end of the mission

- Failures may occur late in the test program, resulting in expensive project delays

- May cause intermittent performance anomalies

- Cause of problems may be difficult to determine

Risk Assessment Methods

- Detailed peer review of design

  - Review worst case timing analysis, if performed

  - What to look for

    - ◆ Potential race conditions

    - ◆ Potential clock skew problems

    - ◆ Gated clocks

    - ◆ Decoder output used as clock or reset input

    - ◆ Potential metastability problems

    - ◆ Asynchronous reset problems

- Complete or partial worst case timing analysis

Risk Assessment Methods

- Testing

  - Use worst case input timing conditions

  - Voltage/temperature margin testing with small temperature steps (such as 1 degree centigrade

- Risk assessment considerations

  - Impact of failure on mission

  - Likelihood of failure

    - This is a judgement call if worst case timing margin is unknown

    - Consider the mission duration and radiation affects

  - Redundancy does not eliminate risk since both the primary and redundant circuits are susceptible to the same problems

Risk Assessment Methods

- Redesign logic to make it synchronous and perform timing analysis to verify worst case timing margin – completely mitigates risk

- Perform complete timing analysis

  - May not be feasible if design is complex

  - If complete timing analysis verifies adequate timing margin exist, risk is completely mitigated

- Modify selected portions of design having highest risk – partially mitigates risk

- Perform voltage/temperature margin testing

  - Use small temperature steps (such as 1 degree centigrade) since timing problem can occur over a narrow temperature range

  - Risk is only partially mitigated if test is successful

- Prayer- may be needed if risk is not completely mitigated

Acknowledgements

- A spacecraft instrument had many problems during system testing due to marginal asynchronous logic in FPGAs

  - The corrective action was to redesign all FPGAs to make them synchronous

  - Significant additional costs and schedule delays resulted

- A spacecraft experienced inadvertent resets during system testing.

  - The reset problem occurred over the 17 to 20 degree centigrade temperature range

  - The problem was solved by redesigning FPGAs contained in the flight computer

- Problems were found while performing breadboard testing a mission critical spacecraft subassembly

  - The following potential problems were found as the result of a peer review of FPGA logic:

    ◆ The FPGA contained complex asynchronous having unknown timing margin

Space Hardware Problems Caused by Asynchronous FPGAs

♦ A potential race condition where the setup time to a flip-flop could not be guaranteed

♦ A decoded counter output was used as a clock (decoding glitches could result in inadvertent clocking)

– The findings of the review along with the following risk assessment were presented to project management:

♦ Potential mission ending failure impact

♦ Likelihood of failure is high

– Project management requested that the asynchronous logic be redesigned to make it synchronous

– This redesign was completed and worst case timing margin was verified to completely mitigate the design risk.

Recommendations

- Prepare a design requirements document for each FPGA-based logic design

  - Ensures that design requirements, including interface signals, are clearly defined

  - Can be reviewed and approved by all affected parties

  - Will help others, including the peer reviewer, understand the design requirements

- Design synchronous logic where possible

- Perform <u>complete</u> worst case timing analyses of <u>all</u> designs

- Conduct a peer review of each logic design by an expert logic designer

  - Potential problems can be identified

  - Even synchronous designs may have potential problems, such as:

    ♦ Potential problems while powering up FPGA

Recommendations

♦ Critical functions affected by single event upsets

♦ Potential initialization problems

♦ Potential lockup problems

♦ Sneak paths through unpowered logic

♦ Potential timing problems at FPGA to FPGA interfaces

● Assess the risk of all asynchronous logic designs which have not been verified to meet worst case timing requirements and submit to project risk management system

● Mitigate asynchronous logic design risks, if feasible

Summary

- Asynchronous logic designs can result in risks to space missions

- Use asynchronous logic only where required, but perform complete worst case timing analyses

- Assess the risk of all asynchronous logic designs which have not been verified to meet worst case timing requirements

- Mitigate risks where feasible

Acknowledgements

- This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

- Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States government or the Jet Propulsion Laboratory, California Institute of Technology.