Fundamentals of Digital Engineering:

Designing for Reliability

A Micro-Course

R. Katz
[redacted]
Design Engineer
May 21, 2001

Abstract

The concept of designing for reliability will be introduced along with a brief overview of reliability, redundancy and traditional methods of fault tolerance is presented, as applied to current logic devices. The fundamentals of advanced circuit design and analysis techniques will be the primary focus. The introduction will cover the definitions of key device parameters and how analysis is used to prove circuit correctness. Basic design techniques such as synchronous vs. asynchronous design, metastable state resolution time/arbiter design, and finite state machine structure/implementation will be reviewed. Advanced topics will be explored such as skew-tolerant circuit design, the use of triple-modular redundancy and circuit hazards, device transients and preventative circuit design, lock-up states in finite state machines generated by logic synthesizers, device transient characteristics, radiation mitigation techniques, worst-case analysis, the use of timing analyzers and simulators, and others. Case studies and lessons learned from spaceflight designs will be given as examples.

Introduction

This Seminar

- This is a seminar, not a class
  - Two Way Conversation
  - Basic Theory
  - Lessons Learned
  - Case Studies for Discussion
    - Present Your Own Case Studies for Discussion and Future Inclusion
- Under Development
  - First Time This Seminar Is Given
  - Not All Topics Are Fully Developed
  - What Areas Are Useful? Guide Development.

Reliability Motivation - A Case Study (1961)

First, I believe that this nation should commit itself to achieving the goal, before this decade is out, of landing a man on the moon and returning him safely to the earth.

Special Message to the Congress on Urgent National Needs
President John F. Kennedy
Delivered in person before a joint session of Congress
May 25, 1961

Reliability Motivation - A Case Study (1986)

It appears that there are enormous differences of opinion as to the probability of a failure with loss of vehicle and of human life. The estimates range from roughly 1 in 100 to 1 in 100,000. The higher figures come from the working engineers, and the very low figures from management. What are the causes and consequences of this lack of agreement? Since 1 part in 100,000 would imply that one could put a Shuttle up each day for 300 years expecting to lose only one, we could properly ask "What is the cause of management's fantastic faith in the machinery?"

R. P. Feynman, Report of the PRESIDENTIAL COMMISSION on the Space Shuttle Challenger Accident, Volume 2: Appendix F - Personal Observations on Reliability of Shuttle, June 6th, 1986

Reliability
Motivation - A Case Study (2001)

When discussing the impact of the high observed FIT rate for the FPGAs, the IAT asked Lockheed Martin "What's the reliability allocation?" Lockheed Martin responded, "Hell if I know."

The IAT followed up by stating that it appeared that there has been no calculation of the probability of mission success. Lockheed Martin concurred and JPL added: "No programmatic requirement for reliability numbers."

From the Mars Odyssey FPGA Independent Assessment Team, April 2, 2001

Increasing Reliability

- Fault Prevention
  - Eliminate Faults
  - In Practice, Reduce Probability of Failure to an Acceptable Level
- Fault Tolerance
  - Faults Are Expected
  - Use Redundancy
    - Additional Hardware, Software, Time

Conventional Techniques for High-Reliable Spaceborne Digital Systems

- Use of Conservative Design Practices
  - Derating, Simplicity, Wide Tolerances
- Parts Standardization
- 100% Screening of Parts and Assemblies, Including Thorough Burn-in
- Detailed Laboratory Analyses and Corrective Action for All Failed Parts
- Use of Extreme Care in Manufacture of Parts
- Thorough Qualification of Parts and Manufacturing Processes

Conventional Techniques for High-Reliable Spaceborne Digital Systems (cont'd)

- Thermal Cycling and Vibration Testing of All Completed Assemblies
- Establishment of an efficient field service feedback system to report on equipment failures in the Field
- Design of the Equipment to Minimize Stress During Assembly and to Facilitate Replacement of Failed Components

NASA SPACE VEHICLE DESIGN CRITERIA (GUIDANCE AND CONTROL)
SPACEBORNE DIGITAL COMPUTER SYSTEMS - SP-8070
MARCH 1971

What We Will Do

- Cover Basic Concepts
- Present Data and Design Techniques
- Case Studies
  - Solutions for Previous Missions
  - Mistakes from Previous Missions

What We Will Not Do

- Provide Exhaustive Coverage
  - We only have a few hours
  - Too much material
- Solve All Problems
  - Goal is to make you think
- Not discuss "Mom and Apple Pie" [well, at least minimize it]

The Lessons of Designing for Reliability

"... we must not repeat the errors of the past. This is blocking and tackling, not rocket science."
Dan Goldin, April 27, 2000.

Barto's Law: Every circuit is considered guilty until proven innocent.

Special Pins

A Very Basic Topic But A Source of Frequent Failures and Problems

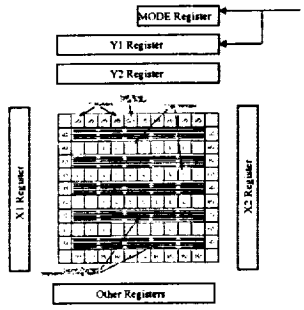
Termination of Special Pins

- MODE pin (test program mode).
- V_{PP} pin (programming voltage).
- TRST* (Reset to JTAG TAP controller)
- TCLK (provides clock to TAP controller)
- SDI, DCLK (varies for each device type)
- Others

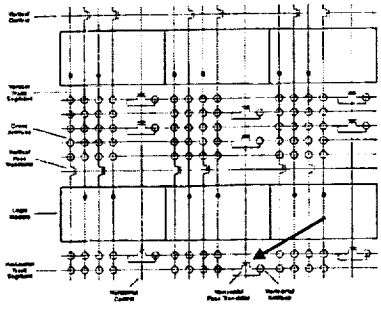
MODE Pin

- Left Floating
  - Device can be non-functional
  - High currents
  - Uncontrolled I/O
- Tied High During Test
  - Working device stopped functioning
  - Power supply rise time key

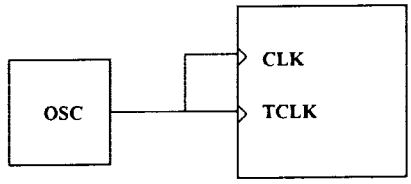
MODE Pin - Test, Debug and Programming Control



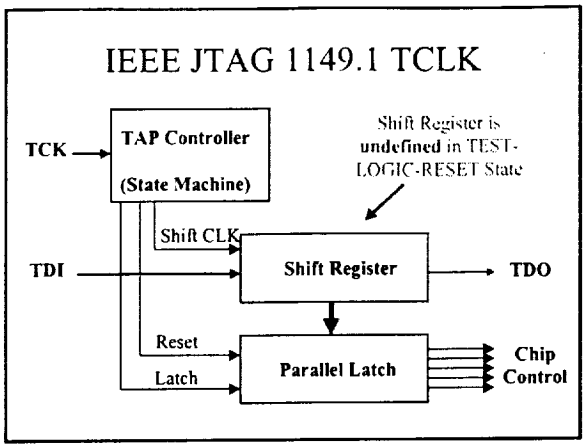
MODE Pin - Test, Debug and Programming Control

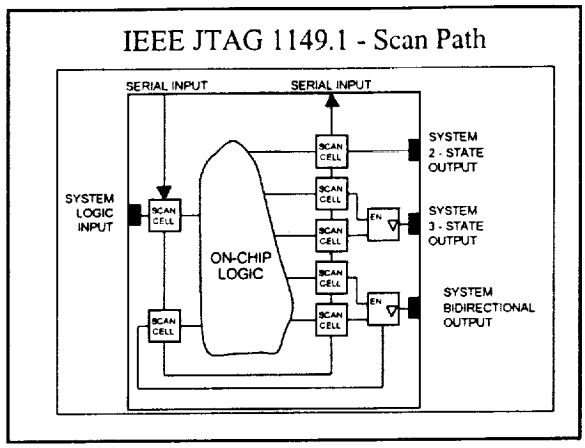


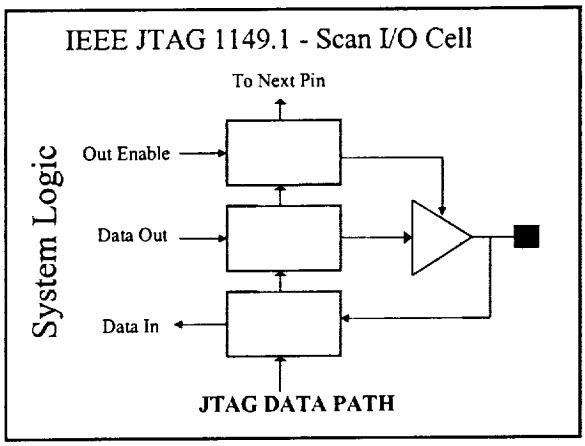
IEEE JTAG 1149.1 TCLK



The CLK pin may turn into an output driving low, clamping the oscillator's output at a logic '0'. The TAP controller can not reset and restore I/O operation. Most FPGAs do not have the optional TRST* pin. Note TRST*, when present, has a pull-up.


IEEE JTAG 1149.1 TCLK
TCK
TAP Controller
(State Machine)
Shift Register is
undefined in TEST-
LOGIC-RESET State
Shift CLK
TDI
Shift Register
TDO
Reset
Latch
Parallel Latch
Chip
Control


IEEE JTAG 1149.1 - Scan Path
SERIAL INPUT
SERIAL INPUT
SYSTEM
LOGIC
INPUT
SCAN
CELL
ON-CHIP
LOGIC
SYSTEM
2 - STATE
OUTPUT
SYSTEM
3 - STATE
OUTPUT
SYSTEM
BIDIRECTIONAL
OUTPUT
EN


IEEE JTAG 1149.1 - Scan I/O Cell
To Next Pin
System Logic
Out Enable
Data Out
Data In
JTAG DATA PATH

Input Stages

Input Stages - Introduction

- Most CMOS inputs have rise/fall time limits
  - Most inputs also have some hysteresis
- Typical symbols in specifications
  - t_R, t_{TLH} - rise time
  - t_F, t_{THL} - fall time
  - t_T - transition time
- Waveform measurement
  - typically from 10% to 90% but not always
  - sometime parameter measurement method is not specified

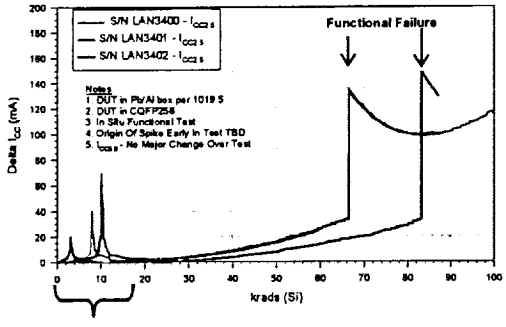
Input Stages - Practice

- Data sheets may list a parameter for information only and not 100% tested
- Laboratory devices have shown that not all qualified devices will meet the data sheet
  - One case was when a part was shrunk
  - Migration to a faster process
  - Oscillations observed
- Conservative margins recommended

Input Stages - Termination

- Floating CMOS inputs are, in general, 'bad.'
  - Totem-pole currents, oscillations, etc.
- Some devices offer pull-up/down resistors
  - SX-S only active during power transitions
  - Xilinx resistors controlled by SRAM
  - Care on internal tri-state lines
- Dedicated Inputs
  - Actel unused inputs were handled by s/w
  - Not true for some SX, SX-S clocks
  - ⇒ Check each case carefully

Input Stages - Termination
Case Study: SX-S Clock Pin



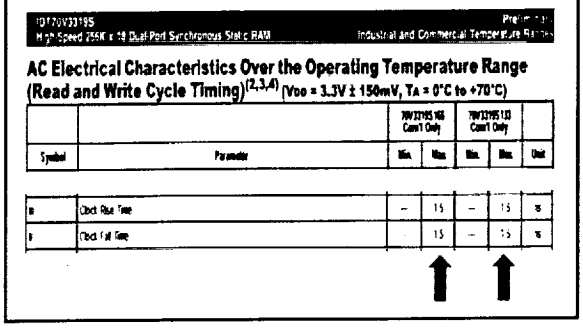
Input Transition Times

Part Number	Reference	t_T max (ns)
A1020	1	500
A1020A	2	500
A1020B	3	500
RH1020	4	500
A1280	2	500
A1280A	4	500
RH1280	4	500
Act 3 - 0.8 µm (5V)	-	500?
Act 3 - 0.8 µm (3.3)	6	500
RT54SX16, 32	7	50
A54SX-A (32, 72)	8	10
RT54SXS	9	10
XQR4000XL	10	250
Virtex	11	250
UT22VP10	12	?
AT6010 (MIL)	13	50
AT6010 (3.3V)	14	50
Quicklogic	-	?

Input Transition Times
References

[1] ACT ™ 1 Field Programmable Gate Arrays, March 1991.
[2] ACT 1 and ACT 2 Military FPGAs, April, 1992.
[3] ACT ™ 1 Series FPGAs, April 1996.
[4] Radiation Hardened FPGAs, v3.0, January 2000.
[5] ACT ™ 2 Series FPGAs, April 1996.
[6] Accelerator Series FPGAs – ACT ™ 3 Family, September, 1997.
[7] 54SX Family FPGAs RadTolerant and HiRel, Preliminary V1.5, March 2000.
[8] HiRel SX-A Family FPGAs, Advanced v.1, April 2000.
[9] RT54SX-S RadTolerant FPGAs for Space Applications, Advanced 0.2, November, 2000.
[10] QPRO XQR4000XL Radiation Hardened FPGAs, DS071 (v1.1) June 25, 2000.
[11] QPRO™ Virtex™ 2.5V Radiation Hardened FPGAs, DS028 (v1.0) April 25, 2000 Advance Product Specification.
[12] Not in data sheet.
[13] Configurable Logic Data Book, Atmel, August 1995.
[14] AT6000LV, Atmel, October 1999.

Clock Transition Time Specification
A Difficult Case



IDT70V33/19S
High Speed 256K x 18 Dual-Port Synchronous Static RAM
Industrial and Commercial Temperature Ranges

AC Electrical Characteristics Over the Operating Temperature Range (Read and Write Cycle Timing)[2,3,4] (VDD = 3.3V ± 150mV, TA = 0°C to +70°C)

Symbol	Parameter	70V33/19S166 Com'l Only Min.	70V33/19S166 Com'l Only Max.	70V33/19S133 Com'l Only Min.	70V33/19S133 Com'l Only Max.	Unit
tR	Clock Rise Time	–	1.5	–	1.5	ns
tF	Clock Fall Time		1.5	–	1.5	ns

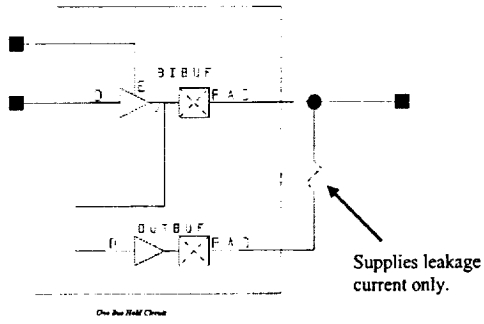
Transition Time Requirements
Implications - Pullup Resistors

- Often used for tri-state or bi-directional busses
- Rise time (10% - 90%) = τ = 2.2 RC
- Example
  - C = 50 pF
  - R = 10 kΩ (keep power levels reasonable)
  - τ = 500 ns
  - ⇒ violates many devices' specifications (see table)

Transition Time Requirements
Implications - Filters and Protection Circuits

- Often used on signals
  - Elimination of noise
  - ESD protection
  - Etc.
- RC filters or clamps (high C) can often substantially degrade transition times
- Consider discrete hysteresis buffers, particularly for clock signals

Bus Hold Circuit in an FPGA



Transition Time Requirements
Implications - Interfacing with older logic families

- Case Study (1)
  - CD4000B CMOS NOR gate
  - V_{DD} = 5V
  - t_T (typ) = 100 ns
- Case Study (2)
  - CD4050B (used as a level shifter, for example)
  - V_{DD} = 5V
  - t_T (max, 25 °C) = 160 ns

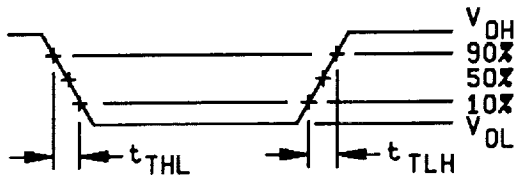
Transition Time Requirements
Implications - Interfacing with older logic families (cont'd)

- Case Study (3) - 54HC00 CMOS NOR gate
  - 5962-8403701VDA, NAND GATE, QUAD 2-INPUT

Test	Symbol	Test conditions 1/ $-55°C \le T_C \le +125°C$ unless otherwise specified		Limits Min	Limits Max	Unit
Transition time, output rise and fall	t_{THL}, t_{TLH} 3/	T_C = +25°C C_L = 50 pF See figure 4	V_{CC}=2.0 V_{CC}=4.5 V_{CC}=6.0		75 15 13	ns
		T_C = -55°C, -55°C C_L = 50 pF See figure 4	V_{CC}=2.0 V_{CC}=4.5 V_{CC}=6.0		110 22 19	ns

3/ Transition time (t_{TLH}, t_{THL}), if not tested, shall be guaranteed to the specified limits in table 1.

Transition Time Requirements
Parameter Measurement

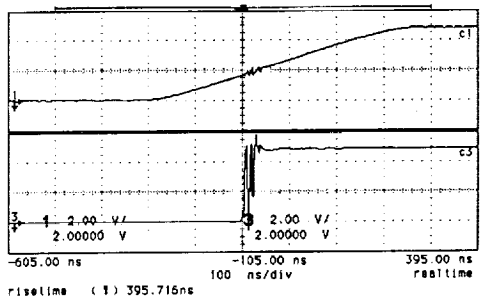


From: Figure 4, 5962-8403701VDA, NAND GATE, QUAD 2-INPUT

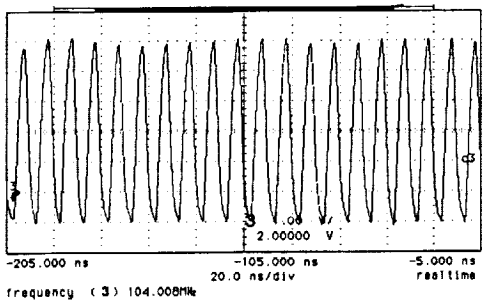
Transition Time Requirements
Case Study: RH1020

- Production Parts
  - Input stage was modified for clock upset
- V_{CC} = +5VDC
- T=25°C
- CLKBUF monitored on output
  - Because of design of the buffer, difficult to see effects on the input pin
- Used a low-impedance signal generator, triangle waveform
- Commercial specification is t_R, t_F of 500 ns
  - RH1020 did not meet this specification
  - SMD 5962-90965 does not specify this parameter

Transition Time Requirements
Case Study: RH1020 CLKBUF



Transition Time Requirements
RH1020 CLKBUF @ V_{IN} threshold



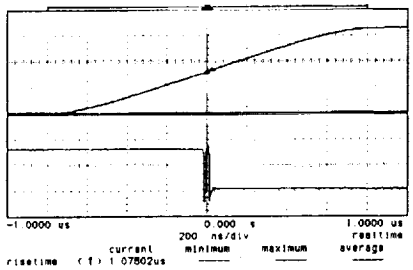
Transition Time Requirements
Case Study: RH1020 CLKBUF Notes

- Conditions: Room temp; V_{CC} = 5.0 V.
- Oscillations detected consistently at t_R = 360 ns
- Sporadic output pulses at t_R = 300 ns
- Transition time requirement not symmetric
  - Oscillations detected consistently at t_F = 1.5 µs
  - Sporadic output pulses observed at t_F = 1.0 µs

Interfacing - Voltage Margin

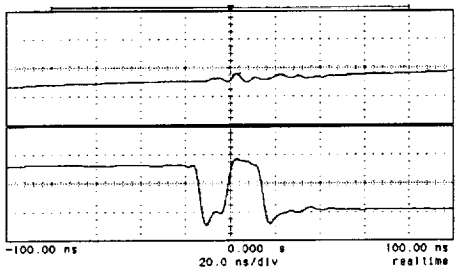
- TTL → CMOS
  - Problem with discrete circuits (still seen)
  - Normally not a problem with 5V FPGAs
  - Issue with new FPGAs
    - 0.35 µm may only pull up to 3.3 VDC
    - 0.25 µm may only pull up to 2.5 VDC
    - Can be issue with parts having a V_{IH} = 70% V_{DD}
    - Ringing can cause false triggering
- V_{IL} = 0.8V and fast devices are sensitive to ringing on a backplane.

Inputs: RT54SX16 t_T



RT54SX16 output (bottom trace) with a slow rising input (top trace) which clocks a divide by two counter resulting in a "glitch." The clock input was provided by an HP8110A pulse generator.

Inputs: RT54SX16 t_T



RT54SX16 output (bottom trace) with a slow rising input (top trace) which clocks a divide by two counter resulting in a "glitch." The clock input was provided by an HP8110A pulse generator.

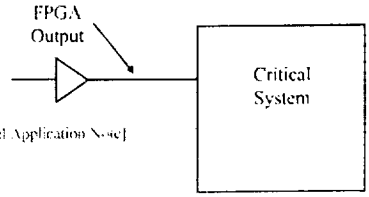
JTAG and Loss of Control

- Run TCK with TMS='1'
  - Guaranteed to return to TEST_LOGIC_RESET state within 5 clocks.
- Share system clock with TCK
- JTAG Hit
- Inputs turn to outputs
  - Clock pin turns to output, clamps system clock

⇒ No TCK, system hangs.

Startup Transients

Start up Transient - Outputs

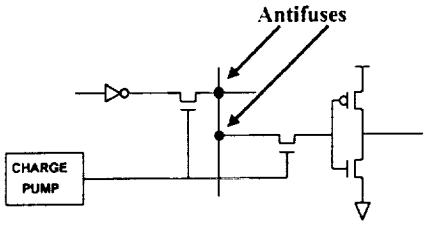


[from Actel Application Note]

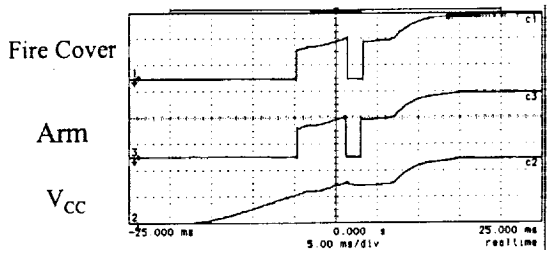
Power Up

Actel FPGAs are nonvolatile and therefore require no external configuration circuitry on power up. However, at power up it does take a finite amount of time for the device to become stable and operate normally. For a V_{CC} slew rate of ~30 ns/V, it takes approximately 250 ms for the device to become fully operational. Power up time varies with temperature, where cold is worst case. At power up, the state of all flip-flops is undefined. Some new designs will be power up safe.

Start up Transient
Charge Pump and Isolation

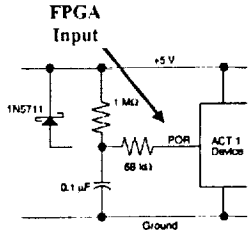


Start up Transient - Outputs



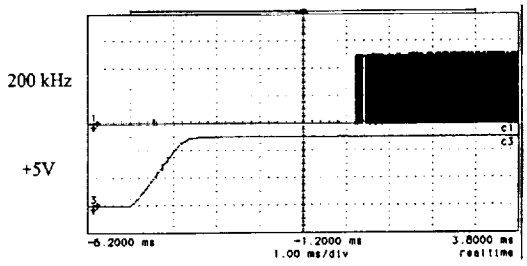
Hor: 5 ms/Division; Ver: 2 volts/Div

Start up Transient - Inputs



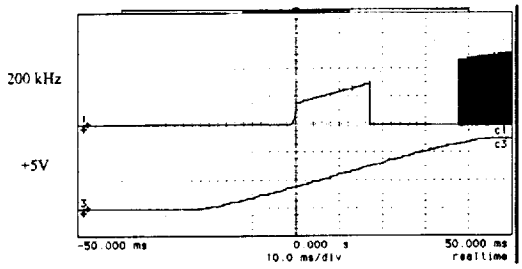
During the start up time with many FPGA models, an input may source current. In this application, a buffer with Schmidt trigger inputs is recommended.

Flight Oscillator Start Time



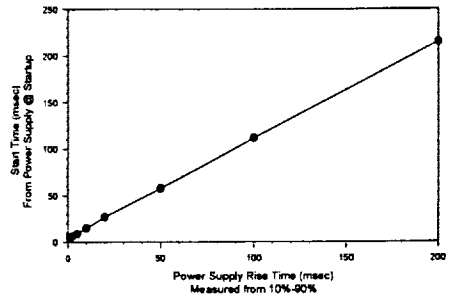
1 ms/div; t_{RISE} = 1 ms

Flight Oscillator Start Time

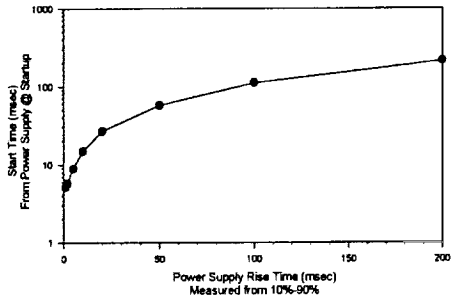


10 ms/div; t_{RISE} = 50 ms

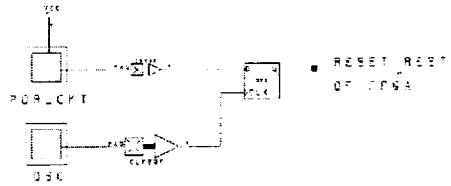
Flight Oscillator Start Time Summary



Flight Oscillator Start Time Summary

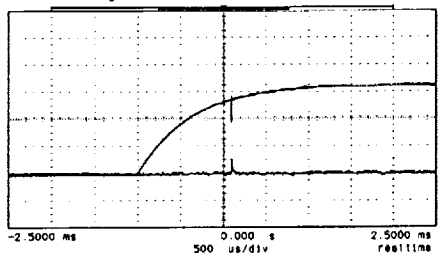


Synchronous Reset



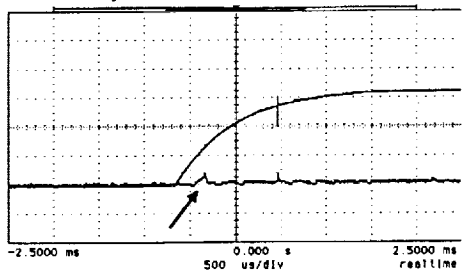
- FPGA may not be functional during power-on transient
- Crystal oscillator start time

Startup Current Transient
Case Study: RT54SX32 Pre-Irradiation



Startup current transient (3.3V supply) of an RT54SX32 pre-irradiation. Voltage at 1V/Div and current at 100mA/Div.

Startup Current Transient
Case Study: RT54SX32 Post-Irradiation



Startup current transient (3.3V supply) of an RT54SX32 after 98 krad (Si). Voltage at 1V/Div and current at 100 mA/Div.

Startup Current Transient
Xilinx Technology

- Two sets of requirements for the power-on transient for Xilinx XQR4000XL and Virtex 2.5V FPGAs.
  - Rise time
  - Current capability of the power supply.
- Noted that unlike Actel FPGAs where slower power supply rise times result in higher current values, in Xilinx devices, *faster* rise times result in higher current values.

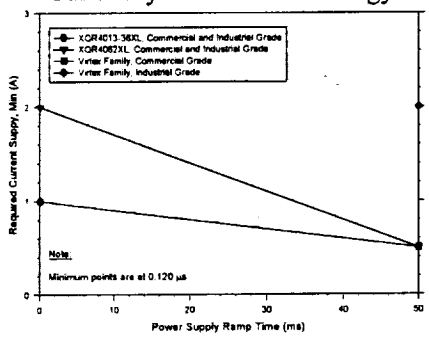
Startup Current Transient
Xilinx XQR4000XL

- Rise Time
  - Slowest power supply rise time is 50 ms. Many power supplies can meet this specification easily
  - Some spaceborne power supplies may have longer rise times.
- Current Levels
  - The minimum current is broken into two groups: XQR4013-36XL and the XQR4062XL. Note that according to the specification, the values refer to commercial and industrial grade products only, with the transition measured from 0 VDC to 3.6 VDC. *Actual* currents may be higher than the minimums specified.
  - Note 3 in the specification states that the duration of the peak current level will be less than 3 ms

Startup Current Transient
Xilinx Virtex

- Complete power supply requirements are not yet specified in the radiation hard data sheet. Some of the information is taken from the commercial data sheet.
- Rise Time
  - Slowest power supply rise time for this series of parts is 50 ms.
  - The fastest suggested ramp rate is 2 ms.
    - May be slow for some power supplies. The parameter measurement criteria on the radiation hard data sheet is from 1 VDC to 2.375 VDC.
- Current Levels
  - The data sheet only specifies a minimum required current supply for Virtex devices at a power supply rise time of 50 ms.
  - According to the non-military specification, it is 500 mA for commercial grade devices and 2 A for industrial grade parts.
  - Additionally, shorter power supply rise times will result in higher currents.
  - The duration of peak currents will be less than 3 ms.

Startup Current Transient
Summary: Xilinx Technology



I_{CC} Start-Up Transient Study in the RT1280A

An examination of the effects of radiation, a detailed look at the response of the part, annealing, and impacts to the board-level and system designs.

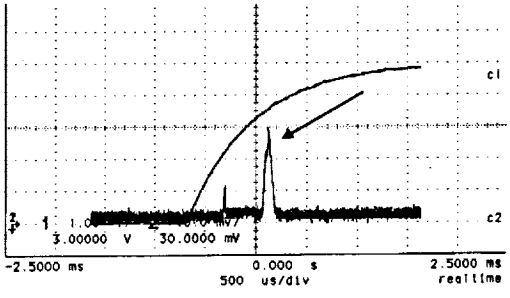
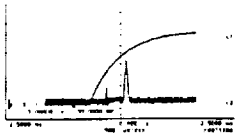


Figure 1. Startup transient after 4 krad (Si) exposure at 1 krad (Si)/day. The left current peak is unchanged from the pre-irradiation measurement and remained unchanged over the course of this experiment. Analysis on next slide.



- Startup transient after 4 krad (Si) exposure at 1 krad (Si)/day.
- Left current peak is unchanged from the pre-irradiation measurement and remained unchanged over the course of this experiment.
  - This current peak is expected as the NMOSFET isolate transistors are not fully conducting, resulting in totem pole currents in the input circuit of the logic modules.
- This current level or width is not specified in either the commercial or military specifications.
- The 350 mA current peak on the right appears when V_{CC} reaches 3.5 VDC.
- The power supply used for these tests had a rise time of < 2 msec.
- Voltage is at 1V/div; current is at 100 mA/div.

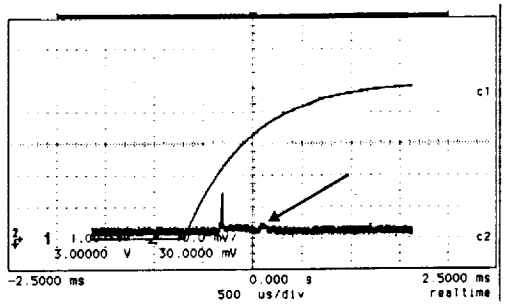


Figure 2. Startup transient after 5 days of room temperature, biased anneal, following the 4 krad (Si) irradiaton. The radiation-induced current peak is essentially gone. Voltage is at 1V/div; current is at 100 mA/div.

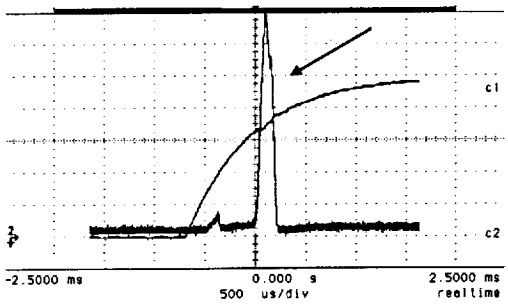
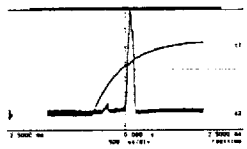
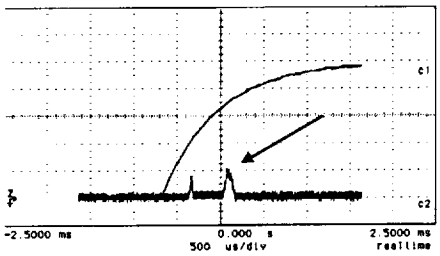


Figure 3. Startup transient after an additional 2 krad (Si) exposure at 1 krad (Si)/day for a total of 6 krads (Si). The radiation-induced current peak is now about 700 mA. Analysis on next slide.



- Startup transient after an additional 2 krad (Si) exposure at 1 krad (Si)/day for a total of 6 krads (Si).
- The radiation-induced current peak is now about 700 mA.
- The current draw still appears when V_{CC} reaches 3.5 VDC, unchanged from the 4 krad (Si) radiation step.
- At V_{CC}=3.5VDC, bulk capacitors on the board will have charge Q = 3.5V x C, which will provide charge in addition to that available from the power supply and helping to support the voltage rail. An 18 µF bulk capacitor will store 630 µC.
  - The current draw for this transient is approximately 100 µC.
- Voltage is at 1V/div; current is at 100 mA/div.



- Effects of 28-day, biased, room temperature anneal after the 6 krads (Si) irradiation step.
- The radiation-induced current peak is now reduced to about 100 mA.
- The current draw for this transient is approximately 12 µC, reduced from approximately 100 µC immediately after the 6 krads (Si) exposure.
- Voltage is at 1V/div; current is at 100 mA/div.

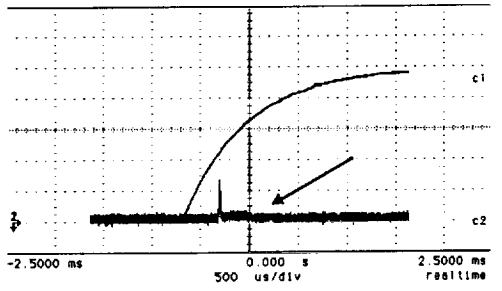
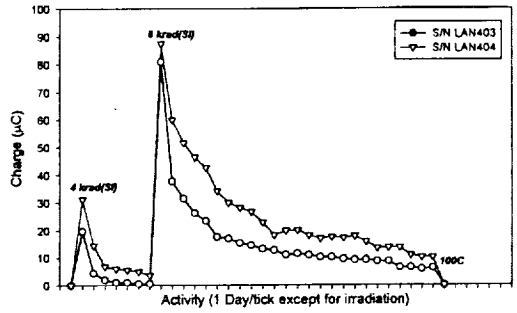
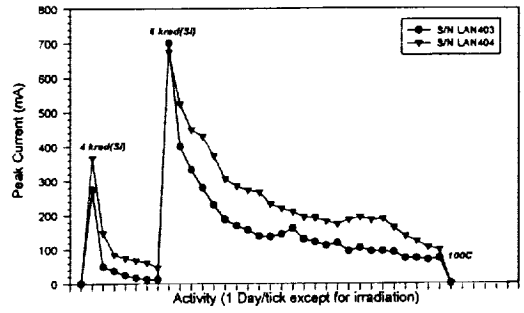


Figure 5. Effects of 100 °C, biased anneal after the 6 krads (Si) irradiation step and room temperature annealing. The radiation-induced startup current is now virtually eliminated, showing that annealing is effective. Voltage is at 1V/div; current is at 100 mA/div.

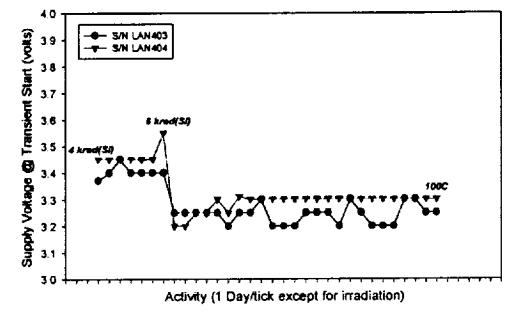
I_{CC} Startup Transient
RT1280A: Charge



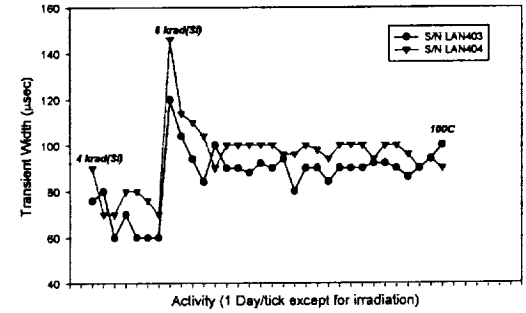
I_{CC} Startup Transient
RT1280A: Peak Current



I_{CC} Startup Transient
RT1280A: Trip Voltage



I_{CC} Startup Transient
RT1280A: Transient Pulse Width



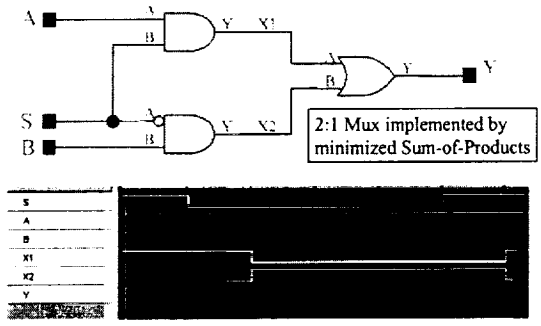
Static Hazards

Definitions

- If the change of a *single* variable causes a momentary change in other variables, which should not occur, then a *static hazard* is said to exist.
- If, after switching an input, the output has multiple transitions for a short time, then a *dynamic hazard* exists. For example
  - S/B: 0 → 1
  - IS: 0 → 1 → 0 → 1

Reference: Analysis and Design of Digital Circuits and Computer Systems, [illegible]

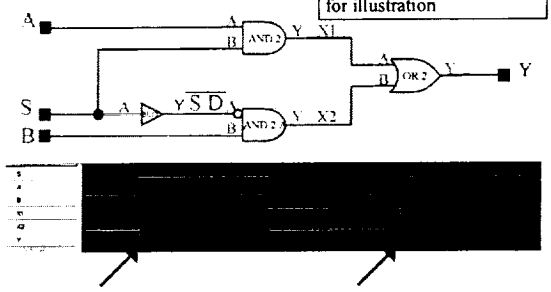
Static Hazard



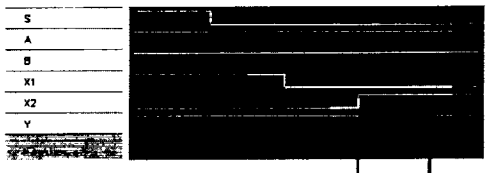
Idealized matched delays

Static Hazard

In real circuits, delays don't exactly match; Added delay for illustration



Static Hazard



We now have a "glitch."

Same waveform, zoomed in.

Static Hazard

	A B: 00	01	11	10
S=0	0	[1]	[1]	0
S=1	0	0	[1]	[1]

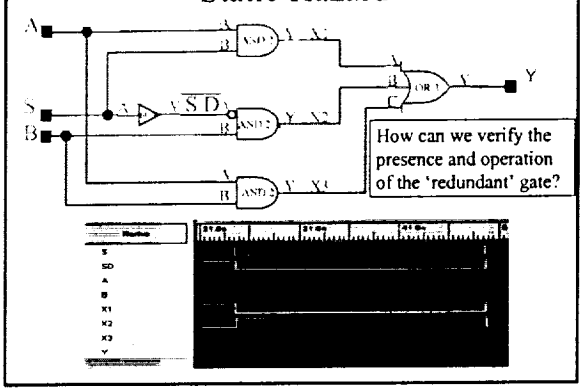
Illustrating the minimized function on a Karnaugh map. Only two 2-input AND gates are needed for the product terms

Static Hazard

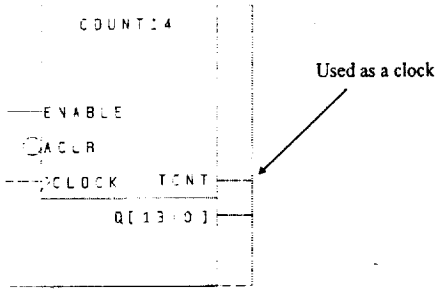
S \ AB	00	01	11	10
S=0	0	1	1	0
S=1	0	0	1	1

The blue oval shows the redundant term used to cover the transition between product terms.

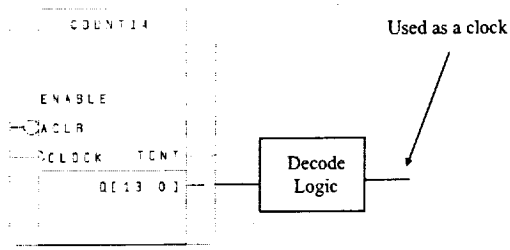
Static Hazard



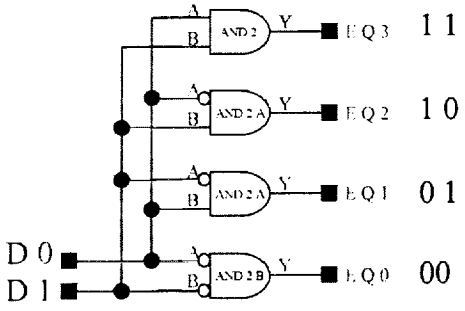
Asynchronous Decoding
High Level



Asynchronous Decoding
High Level - Another Form

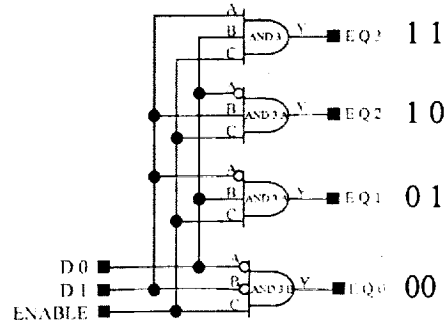


2:4 Decoder

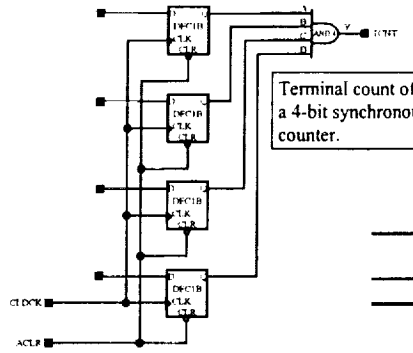


What happens when the inputs goes from 01 to 10?

2:4 Decoder with Enable



Implementation Level



Terminal count of a 4-bit synchronous counter.

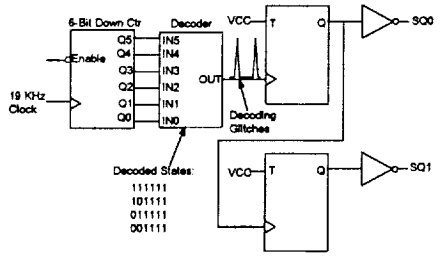
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15
0000	16

Asynchronous Decoding
Glitch Generation

```
01111111111111
10000000000000
...
11111111011111
11111111100000
```

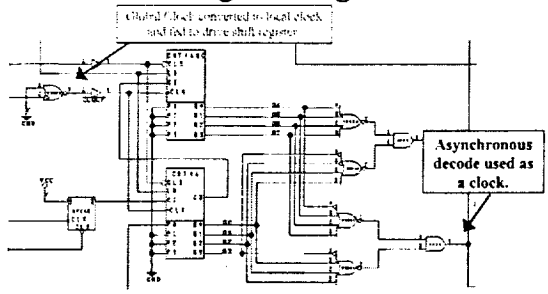
Because of unequal propagation delays, the sequence can momentarily go through state 1111111111111 generating a glitch.

Decoder Output Used As Clock



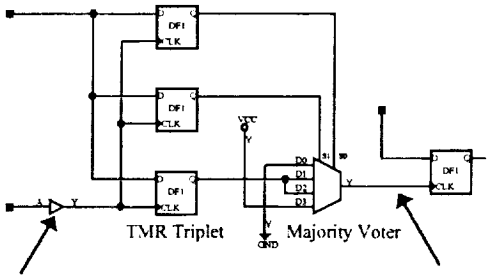
From Erickson, MAPLD 2000

Logic Design



Designer unaware that a parallel asynchronous decode may glitch. Relied on back-annotated logic simulation. This construct appears repeatedly.

Static Hazard
Flight Design Example



Static Hazard
Flight Design Example

Care is needed when using TMR circuits. First, the output of the voter may be susceptible to a logic hazard "glitch." This is not a problem if the TMR is feeding the input of another synchronous input. However, the TMR output should never feed asynchronous inputs such as flip-flop clocks, clears, sets, read/write inputs, etc.

"Design Techniques for Radiation-Hardened FPGAs" Actel Corporation, September 1997

Based on "SEU Hardening of Field Programmable Gate Arrays (FPGAs) for Space Applications and Device Characterization," R. Katz, R. Barto, et. al., IEEE Transactions on Nuclear Science, Dec. 1994.

Dynamic Hazards

We have covered static hazards. There are also dynamic hazards. An example of a dynamic hazard would be when a circuit is supposed to switch as follows:

$$0 \rightarrow 1$$

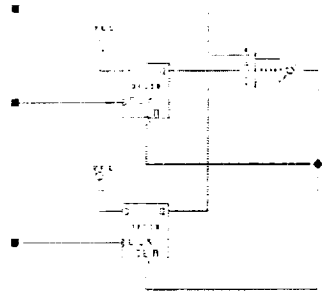
But instead switches:

$$0 \rightarrow 1 \rightarrow 0 \rightarrow 1$$

Any circuit that is static hazard free is also dynamic hazard free.

Asynchronous Logic

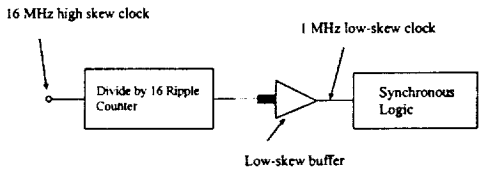
Asynchronous Clears



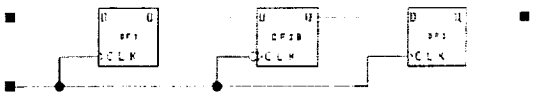
Synchronous vs. Asynchronous Logic

- Asynchronous signals are not synchronized to a clock.
- Timing Analysis for Asynchronous Circuits
  - Many tools do not support this
  - Complex, sometimes not tractable
  - Error-prone
- Asynchronous logic may result in smaller, faster, or lower power circuits
- Asynchronous logic, well done, is reliable.

Is It Or Isn't It?



Is It Or Isn't It?



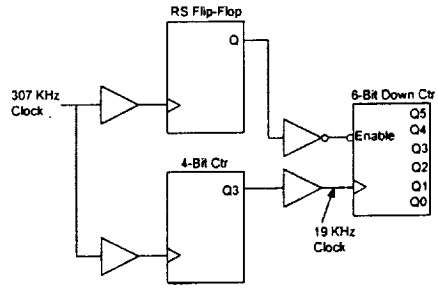
Common Asynchronous Design Problems

- Design may be marginal
  - Adequate margin non-verifiable
- Aging and radiation effects
  - Can not test for these
- Failures may occur late in the test program
  - i.e., thermal of thermal/vacuum testing
  - **This is always on Friday night**
- System may have unexplained glitches
  - Often difficult to troubleshoot

Some Examples of Problems

- Spacecraft Experienced Inadvertent Reset During System Testing
  - Only from 17 to 20 °C
  - FPGAs were redesigned
- Lots and lots of 'rookie mistakes.'
  - No analysis and unknown margin
  - Decoded outputs used as clocks
  - High-skew signals used as clocks
    - Counters
    - Shift Registers

Case Study
Potential Race Condition



Signal Quality

Transition Time (t_T) Compliance

DATE: 5-21-90 RADIATION TEST RESULTS SUMMARY

DEVICE TYPE AND DESCRIPTION: 54HCT8244 (SOS) OCTAL BUFFER DRIVER(ENGINEERING SAMPLES)
MANUFACTURER: HAR FACILITY: JPL ENERGY: 1.25MeV
LOG NUMBER: 1440 PACKAGE TYPE: 20 PIN DIP ETH: 445B
LOT NUMBER: 72010 TEST DATE: 3-13-90 DATE CODE: PB747
SAMPLE SIZE: 3 + CONTROL

DOSE LEVELS AND RATE, RAD(SI): 15K, 40K, 75K, 100K, 200K, 600K, 800K, AND 1000K RADS AT 100 RAD(SI)/SECOND
PIH IN HOURS: 1, 3, 24, 72, 168, AND 336 HOUR

COMMENTS:
1) IRRADIATION TEST CONDUCTED IN ACCORDANCE WITH MIL-STD-883 METHOD 1019.3.
4) BIAS DURING IRRADIATION: INPUTS HIGH, OUPUTS HIGH, TRI-STATED

PARAMETER	TEST CONDITIONS	FAIL LEVEL	COMMENTS
1) ICCH	SEE NOTE 3	40 KRADS(SI)	SPEC. LIMIT = 40 UA
2) ICCL	SEE NOTE 3	40 KRADS(SI)	SPEC. LIMIT = 40 UA
3) VOH1	SEE NOTE 3	800 KRADS(SI)	SPEC. LIMIT = 4.4 V
4) VOL1	SEE NOTE 3	800 KRADS(SI)	SPEC. LIMIT = 10 MV
5) IOZH	SEE NOTE 3	15 KRADS(SI)	SPEC. LIMIT = 1 UA
6) TPHL	SEE NOTE 3	75 KRADS(SI)	RAD. LIMIT = 26 NS
7) TPLH	SEE NOTE 3	168 HOUR PIH	SPEC. LIMIT = 17 NS
8) TTLH	SEE NOTE 3	15 KRADS(SI)	SPEC. LIMIT = 12 NS
9) TTHL	SEE NOTE 3	600 KRADS(SI)	RAD. LIMIT = 16 NS

From: http://radnet.jpl.nasa.gov/TID/1440.TXT

Transition Time (t_T) Compliance

01/20/92 PAGE 1 RADIATION AND PIH TEST RESULTS

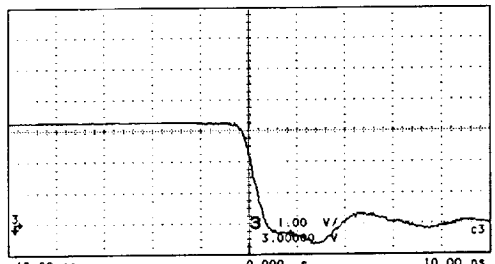
DEVICE TYPE: 54HCT244: OCTAL BUFFER LINE DRIVER
MANUFACTURER: TIX LOG NUMBER: 1492
PACKAGE TYPE: DIP RAD TEST REQ.: 445D-1
TEST DATE: 07/11/91 LOT NO.: N/A
DATE CODE: N/A FACILITY: Co60
SAMPLE SIZE: 3 + CONTROL

PARAMETER NAME	U NO. N PINS I /PARAM. T /DEVICE	FAILURE LEVELS Spec.	Rad.	Req.	PARAMETER LIMITS Specification *	Radiation #	Request #
TTLH	# #	1K		MAX	1.20-08	1.60-08	
TTHL	# #	INITI	INITI	MAX	1.20-08	1.60-08	

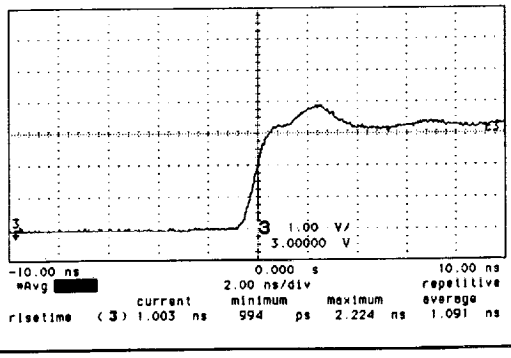
* = EXCEEDED VENDOR'S PARAMETRIC SPECIFICATION LIMIT.
# = EXCEEDED VENDOR'S RADIATION PARAMETRIC SPECIFICATION LIMIT.

http://radnet.jpl.nasa.gov/TID/1492.TXT

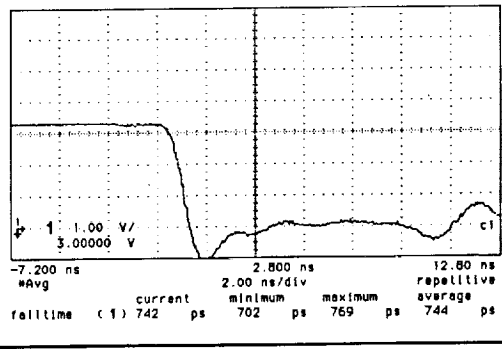
Transition Time (t_{THL})
High-Speed: RT54SX16



Transition Time (t_{TLH})
High-Speed: RT54SX16



Transition Time (t_{THL})
High-Speed: QL3025



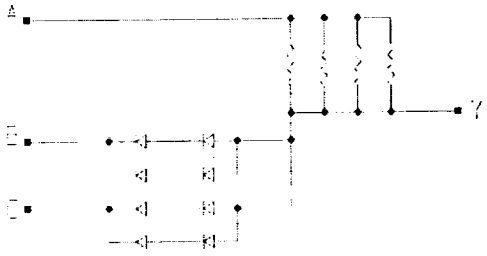
Fail Safe Logic

This section barely started, a lot of material to add.

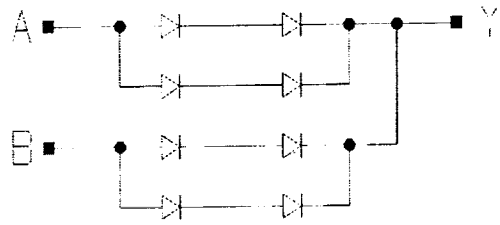
Orbiting Astronomical Observatory (OAO) Technology

"Primary Processor and Data Storage Equipment for the Orbiting Astronomical Observatory," Thomas B. Lewis, IBM Corporation, Space Guidance Center, Owego, NY. IEEE Transactions on Electronic Computers, December 1965, pp. 677-687

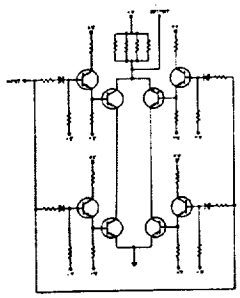
Quad Redundant AND Gate Orbiting Astronomical Observatory



Quad Redundant OR Gate Orbiting Astronomical Observatory



Quad Redundant Inverter Orbiting Astronomical Observatory

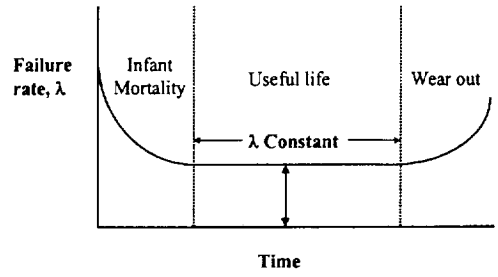


Reliability

Reliability

- Introduction to Reliability
- Historical Perspective
- Current Devices
- Trends

The Bathtub Curve



Introduction to Reliability

- Failure in time (FIT)
  Failures per 10^9 hours
  (~ 10^4 hours/year)
- Acceleration Factors
  - Temperature
  - Voltage

Introduction to Reliability (cont'd)

Most failure mechanisms can be modeled using the Arrhenius equation.

$$ttf = C \cdot e^{E_A/kT}$$

ttf - time to failure (hours)
C - constant (hours)
E_A - activation energy (eV)
k - Boltzman's constant (8.616×10^{-5}eV/°K)
T - temperature (°K)

Integrated Circuit Reliability
Historical Perspective

Application		Reliability
• Apollo Guidance Computer		< 10 FITs
• Commercial	(1971)	500 Hours
• Military	(1971)	2,000 Hours
• High Reliability	(1971)	10,000 Hours
• SSI/MSI/PROM 38510	(1976)	44-344 FITs
• MSI/LSI CICD Hi-Rel	(1987)	43 FITs

Actel FPGAs

Technology (µm)	FITS	# Failures	Device-Hours
2.0/1.2	33	2	9.4×10^7
1.0	9.0	6	6.1×10^8
0.8	10.9	1	1.9×10^8
0.6	4.9	0	1.9×10^8
0.45	12.6	0	7.3×10^7
0.35	19.3	0	4.8×10^7
RTSX 0.6	33.7	0	2.7×10^7
0.25	88.9	0	1.0×10^7
0.22	78.6	0	1.2×10^7

Xilinx FPGAs

- XC40xxXL
  - Static: 9 FIT, 60% UCL
  - Dynamic: 29 FIT, 60% UCL
- XCVxxx
  - Static: 34 FIT, 60% UCL
  - Dynamic: 443 FIT, 60% UCL

UTMC and Quicklogic

- FPGA
  - < 10 FITS (planned)
  - Quicklogic reports 12 FIT, 60% UCL
- UT22VP10
  UTER Technology, 0 failures, 0.3 [double check]
- Antifuse PROM
  - 64K: 19 FIT, 60% UCL
  - 256K: 76 FIT, 60% UCL

Actel FIT Rate Trends

Table 8-1 Fit Rates

Time Period	Q2,96	Q1,97	Q2,97	Q3/Q4, 97	Q1,98	Q2,98	Q3,98	Q4,98	Q1,99	Q2,99	Q3,99	Q4,99	Q1,00
1µ (FITS)	13.87	13.37	13.26	12.9	10.8	10.3	10.3	10.3	10.3	10.3	10.3	9.43	9.43
0.8µ (FITS)	20	19	19	18.5	18.6	16.6	15.8	15.9	16.5	15.9	15.9	15.9	14.77
0.6µ (FITS)	18	12.11	10.87	5.75	5.88	5.36	5.36	5.36	5.36	5.36	5.36	5.01	4.86
0.45µ (FITS)						90.4	70	25	25	25	23.53	23.53	23.53	16.38
0.35µ (FITS)								39	39	39	39	29.2	29.2	22.51
RTSX 0.6µ (FITS)									145	145	74.6	74.7	74.7	36.18
0.25µ (FITS)									145	145	74.6	74.7	74.7	112.33

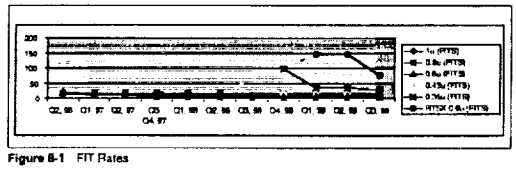


Figure 8-1 FIT Rates

Power Switching

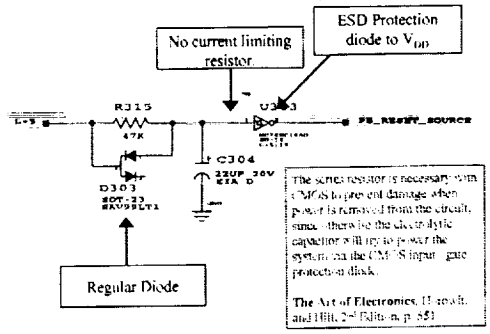
Power Supply Sequencing

- Protecting I/O's
- Powering Circuits
- RT54SX16/32
  - Perhaps RT54SX32S
  - UTMC buffers
- EEPROMs/write protection
- SMEX/WIRE

Power Supply Sequencing
Protecting I/O's

- Parasitic/ESD diodes
- PCI clamp diodes
- cold-sparing capable I/O's

Power-On Reset (POR)



Power Supply Sequencing
RT54SX16/32

Power-Up Sequencing

RT54SX16, AS4SX16, RT54SX32, AS4SX32

V_{CCA}	V_{CCI}	V_{OUT}	Power-Up Sequence	Comments
3.3V	5.0V	3.3V	5.0V First 3.3V Second	No possible damage to device.
			3.3V First 5.0V Second	Possible damage to device

Power-Down Sequencing

RT54SX16, AS4SX16, RT54SX32, AS4SX32

V_{CCA}	V_{CCI}	V_{OUT}	Power-Down Sequence	Comments
3.3V	5.0V	3.3V	5.0V First 3.3V Second	Possible damage to device
			3.3V First 5.0V Second	No possible damage to device.

54SX Family FPGAs, RadTolerant and HiRel, v 2.0, March 2001

Power Supply Sequencing
RT54SX32S

- To date, our lab work has shown, on some parts, that when V_{CCI} is applies before V_{CCA}, significant currents, > 10 mA, can be seen flowing into the V_{CCI} pin.
- Power supply sequencing may also affect reliability of the safe power on/off feature.
- These are under investigation.

Power Supply Sequencing
EEPROMs: Hardware Write Protection

3.11.5 **Power supply sequence of EEPROMs.** In order to reduce the probability of inadvertant writes, the following power supply sequences shall be observed.

a. For device types 1-18, a logic high state shall be applied to WE and/or CE at the same time or before the application of V_{cc}. For device types 16-18, an additional precaution is available, a logic low state shall be applied to RES at the same time or before the application of V_{cc}.

b. For device types 1-18, a logic high state shall be applied to WE and/or CE at the same time or before the removal of V_{cc}. For device types 16-18, an additional precaution is available, a logic low state shall be applied to RES at the same time or before the removal of V_{cc}.

Microcircuit, Memory, Digital, CMOS, 128K x 8-Bit EEPROM, Monolithic Silicon, SMD 5962- 38267, Revision. F, Dated. 6 October 1999

Power Supply Sequencing
EEPROMs: Software Write Protection

To protect against unintentional programming caused by noise generated by external circuits, AS58C1001 has a **Software data protection function**. To initiate Software data protection mode, 3 bytes of data must be input, followed by a dummy write cycle of any address and any data byte. This exact sequence switches the device into protection mode. This 4th cycle during write is required to initiate the SDP and physically writes the address and data. While in SDP the entire array is protected in which writes can only occur if the exact SDP sequence is re-executed or the unprotect sequence is executed.

The Software data protection mode can be cancelled by inputting the following 6 Bytes. This changes the AS58C1001 to the Non-Protection mode, for normal operation.

AS58C1001: 128K x 8 EEPROM, Austin Semiconductor, Inc

Power Supply Sequencing
EEPROMs: Software Write Protection

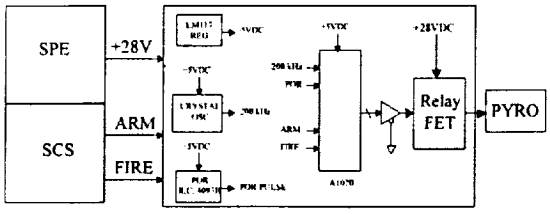
Enable Protection		Disable Protection	
Address	**Data**	**Address**	**Data**
5555	AA	5555	AA
2AAA	55	2AAA	55
5555	A0	5555	80
		5555	AA
		2AAA	55
		5555	20

AS58C1001: 128K x 8 EEPROM, Austin Semiconductor, Inc.

Power Supply Sequencing
SMEX/WIRE

- System applied power simultaneously to the FPGA, drive circuitry, and relay.
- Control FPGA generated both ARM and FIRE signals based on spacecraft opto-isolated inputs.
- Transient analysis not performed.
- Saved 1 relay.

Power Supply Sequencing
SMEX/WIRE



Redundancy

Definitions

- Simplex
  - Single Unit
- TMR or NMR
  - Three or n units with a voter
- TMR/Simplex
  - After the first failure, a good unit is switched out with the failed unit.
- TMR/Switchable Spare
  - After the second failure is detected, the last good unit is switched in.

Types of Redundancy

- Static Redundancy
- Dynamic Redundancy
- Hybrid Redundancy

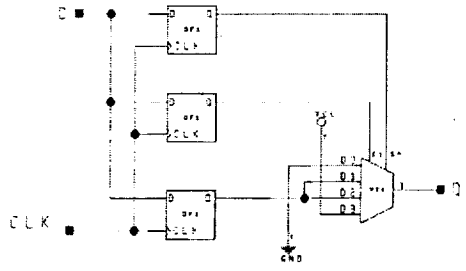
Static Redundancy

- Uses Extra Components
- Effect of a Fault is Masked Instantaneously
- Two Major Techniques
  - N-Modular Redundancy (generalization of TMR or Triple Modular Redundancy)
  - Error Correcting Codes

Static Redundancy

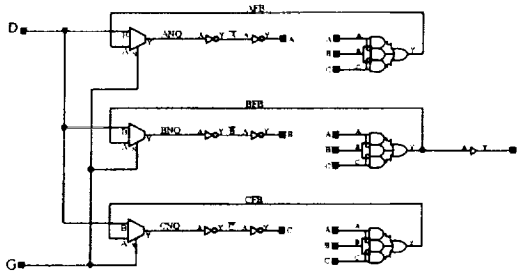
- TMR flip-flops
- What happens when you add a Hamming code and error correct to a finite state machine?
  - Hint: Are SEUs synchronous?

TMR/Voter Structures



With no active clock, it's an SEU integrator.

Static Redundancy Example
SEU-Hardened Flip-Flop



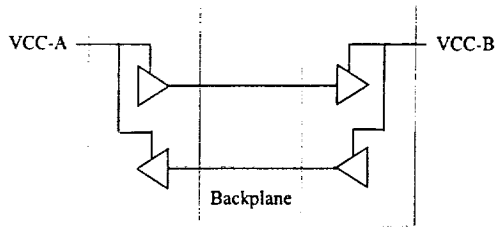
Dynamic Redundancy

- Uses Extra Components
- Only 1 Copy Operates At A Times
  - Fault Detection
  - Fault Recovery
- Spares Are On "Standby"
  - Hot Spares
  - Cold Spares

Hot and Cold Spares

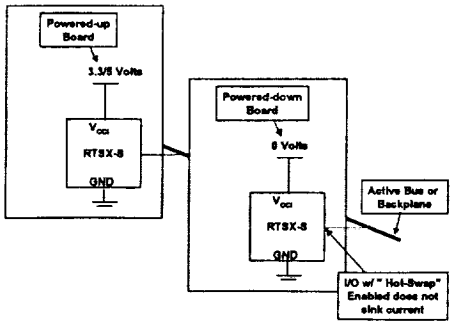
- Hot Spares
  - Modules/components are powered or 'hot'
- Cold Spares
  - Modules/components have their power removed or are 'cold'
  - Sneak path analysis is necessary, particularly with CMOS interfaces
    - Some CMOS I/O structures are high-impedance when powered down

Interfacing - Blocks



ESD and parasitic diodes (not shown here) to the power bus (present in most CMOS devices) form a sneak path.

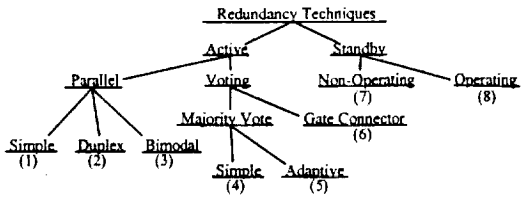
Cold Sparing - SX-S



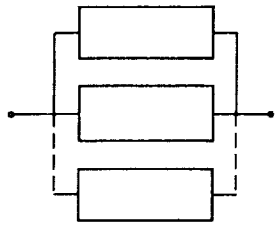
Types of Redundancy

- Classified on how the redundant elements are introduced into the circuit
- Choice of redundancy type is application specific
- Active or Static Redundancy
  - External components are not required to perform the function of detection, decision and switching when an element or path in the structure fails.
- Standby or Dynamic Redundancy
  - External elements are required to detect, make a decision and switch to another element or parth as a replacement for a failed element or path.

Redundancy Techniques

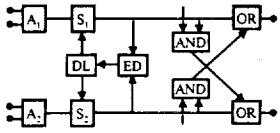


Simple Parallel Redundancy
Active - Type 1



In its simplest form, redundancy consists of a simple parallel combination of elements. If any element fails open, identical paths exist through parallel redundant elements.

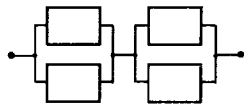
Duplex Parallel Redundancy
Active - Type 2



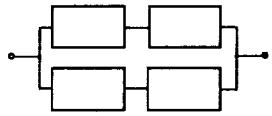
This technique is applied to redundant logic sections, such as A1 and A2 operating in parallel. It is primarily used in computer applications where A1 and A2 can be used in duplex or active redundant modes or as a separate element. An error detector at the output of each logic section detects noncoincident outputs and starts a diagnostic routine to determine and disable the faulty element.

Bimodal Parallel Redundancy
Active - Type 3

(a) Bimodal Parallel/ Series Redundancy

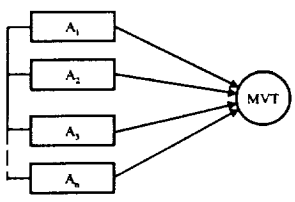


(b) Bimodal Series/ Parallel Redundancy



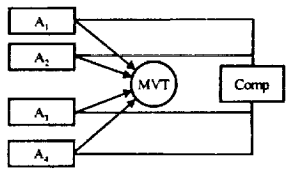
A series connection of parallel redundant elements provides protection against shorts and opens. Direct short across the network due to a single element shorting is prevented by a redundant element in series. An open across the network is prevented by the parallel element. Network (a) is useful when the primary element failure mode is open. Network (b) is useful when the primary element failure mode is short.

Simple Majority Voting
Active - Type 4



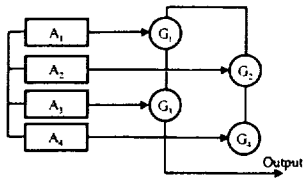
Decision can be built into the basic parallel redundant model by inputting signals from parallel elements into a voter to compare each signal with remaining signals. Valid decisions are made only if the number of useful elements exceeds the failed elements.

Adaptive Majority Voting
Active - Type 5



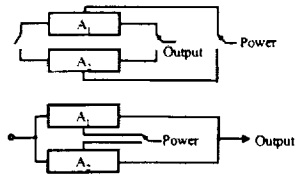
This technique exemplifies the majority logic configuration discussed previously with a comparator and switching network to switch out or inhibit failed redundant elements.

Gate Connector Voting
Active - Type 6



Similar to majority voting. Redundant elements are generally binary circuits. Outputs of the binary elements are fed to switch-like gates which perform the voting function. The gates contain no components whose failure would cause the redundant circuit to fail. Any failures in the gate connector act as though the binary element were at fault.

Non-Operating Redundancy
Standby - Type 7

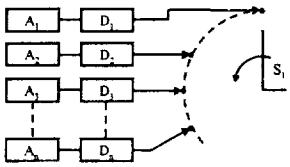


A particular redundant element of a parallel configuration can be switched into an active circuit by connecting outputs of each element to switch poles. Two switching configurations are possible.

1) The element may be isolated by the switch until switching is completed and power applied to the element in the switching operation.

2) All redundant elements are continuously connected to the circuit and a single redundant element activated by switching power to it.

Operating Redundancy
Standby - Type 8



In this application, all redundant units operate simultaneously. A sensor on each unit detects failures. When a unit fails, a switch at the output transfers to the next unit and remains there until failure.

Redundant Processors
Software Voting for the Space Shuttle

Killingbeck - There are approaches to the instability problem that involve equalization and periodic exchanges of data - some kind of averaging, middle select, or whatever, to keep things from getting too far apart. The problem is that, for every sensor, an analysis has to be made of what values are reasonable and how an average should be picked. The extra computation consumes a lot of manpower and time, and creates a lot of accuracy problems It's very hard to set a tolerance level that throws away bad data and doesn't somehow throw away some good data that happen to be extreme It wasn't so much that we felt that this scheme couldn't be made to work, it's just that we believe there had to be a better way.

Communications of the ACM, September 1984, p. 894

Redundant Processors
Architecture for the Space Shuttle

Killingbeck - We originally looked at three redundancy management schemes. First, we considered running as a number of totally independent sensor, computer, and actuator strings. This is a classic operating system for aircraft - the Boeing 767, for example, uses this basic approach. We also looked at the master/slave concept, where one computer is in charge of reading all the sensors and the other computers are in a listening mode, gathering information. One of the backups takes over only if the master fails. The third approach we considered is the one we decided to use, the distributed command approach, where all the computers get the same inputs and generate the same outputs.

Communications of the ACM, September 1984, p. 894.

Calculation of TMR Reliability for SEUs

The probability of i arrivals in a time t is calculated as:

$$P(i,t,\lambda) = \frac{(\lambda t)^i \times e^{-\lambda t}}{i!} \qquad (1)$$

Following this, the interarrival time is a continuously distributed exponential random variable with the average time between arrivals of 1/ .

Each particular bit is modeled independently of all other bits. In practice, this is not always true. For instance, certain memory devices may have multiple upsets in a single byte within one address [6]. This phenomena has not been seen in FPGAs.

Calculation of TMR Reliability for SEUs

The probability for a single bit not being upset can now be computed as the probability of an even number of arrivals in the scrub period and the probability for a bit being upset is computed as the probability of an odd number of arrivals.

PS = Probability of Success (2)
= Probability of no upset (3)
= Probability of an even number of upsets (4)
= $P(0,t,\lambda)+P(2,t,\lambda)+P(4,t,\lambda)+\ldots$ (5)

and

PF = Probability of Failure (6)
= Probability of upset (7)
= Probability of an odd number of upsets (8)
= $P(1,t,\lambda)+P(3,t,\lambda)+P(5,t,\lambda)+\ldots$ (9)

Calculation of TMR Reliability for SEUs

Now we have the following for each 'word' in memory:

1. The word consists of n (word length) "repeated" trials.
2. Success (no upset) or failure (upset).
3. Probability of success remains constant from bit to bit.
4. Each bit is independent.

which is a description of a binomial experiment.

The probability of a failure for an experiment is having more errors than the code can correct, which is either 2 or 3 for the TMR flip-flop.

Calculation of TMR Reliability for SEUs

So, P (Failure of a word) = $\sum_{i=2}^{n} P(i \text{ upsets in a word})$ (10)

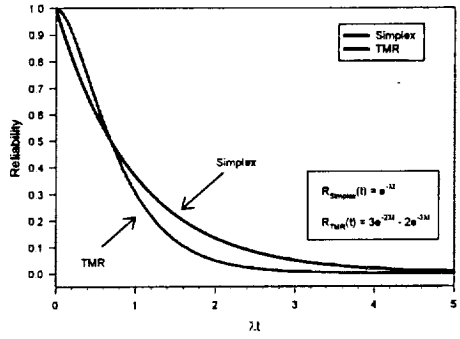
where n is equal to the total word length, and

$P(i \text{ upsets in a word}) = C(n,i)\times PS^{(n-i)}\times PF^{i}$ (11)

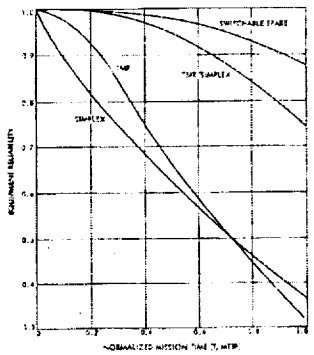
where $C(n,i)$ is defined as $\frac{n!}{i!\times(n-i)!}$ (12)

Once the probability of a word failing is calculated, multiplication by the number of words will give a failure rate.

Simplex vs. TMR Reliability



Reliability of Redundant Systems



NASA Space Vehicle Design Criteria (Guidance and Control)
Spaceborne Digital Computer Systems
NASA SP-8070
W. Hoffman, Aerospace Systems, Inc.
A. Perkins, Jr., SAFI
J. Green, Jr., Intermetrics, Inc.
March, 1971

Diverse Design

Diverse Design
Case Studies and Topics for Discussion

- Definition
- LEM abort computer
- Skylab Lessons Learned
- Space Station - ISS
- Software
- Shuttle Computers
- Small Satellites, University of Surrey

Diverse Design
Definition

In diverse design redundancy two or more components of different design furnish the same service.

This has two advantages: it offers high protection against failures due to design deficiencies, and it can offer lower cost if the back-up unit is a "life-boat," with lower accuracy and functionality, but still adequate for the minimum mission needs. The installation of diverse units usually adds to logistic cost because of additional test specifications, fixtures, and spare parts. This form of redundancy is, therefore, economical primarily where the back-up unit comes from a previous satellite design, or where there is experience with it from another source. Where there is concern about the design integrity of a primary component, diverse design redundancy may have to be employed regardless of cost.

Reducing Space Mission Cost, Wertz and Larson, p. 297

Diverse Design
Case Study - LEM Abort Guidance Computer

- Main computer
  - 15-bit AGC, common with the CSM
  - Single string
- Not enough resources for redundancy
- TRW produced a small computer
  - MARCO 4418
  - 8-bit
- Limited functionality
  - Put the LEM in lunar orbit

Computers Take Flight: A History of NASA's Pioneering Digital Fly-by-Wire Project, Earl Tomayko

Diverse Design
Skylab Lessons Learned

When designing redundancies into systems, consider the use of nonidentical approaches for backup, alternate, and redundant items.

Background

A fundamental design deficiency can exist in both the prime and backup system if they are identical. For example, the rate gyros in the Skylab attitude control system were completely redundant systems, i.e., six rate gyros were available, two in each axis. However, the heater elements on all gyros were identical and had the same failure mode. Thus, there was no true redundancy and a separate set of gyros had to be sent up on Skylab 4 for an in-flight replacement.

SKYLAB LESSONS LEARNED APPLICABLE TO A LARGE SPACE STATION, A dissertation submitted to the Faculty of the School of Engineering and Applied Science of The George Washington University of America for the Degree Doctor of Engineering by William C. Schneider, Washington, D.C., 1970.

Diverse Design
Case Study: Space Station

- No intentional diverse design, despite Skylab's lessons learned[1]. Very expensive.
- Overlap in functions between US and Russia provides some diversity in ISS.
- Russian side has some diversity more as a result of heritage then an objective.

[1]As far as I know.

Diverse Design
Topic for Discussion: Software

- Not widely applied in software
  - Difficult to quantify expected improvement
- N-version Programming
  - In hardware NMR, there are identical copies; in software NMR, independent coding.
  - Voted: Reference states "sufficiently similar."
- Limitation: 50% of faults in software control systems are in the specification

Fault Tolerant & Fault Testable Hardware Design, P. Lala, Syracuse University, pp. 104-106

Diverse Design
Software Voting

In the N-version programming approach a number of independently written programs for a given function are run simultaneously; results are obtained by voting upon the outputs from the individual programs. In general the requirement that the individual programs should provide identical outputs is extremely stringent. Therefore, in practice "sufficiently similar" output from each program is regarded as equivalent; however, this increases the complexity of the voters [4.54].

Fault Tolerant & Fault Testable Hardware Design, P. Lala, 1985, p. 105

Diverse Design
Case Study: Space Shuttle Computers

- Five Identical Sets of Computer Hardware
  - 4 run the primary software (PASS)
    - Each computer sees all I/O
    - Displays status to crew
  - 1 runs the Backup Flight System (BFS)
    - Runs during critical stages but does not control I/O unless engaged by the crew
  - Voting is done at the actuators (dynamic)
  - Crew provides decision making on switching redundancy (static)

Diverse Design
Case Study: Space Shuttle Computers

DG: How do you make the system reliable?

As I mentioned, there is a fifth computer that runs the Backup Flight System (BFS). Early on, NASA was concerned about the possibility of a generic software problem in the PASS: what if there were a "bug" in the PASS that brought the entire primary system down? The way they alleviated their fears was by developing independent ascent and entry software from a subset of the requirements they had given us. This independent software was written by Rockwell International and resides in the fifth computer.

The decision to engage the VGS is totally a crew function. Their procedures identify certain situations for which the switch should be made: for instance, loss of control, multiple consecutive failures of PASS computers, or the infamous two-on-two split where the computers split up into two pairs (we've never seen this occur). To date the crew has never had to use the BFS during a mission.

Diverse Design
Case Study: Space Shuttle Computers

Some more information on this is available from _Computers in Spaceflight - The NASA Experience_, James E. Tomayko, Wichita State University:

At first the backup flight system computer was not considered to be a permanent fixture. When safety level requirements were lowered, some IBM and NASA people expected the fifth computer to be removed after the Approach and Landing Test phase of the Shuttle program and certainly after the flight test phase (STS-1 through 4). However, the utility of the backup system as insurance against a generic software error in the primary system outweighed considerations of the savings in weight, power, and complexity to be made by [104] eliminating it.

[104] A.D. Aldrich, "A Sixth GPC On-Orbit," Memorandum, Johnson Space Center, Houston, TX, October 13, 1978, JSC History Office.

Diverse Design
Case Study: Small Satellites/Surrey

- Components: risk inherent in the use of components which are not formally "space qualified"
- New technologies: employed alongside flight-proven technologies in a "layered architecture"
  - Top-layer systems use state-of-the-art high-performance device types
  - Lower-layer systems use device-types which have been flown and tested in previous spacecraft, and which are able to carry out most of the same functions, albeit with a possible loss of performance.
- Layered architecture protects against design faults.

Diverse Design

Case Study: Small Satellites/Surrey

From the "Design Philosophy" section:

Recognising the risk inherent in the use of components which are not formally "space qualified", we use redundancy at many levels to reduce the risk of total mission failure. When adopting new technologies, we employ them alongside flight-proven technologies in order to reduce risk. Thus we build a "layered architecture", in which each successive layer relies on different systems comprising increasingly well-proven technologies. The top-layer systems use state-of-the-art high-performance device types - often without flight-heritage - but which give a high degree of functionality. Whereas the lower-layer systems use device-types which have been flown and tested in previous spacecraft, and which are able to carry out most of the same functions, albeit with a possible loss of performance. In this way, problems caused by an inherent system design fault, or by the failure of a particular device-type, are not duplicated in the different layers.

Configuration Control

This sounds boring and what is this topic doing in the middle of a design reliability seminar?

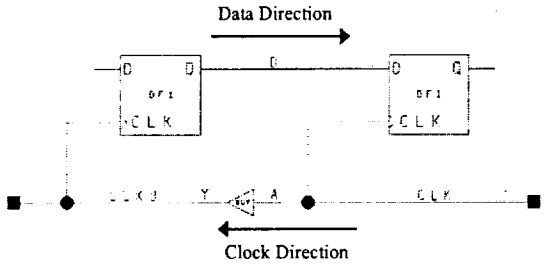
Configuration Control

Use of a "Standard" I/F Module

- Design team comprised of members from multiple organizations
- "Standard" module (Shift Register) intended to be used throughout the system.
  - Four different versions found in 11 FPGAs.
    - Two use "reverse buffering" for the clocks
    - Two use clock trees.

This programs design rules dictated that "reverse buffering" of clocks were to be used to control skew. Although that method can not guarantee performance, the rules were repeatedly violated.

"Reverse Buffering"



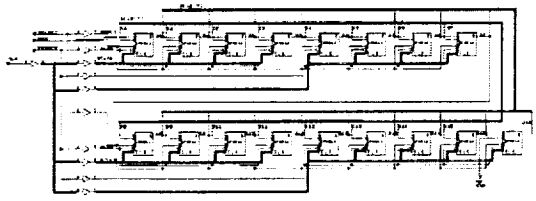
Configuration Control

Details on "Standard" Structure Usage.

Subsystem	FPGA	Version Used	Clock Buffer Tree Used?
A	A1	1	Yes
B	B1	2	
	B2	2	
	B3	3	
C	C1	2	
	C2	2	
D	D1	2	
	D2	2	
E	E1	2	
	E2	1	Yes
	E3	4	Yes

Configuration Control

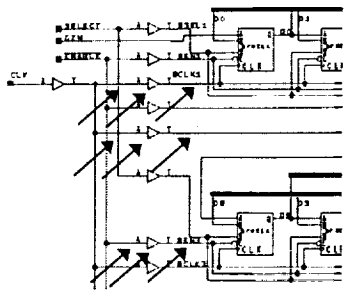
Sample Schematic



Violation of the Projects "reverse buffering" clock topology.

Configuration Control

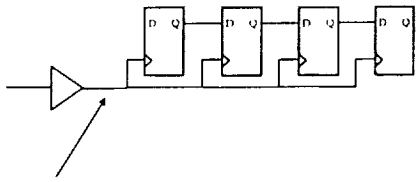
Sample Schematic - Further Detail



Sources of skew include routing between elements as well as the buffers in the tree. For Act 1 and Act 2 devices, routing and buffer delays can not be separated. Other considerations include rise time of the signal and the receivers threshold.

Clock Skew

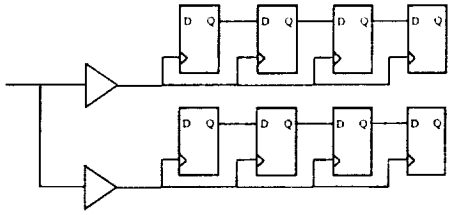
Clock Skew



Normal Routing Resource

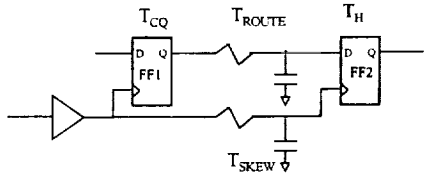
Shift register is given as an example. Also seen in counters and other logic structures.

Clock Skew



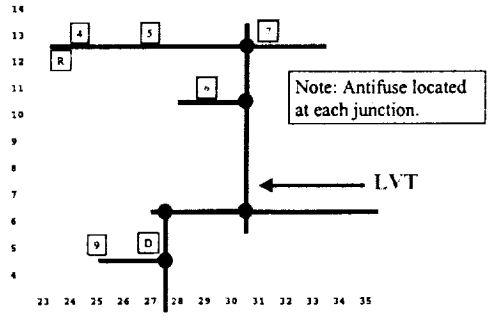
- Clock trees are made to increase fanout.
- Not placing buffers and flip-flops on the same row
  - Can increase skew problem.

Clock Skew - Timing Model



- Hold time at FF2 is the concern.
  - Worst-case
  - Low V_{IH} FF1
  - Hi V_{IH} FF2
  - Fast T_{CQ}, T_{Route}
  - High T_{SKEW}
- $T_{CQ} + T_{ROUTE} + T_H > T_{SKEW}$

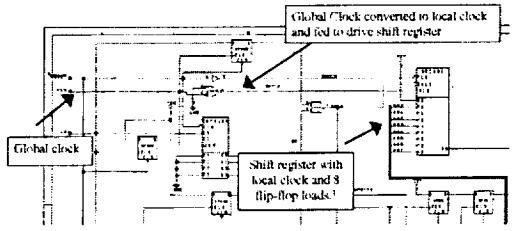
Local Clock: Physical Realization



The net CMDREG/CLK3 driven at location XY = (27, 5) uses an LVT.
LVT data: column = 30, Y-span = (14, 6).
Net data: fanout = 13, Y-spread of inputs = (13, 5).

Design Strategy (2)

Use of Local, High-Skew Clock



[1]This project had a design rule of no more than 5 loads on a local, high-skew clock. This was repeatedly violated.

Clock Skew - Timing Analysis

Most static timing analyzers give bounded numbers for min, max.

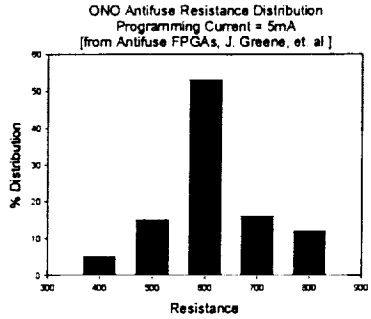
Just setting "MAX" or "MIN" does not account for variations as a result of fabrication differences, anti-fuse resistance, changes as a result of aging, etc, and will be too liberal

A full MIN/MAX analysis is too conservative since elements near each other on the same die can vary that widely. I.e., one part can't be at 4.5VDC, the other at 5.5VDC.

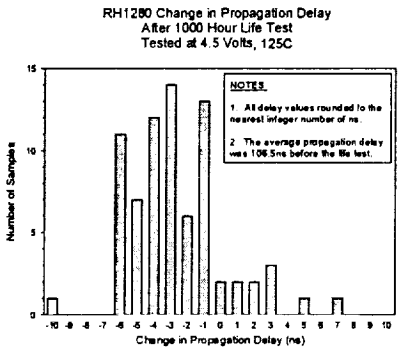
For each environmental condition, it is fair to hold temperature, voltage, fixed.

MIN/MAX will still be a bit conservative, since will range over all manufacturing conditions, not limited to variation within a single die.

Antifuse Resistance Variation

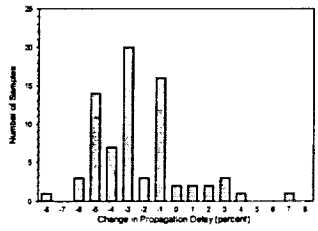


Prop Delay Delta vs. Life



Prop Delay Delta vs. Life

RH1280 Change in Propagation Delay
After 1000 Hour Life Test
Tested at 4.5 volts, 125C

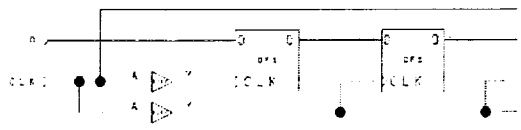


Note: Over a long path, 16 modules + I/O, T_p exceeding 100 ns.

Clock Skew - From VHDL Coding Example

```
Library IEEE;
  Use IEEE.Std_Logic_1164.All;

Entity Skew Is
  Port ( Clk   : In  Std_Logic;
         D     : In  Std_Logic;
         Q     : Out Std_Logic  );
End Skew;

Library IEEE;
  Use IEEE.Std_Logic_1164.All;

Architecture Skew of Skew Is
Signal ShiftReg : Std_Logic_Vector (31 DownTo 0);
Begin
  P: Process ( Clk )
    Begin
      If Rising_Edge (Clk)
        Then Q                      <= ShiftReg(0);
             ShiftReg (30 DownTo 0) <= ShiftReg (31 DownTo 1);
             ShiftReg (31)          <= D;
        End If;
    End Process P;
End Skew;
```

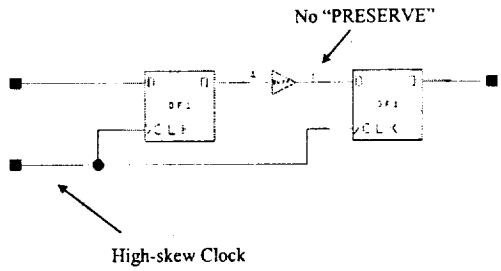
Clock Skew - From VHDL Synthesized Results



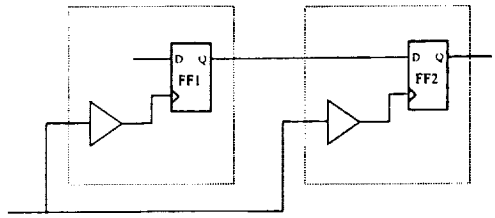
Results will depend on coding, directives and attributes, synthesizer, and synthesizer revision.

Here we see that the logic synthesizer generated a poor circuit.

Clock Skew Correction



Clock Skew - Chip-to-Chip



Analysis may show problems. Some architectures are designed with 0 ns t_H; others incorporate delay elements (configurable) on the data inputs to ensure reliable clocking.

Self-Test: Processors

Processor Hardware Self-Test

Typically, a self-test program for checkout or restarting is a boot-strapping procedure which begins with the verification of the most elementary set of instructions, i.e., those which rely on only a fraction of the computer hardware in order to operate. These instructions are then used to construct a decision-making subroutine which verifies some primitive condition on a YES-NO basis. Once verified, this subroutine (or several similarly constructed) is used to check all other instructions and variations in sequence, beginning with the next least complex instruction and working up to the most complex instruction. After all instructions are verified, input/output (I/0) and memory self-test programs check the remaining hardware.

Processor Hardware Self-Test
Case Study: Gemini

Self-test routines are also important for detecting malfunctions during operation. In the Gemini project, for example, diagnostic subroutines were interleaved in the operational computer program. When they detected a fault, a discrete command was issued to light a malfunction indicator lamp on the control panel. The circuit had a manual reset capability to test whether it was set by a transient malfunction.

Processor Hardware Self-Test
Case Study: Gemini (cont'd)

Three self checks were performed during flight:

- A timing check, based on the noncoincidence of certain signals within the computer under proper timing conditions.
- A thorough diagnostic test which exercised all of the computer's arithmetic operations during each computer cycle in all modes.
- A looping-check, to verify that the computer was following a normal program loop. A counter in the output processor was designed to overflow every 2.75 sec. Each program was written to reset this counter every 2.7 sec; thus, any change in the program flow would cause an overflow and indicate a malfunction.

Processor Hardware Self-Test
Case Study: Apollo Guidance Computer

The Apollo guidance computer is equipped with a restart feature comprising alarms to detect malfunction and a standard initiation sequence which leads back into the programs in progress. The AGC has six malfunction detection devices that cause a restart, as follows:

- A parity test of each word read from memory. An odd-parity bit is added to each fixed- memory word at manufacture time and to each erasable word at write time.
- A looping check much like the one on Gemini. A specified register must be periodically tested by any correctly operating program. This register is "wired" and if it is not tested often enough will cause restart.

Processor Hardware Self-Test
Case Study: Apollo Guidance Computer

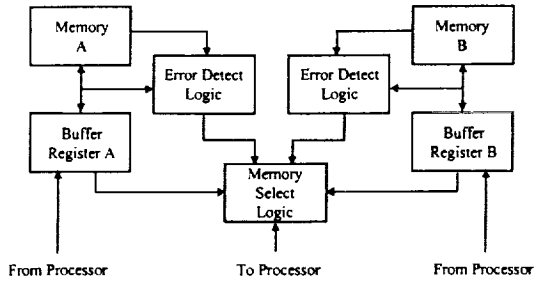
- A transfer control trap, which detects endless loops containing only control transfer instructions, such as a location L which contains the instruction "transfer control to location L."
- An oscillator fail check caused by stopping of the timing oscillator.
- Voltage fail circuits to monitor the 28-, 14-, and 4-V power levels which drive the computer.
- An interrupt check, which detects excessive time spent in the interrupt mode, or too much time spent between interrupts.

Processor Hardware Self-Test
Case Study: Saturn V Launch Vehicle

- Logic used TMR
  - Disagreement detector for faults
  - Switches to simplex if fault detected.[1]
  - Memory was dual-redundant with parity
  - Both memories read in parallel
  - If fault, then backup memory read, correct data written to both memories (DRO core)
  - Switch prime and backup units

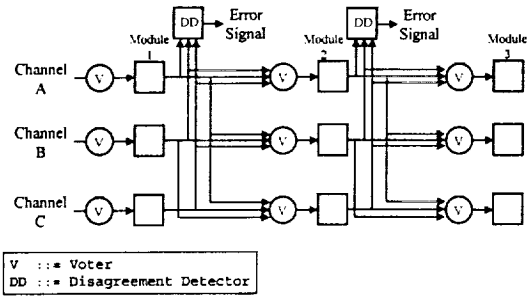
[1]Need to verify from a second source.

Processor Hardware Self-Test
Case Study: Saturn V Launch Vehicle



Saturn V LVDC Duplex Memory Diagram
Self-Correcting Duplex Logic

Processor Hardware Self-Test
Case Study: Saturn V Launch Vehicle

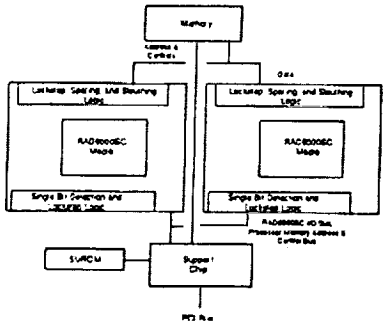


Saturn V LVDC TMR Logic

Processor Hardware Self-Test
Case Study: Space Shuttle

- 4 of the 5 identical computers operate in an NMR configuration
  - Computers synchronized and outputs between computers are compared on the I/O busses
- Voting at the actuator
  - hydraulic voting mechanism: force-fight voter
- After two failures, operates as a duplex system with comparison and self-test techniques

Case Study: Lockstep Operation



[illegible]

Processor Hardware Self-Test
Case Study: MA31750/MIL-STD-1750A

- On-chip parity generation/checking
- Built-In test
  - Part of initialization
  - Manufacturer defined XIO Instruction
    - Code $840D_{16}$
  - For Tracor RHEC and MAS281
    - BIT part of initialization
    - Called using Built-In Function (BIF) 4F

Processor Hardware Self-Test
Case Study: MA31750/MIL-STD-1750A

Built-In Test (BIT) Coverage

- Temporary Registers (T0-T11)
- General Registers (R0-R15)
- Flags Block
- Sequencer Operation and ROM Checksum
- Divide Routine Quotient Shift Network
- Multiplier and ALU
- Barrel Shift Network
- Interrupts and Fault Handling and Detection
- Address Generator Block
- Instruction Pipeline

Processor Hardware Self-Test
Case Study: MAS281/MIL-STD-1750A

Built-In Test (BIT) Coverage

- Microcode sequencer; IB Register Control; Barrel Shifter; Byte Operations and Flags
- Temporary Registers (T0-T7); Microcode Flags; Multiply; Divide
- Interrupt Unit - MK, PI, FT; Enable/Disable Interrupts
- Status Word Control; User Flags; General Registers (R0-R15)
- Timer A; Timer B

Hardware Self-Check
Case Study: IA-64

- L2 and L3 are ECC protected
  - L2 is on-chip, 96 kB unified, 6-way set associated, 64-byte line
  - L3 is on-cartridge, up to 4 MB, 4-way set associated, 64-byte line
- "The processor implements a machine check architecture (MCA) that provides the ability to continue, Recover, or Contain detected errors. All significant structures on the chip are protected by parity of ECC."

"The First IA-64 Microprocessor," S. Rusu and G. Singer, IEEE Journal of Solid-state Circuits, November, 2000

Hardware Self-Test
Case Study: MIL-STD-1553B

- Mode Code 00011 - Initiate Self-test
- Terminal fail-safe. Hardware ensures that no transmission is greater than 800.0 μs (4.4.1.3)
- Listening to the transmitted signal to ensure it matches what was sent.
  - (Look up to see if 1553 requirement or implementation)

Metastable States

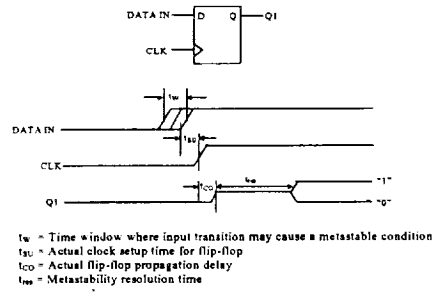
Metastability - Introduction

- Can occur if the setup (t_{SU}), hold time (t_H), or clock pulse width (t_{PW}) of a flip-flop is not met.
- A problem for asynchronous systems or events.
- Can be a problem in synchronous systems.
- Three possible symptoms:
  - Increased CLK -> Q delay.
  - Output a non-logic level
  - Output switching and then returning to its original state.
- Theoretically, the amount of time a device stays in the metastable state may be infinite.
- Many designers are not aware of metastability.

Metastability

- In practical circuits, there is sufficient noise to move the device output of the metastable state and into one of the two legal ones. This time can not be bound. It is statistical.
- Factors that affect a flip-flop's metastable "performance" include the circuit design and the process the device is fabricated on.
- The resolution time is not linear with increased circuit time and the MTBF is an exponential function of the available slack time.

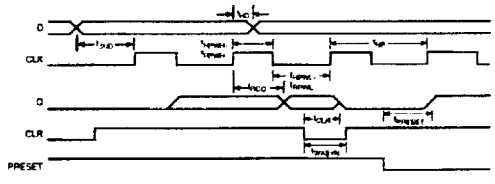
Metastability



t_w = Time window where input transition may cause a metastable condition
t_{SU} = Actual clock setup time for flip-flop
t_{CO} = Actual flip-flop propagation delay
t_{res} = Metastability resolution time

A5

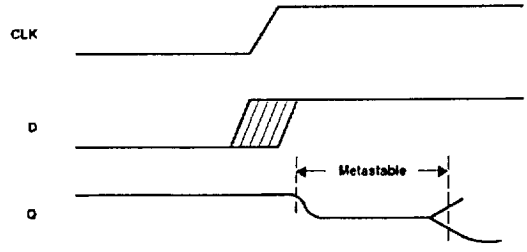
Flip-Flop Timing: RT54SX-S



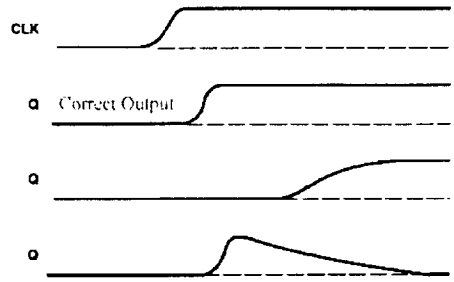
Worst-case Military Conditions, V_{CCA}=2.3, V_{CCI}=3.0V, T_J=125°C
-1 Speed Grade

		Min	Max	Units
t_{RCO}	Sequential Clock-to-Q		1.0	ns
t_{CLR}	Asynchronous Clear-to-Q		0.9	ns
t_{PRESET}	Asynchronous Preset-to-Q		1.0	ns
t_{SUD}	Flip-Flop Data Input Set-Up	0.6		ns
t_{HD}	Flip-Flop Data Input Hold	0.0		ns
t_{WASYN}	Asynchronous Pulse Width	1.5		ns

Metastable State: Possible Output from a Flip-flop



Metastable State: Possible Outputs from a Flip-flop



Metastability - Calculation

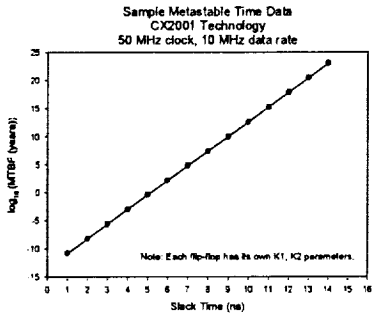
- $MTBF = e^{K2*t} / (K1 \times F_{CLK} \times F_{DATA})$

  t is the slack time available for settling

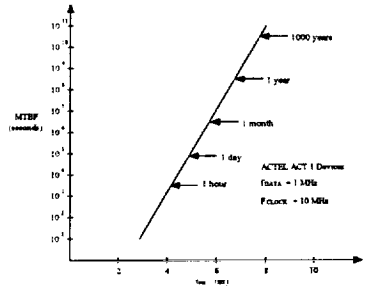
  K1 and K2 are constants that are characteristic of the flip-flop

  F_{CLK} and F_{DATA} are the frequency of the synchronizing clock and asynchronous data.
- Software is available to automate the calculations with built-in tables of parameters.
- Not all manufacturers provide data.

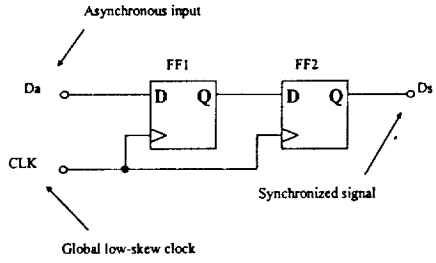
Metastability - Sample Data



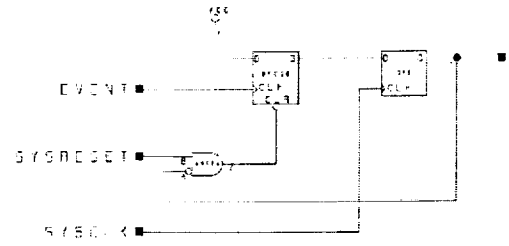
MTBF versus Metastability Resolution Time



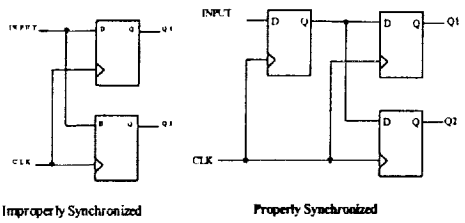
Synchronizer



Synchronizer - Bad



Synchronizing an Asynchronous Input



Improperly Synchronized

Properly Synchronized

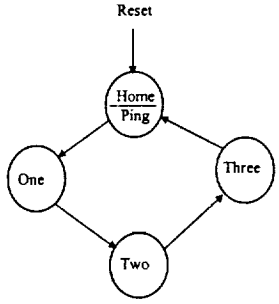
A5

Finite State Machines

Finite State Machines

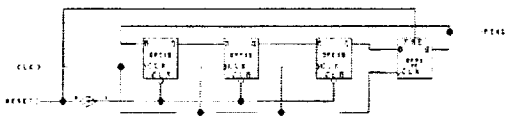
- One-Hot Finite State Machines
  - Normal operation has exactly one flip-flop set, all other flip-flops reset.
  - Next state logic equations for each flip-flop depend solely on a single state (flip-flop) and external inputs.
- Binary encoded state machines
  - Next state logic equations are dependent on all of the flip-flops in the implementation.
- Lockup State
  - A state or sequence of states outside the normal flow of the FSM that do not lead back to a legal state.
- CAE Tools - Synthesizers
  - Generates logic to implement a function, guided by the user.
  - Typically does not generate logic for either fault detection or correction.

Lockup States
Sample State Machine

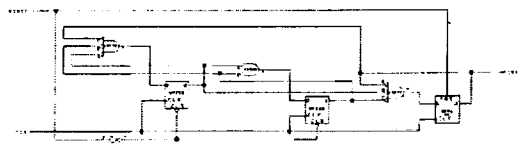



```
Library IEEE;  Use IEEE.Std_Logic_1164.All;
Entity Oneshot_Simple_Act Is
  Port ( Clk   : In  Std_Logic;
         Reset : In  Std_Logic;
         Ping  : Out Std_Logic   );
End Oneshot_Simple_Act;

Library IEEE;  Use IEEE.Std_Logic_1164.All;
Architecture Oneshot_Simple_Act of Oneshot_Simple_Act Is
Type     StateType Is ( Home, One, Two, Three );
Signal   State     :  StateType;

Begin
  M: Process ( Clk, Reset )
     Begin
       If ( Reset = '1' )
          Then State <= Home;
          Else If Rising_Edge (Clk)
                  Then Case State Is
                          When Home  => State <= One;
                          When One   => State <= Two;
                          When Two   => State <= Three;
                          When Three => State <= Home;
                          End Case;
               End If;
       End If;
     End Process M;
  O: Process (State)
     Begin
       If (State = Home)
          Then Ping <= '1';
          Else Ping <= '0';
          End If;
     End Process O;
End Oneshot_Simple_Act;
```

Lockup States
A One-Hot Implementation



Lockup States
Another One-Hot Implementation

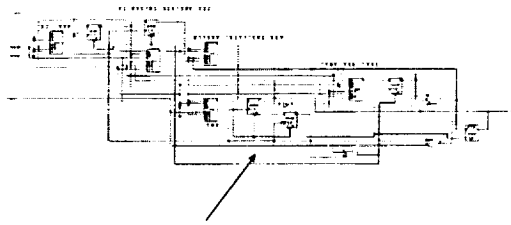


Note: Results depend on version of synthesis software.

Lockup States
Yet Another One-Hot Implementation

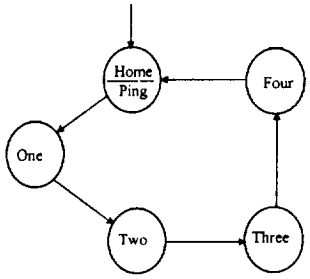
Modified one-hot state machine (reset logic omitted) for a 4-state, two-phase, non-overlapping clock generator. A NOR of all flip-flop outputs and the home state being encoded as the zero vector adds robustness. Standard one-hot state machines [Q3 would be tied to the input of the first flip] have 1 flip-flop per state, with exactly one flip-flop set per state, presenting a non-recoverable SEU hazard.

Lockup States
A "Safe" One-Hot Implementation



Reset flip-flops. Note second one is on falling edge of the clock. This implementation uses 6 flip-flops.

Lockup States - Binary Encoding



Three unused states.

Lockup States
Binary Encoding

```
Type    StateType Is ( Home, One, Two, Three , Four);
Signal  State     :  Statetype;

          -

Case State Is

          -

       When Others => State <= Home;
```

"When Others" refers to states in the enumeration, not the physical implementation. Also, states that are not reachable can be deleted, depending on the software and settings.

Two Most Common Finite State Machine (FSM) Types

- Binary: Smallest m (flip-flop count) with $2^m \geq n$ (state count), highest encoding efficiency.
  - Or Gray Coded, a re-mapping of a binary FSM
- One Hot: m = n, i.e., one flip-flop per state, lowest encoding efficiency.
  - Or Modified One Hot: m = n-1 (one state represented by 0 vector).

Issue: How To Protect FSMs Against Transient Errors (SEUs and MEUs):

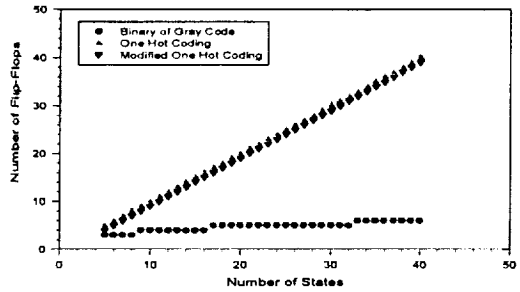
- Illegal State Detection
- Adding Error Detection and Correction (EDAC) Circuitry

Many of the following slides are from:

Sequential Circuit Design for Spaceborne and Critical Electronics

Mil/Aero Applications of Programmable Logic Devices (MAPLD) International Conference, 2000.

Encoding Efficiency: Binary vs. One Hot

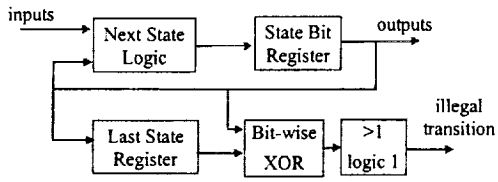


Binary and Gray Codes
FSM State Sequences

3-bit Reflected Gray Code	Binary Code
0 0 0	0 0 0
0 0 1	0 0 1
0 1 1	0 1 0
0 1 0	0 1 1
1 1 0	1 0 0
1 1 1	1 0 1
1 0 1	1 1 0
1 0 0	1 1 1

- Binary sequence can have 0 (hold), 1, 2, ..., n bits changing from state to state.
- Gray code structure ensures that either 0 (hold) or 1 bit changes from state to state.
- Illegal states in either type are detected in the same way, i.e., by explicit decoding.

Gray Code
Illegal Transition Detection



False illegal transition indications can also be triggered by errors in the Last State Register, and doubling the number of bits doubles the probability of an SEU.

One Hot FSM Coding

One Hot Coding	Binary Code
1 0 0 0 0 0 0 0	0 0 0
0 1 0 0 0 0 0 0	0 0 1
0 0 1 0 0 0 0 0	0 1 0
0 0 0 1 0 0 0 0	0 1 1
0 0 0 0 1 0 0 0	1 0 0
0 0 0 0 0 1 0 0	1 0 1
0 0 0 0 0 0 1 0	1 1 0
0 0 0 0 0 0 0 1	1 1 1

- Many (2^n-n) unused states - not "reachable" from VHDL[2].
- Illegal state detection circuitry complex
- Parity (odd) will detect all SEUs, not MEUs

[illegible]

One Hot FSM Coding
Lockup States

SEU →

7	6	5	4	3	2	1	0	
1	0	0	0	0	0	0	0	
0	1	0	0	0	0	0	0	
0	0	1	0	0	0	0	0	
0	0	0	1	0	0	1	0	FSM is locked up.
0	0	0	0	1	0	0	1	
1	0	0	0	0	1	0	0	
0	1	0	0	0	0	1	0	
0	0	1	0	0	0	0	1	

One Hot FSM without protection.

Modified One Hot FSM Coding

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

One Hot Coding

6	5	4	3	2	1	0
0	0	0	0	0	0	0
1	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	1

Modified One Hot Coding

Note: Often used by synthesis when one hot FSM specified. Modified one hot codings use one less flip-flop.

Modified One Hot FSM
Illegal State Detection

- Error detection more difficult than for one hot
  - 1 → 0 upsets result in a legal state.
  - Parity will not detect all SEUs.
  - If an SEU occurs, *most likely* the upset will be detectable
- Recovery from lockup sequence simple
  - If all 0's (NOR of state bits), then generate a 1 to first stage.
  - If multiple 1's (more difficult to detect), then will wait until all 1's are "shifted out."

Is There a Best FSM Type, and Is It Best Protected Against Transient Errors By Circuit-Level or System-Level EDAC?

- Circuit-level EDAC
  - Expensive in power and mass if used to protect all circuits
  - Can be defeated by multiple-bit transient errors
- System-level EDAC
  - Required for hard-failure handling
  - Relies on inherent redundancy in system, high-level error checking, and some EDAC hardware

System-Level Error Checking Mechanisms

- Natural error checking mechanisms
  - e.g., fire a thruster, check for spacecraft attitude change
- Checking mechanisms arising from multiple subsystems
  - e.g., command a module to power on, check its current draw and temperature
- Explicitly added checking mechanisms
  - Watchdog timers
  - Handshake protocols for command acknowledgement
  - Monitors, e.g., thruster on-time monitor

Transient Errors Cause FSM Jumps to Erroneous States

Jump to	Pathology	Circuit Level Response
Illegal state	•Impartially decoded states allow erroneous state machine outputs •Appropriate recovery state difficult to determine	•Homing sequence, reset controlled circuitry –Success depends on nature of system •Stop, raise error flag, handle at system level
Legal state	Incorrect sequencing of state machine activities	Probably detectable at system level only based on incorrect module operation

System-Level Error Handling Mechanisms Also Handle Transient Error Effects

Transient Error Effect	System Response
Command Rejection	Command Retry
Telemetry or Data Corruption	Data Filtering, also required to handle system noise
FSM Lock-up, e.g., detected by multiple command rejections	Indistinguishable from hard error

EDAC Required For Some FSMs Based on Criticalness of Circuit and Probability of Error

Common EDAC Types

Type	Capability	Power & Mass Impact
Parity	Detect 1 bit error, correct 0	Extra bit, parity trees to set and check
NMR	Correct int(N/2) bit errors (strong correction)	Multiplies gate count by N+ and clock loading by N
Hamming	Correct 1 bit error, Detect 2 (or more, depending on code) (weak correction)	Close to TMR in gate count, much lower clock loading

Impact of Adding EDAC to Common FSM Types

FSM Type	Protecting with EDAC
Binary	High encoding efficiency => smallest EDAC impact Potentially few illegal states => fairly easy to detect Full decoding eliminates effects of illegal states
One-hot	Poor encoding efficiency => greatest EDAC impact Many illegal states => complex circuit to detect Full decoding defeats advantage of easy state decoding

FSM Conclusion

- Binary state machine may be optimal for highly reliable systems
  - Most amenable to the addition of EDAC circuitry if necessary because of high encoding efficiency
  - Full state decoding protects against erroneous outputs
  - Easier to detect illegal states
- Overall EDAC scheme must also consider system-level action
  - Will be there for hard failures, anyhow
  - Must consider system response to defeated circuit-level EDAC

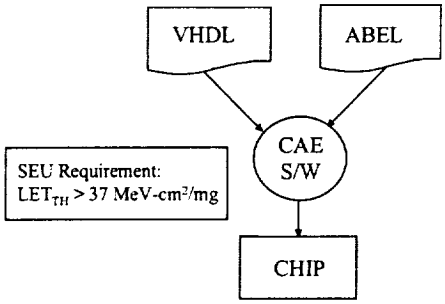
VHDL and Software Issues

VHDL "Interface"

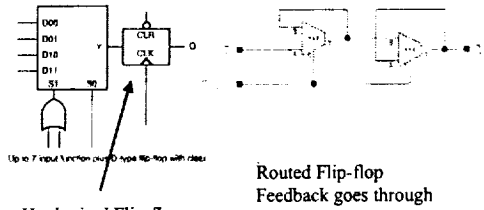
```
Library IEEE;
  Use IEEE.Std_Logic_1164.All;
Entity Bool Is
  Port ( X   : In  Std_Logic;
         Y   : In  Std_Logic;
         Z   : Out Boolean    );
End Bool;

Library IEEE;
  Use IEEE.Std_Logic_1164.All;
Architecture Bool_Test of Bool Is
Begin
  P: Process ( X, Y )
     Begin
       If ( X = Y )
         Then Z <= True;
         Else Z <= False;
         End If;
     End Process P;
End Bool_Test;
```

Boolean signal was mapped to different logical values in different versions of the same VHDL logic synthesizer

An HDL Flow



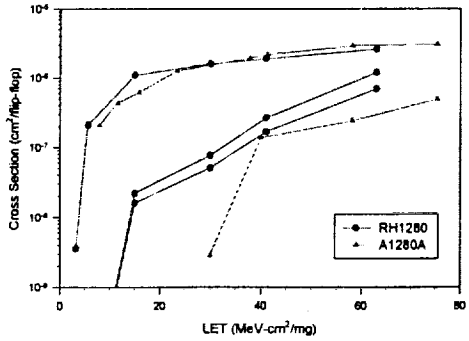
Act 2 Flip-flop Implementation



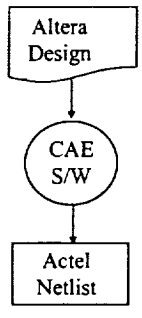
Hard-wired Flip-flop

Routed Flip-flop
Feedback goes through antifuses (R) and routing segments (C)

Act 2 SEU Flip-Flop Data

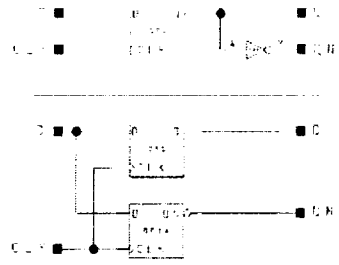


Logic Translation/Optimization Flow



Logic Translation/Optimization

Implementation

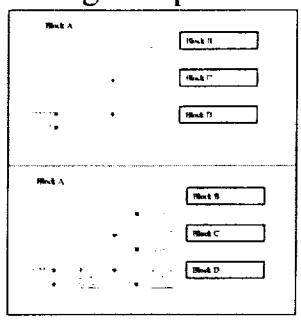


Original

"Optimized"

The two circuits are logically equivalent when analyzed with Boolean logic equations with the lower, CAE-optimized circuit, permitting higher device speeds. An SEU analysis shows the addition of a second state variable with an upset resulting in the "optimized" circuit containing a state where Q = QN, violating the system equations and causing a failure.

Logic Replication



Two methods of signal distribution. The top version shows a signal distributed to multiple blocks with buffs driving multiple loads. The bottom version replicates flip-flops, resulting in higher system speeds. Routing delays are significant. Recovery from SEU's with multiple flip-flops are not considered by current computer-aided engineering tools.

Delay Generation

VHDL Code and Synthesizer Analysis

Case Study - Hardened Clock Generator

- The VHDL synthesizer, unknown to the designer, generated a poor circuit for a TMR voter
  - Used 3 C-Cells for a voter
  - Slowed the circuit down
- The implementation of the voter is hidden from the user
  - Synthesizer generated a static hazard
  - An SEU can result in a glitch on the "hardened" clock signal.

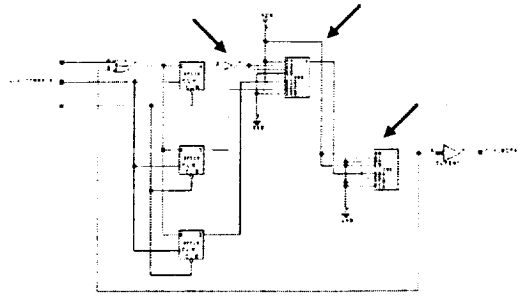
VHDL Code and Synthesizer Analysis

Case Study - Hardened Clock Generator

```
-- Divide 25 MHz (40 ns) clock by 4
-- to produce 6.25 MHz clock (160 ns)
-- This clock should be placed on
-- an internal global buffer

clkint1: clkint
Port Map ( A => clk_div_cnt(1),
           Y => clk_div4       );

clkdiv: Process (reset_n, clk)
Begin
  If reset_n = '0' Then
     clk_div_cnt <= "00";
  Elsif clk = '1' And clk'EVENT Then
        clk_div_cnt <= clk_div_cnt + 1;
  End If;
End Process clkdiv;
```

VHDL Code and Synthesizer Analysis

Case Study - Hardened Clock Generator

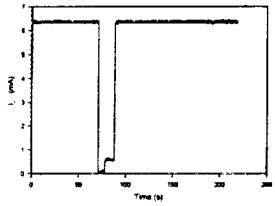


Most significant bit of the counter. 3 C-Cells are used for the voter.

Loss of Functionality

- FRAM
- DRAM - JEDEC
- JTAG
- PROM
- Microprocessor

FRAM Memory Functionality Loss During Heavy Ion Test



Strip chart of FM1608 (research fab) current during heavy ion irradiation. The device lost functionality during the test while the current decreased from it's normal dynamic levels of approximately 6.3 mA to it's quiescent value, near zero. The device recovered functionally and operated normally throughout the latter part of the test. This effect was seen at least three times during the limited testing of this device.

DRAM Modes

DRAM Special Test and Operational Modes

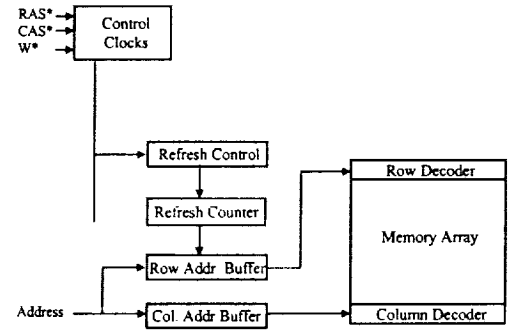
This standard defines a scheme for controlling a series of special modes for address multiplexed DRAM. The standard defines the logic interface required to enter, control, and exit from the special modes. In addition, it defines a basic special test mode plus a series of other special test and operational modes.

TEST MODES are those that implement some special test of measurement function or algorithm designed to enhance the ability of the Vendor or User to determine the integrity of, or to characterize, the part.

OPERATIONAL MODES are those that alter the operational characteristics of the part but do not interfere with its function as a storage device and are intended to be used in system operation.

JEDEC Standard No. 21-C, page 3.9.5-7, Release 4

DRAM Refresh



Adapted from: http://www.tecchannel.de/hardware/173/6.html

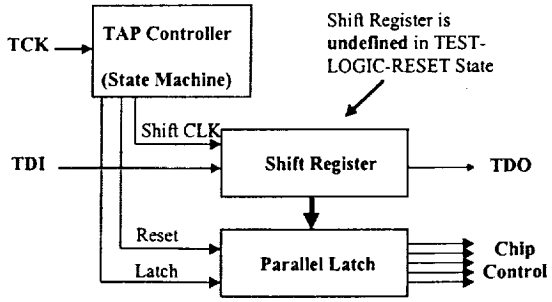
DRAM Refresh

CAS#-BEFORE-RAS# REFRESH is a frequently used method of refresh because it is easy to use and offers the advantage of a power savings. Here's how CBR REFRESH works. **The die contains an internal counter** which is initialized to a random count when the device is powered up. Each time a CBR REFRESH is performed, the device refreshes a row based on the counter, and then the counter is incremented. When CBR REFRESH is performed again, the next row is refreshed and the counter is incremented. The counter will automatically wrap and continue when it reaches the end of its count. There is no way to reset the counter. The user does not have to supply or keep track of row addresses.

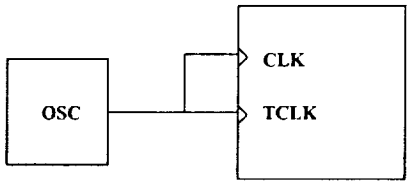
Since CBR REFRESH uses the internal counter and not an external address, the address buffers are powered down. **For power-sensitive applications**, this can be a benefit because there is no additional current used in switching address lines on a bus, nor will the DRAMs pull extra power if the address voltage is at an intermediate state.

Adapted from "Micron Technical Note TN-04-30: Various Methods of DRAM Refresh"

IEEE JTAG 1149.1

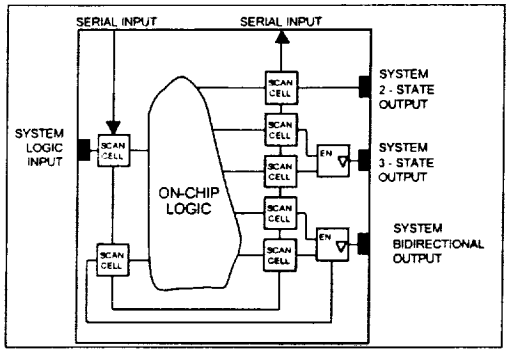


IEEE JTAG 1149.1 TCLK

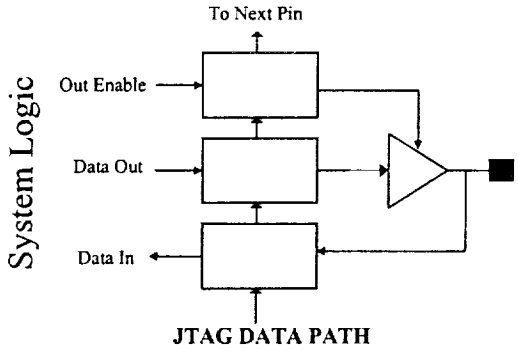


The CLK pin may turn into an output driving low, clamping the oscillator's output at a logic '0'. The TAP controller can not reset and restore I/O operation. Most FPGAs do not have the optional TRST* pin. Note TRST*, when present, has a pull-up.

IEEE JTAG 1149.1 - Scan Path

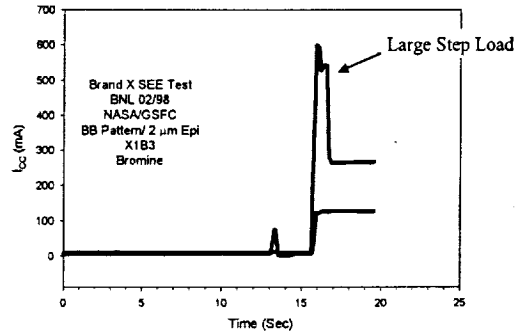


IEEE JTAG 1149.1 - Scan I/O Cell



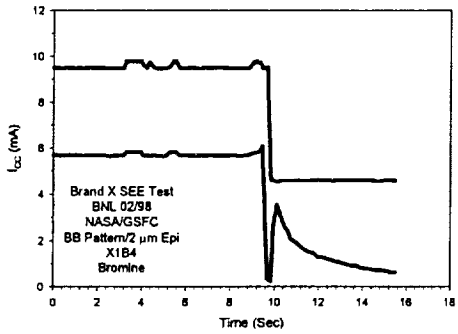
JTAG Upset Effect - Step Load

TCK and TMS=1 Not Guaranteed Solution

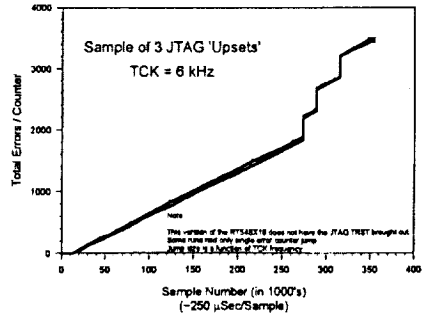


JTAG Upset Effect - Step Load

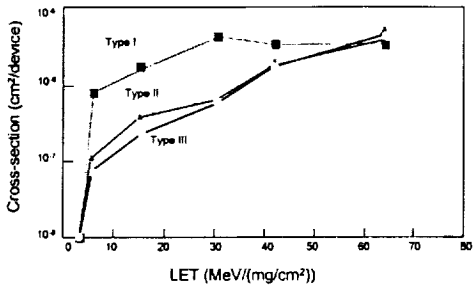
Second Distinct Failure Mode



JTAG Upset Effect - TCK On



SEE Results - Loss of Functionality
Atmel AT28C010 EEPROM, D/C 9706



"Single Event Functional Interrupt (SEFI) Sensitivity in EEPROMs," R. Koga, 1994 IEEE/NSREC Conference, Greenbelt, MD

Atmel AT28C010 EEPROM, D/C 9706
Type I Errors

- Manifested by the appearance of repeated errors, once the first error had been detected during ion irradiation. Here, the first error appeared at some point in time, which was tens of reading cycles ("cycle" is defined in Section II) after the exposure had started. Thereafter we observed one error every few cycles.
- Errors were altered bits in one word at various address locations.
- Simultaneously with the observation of the first error, the device bias current increased to 26 mA from 20 mA (normal, pre-error condition). The bias current continued to be 26 mA until the reading process stopped. At that time, the current became 0.2 mA (quiescent level).
- When the device was read again (without power-cycling), the bias current returned to 26 mA and errors appeared again (even without the beam).
- If the power to the device was shut off and re-started again (power-cycled), the device again functioned properly (i.e., no errors).
- In one instance we continued the irradiation without power-cycling for a long time, until the device no longer showed any errors. It appeared that the affected bit underwent additional upset, returning to the original polarity and thereby correcting the problem.

Atmel AT28C010 EEPROM, D/C 9706
Type II Errors

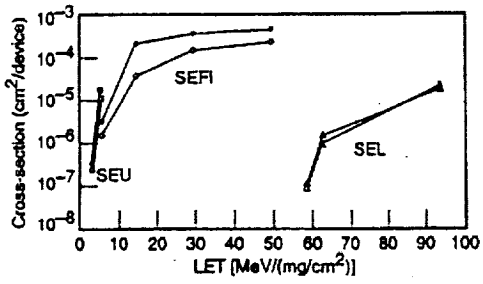
- Manifested by "00" in all address locations, once the first "00" was read.
- These errors could be removed only by power-cycling the device.

Atmel AT28C010 EEPROM, D/C 9706
Type III Errors

- Characterized by occasional errors in a byte, which appeared once in many cycles. There was no 'after-effect' for this type of error. In other words, one error appeared independently once in a while.
- Caused by an upset in the output buffer.

X28HC256 CMOS EEPROM
Xicore, D/C 9140

- Upset mode which also required the cycling of power to clear.



Loss of Functionality
Serial PROM

- Xilinx XQR1701L
  - 10% saturated intercept at LET=6, 1.2×10^{-5} cm²/device

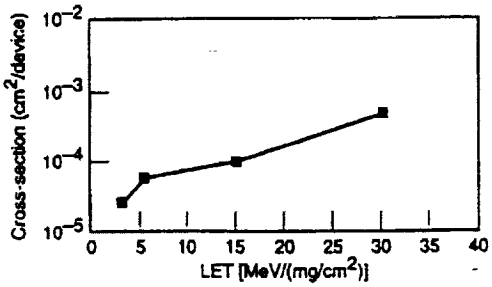
Reference: DS062 (v3.0) February 8, 2001.

Loss of Functionality
Processors

- Processor simply stopped functioning without showing any observable bit errors.
- Noticed lockup in many microprocessors including MG80C186, MG80C286, and XC68302.
- Sensitivity to lockup was essentially independent of the test programs.

"Single Event Functional Interrupt (SEFI) Sensitivity in EEPROMs," R. Koga, 1994 IEEE/NASA Radiation Conference, Greenbelt, MD.

Loss of Functionality
Processors: XC68302 Example



Specifications

Specifications
General Principles

- No Specification Produced
- Specification not Followed

Common Error - Seen More Often Than One Would Expect

Specifications
Case Study 1

- Gate Array Operation Differed from Specification
  - No Continuity of Personnel on Project
  - Features Added and Deleted During Development
  - Changes Were Not Documented in Specification

Specifications
Case Study 2

- Continual Updates to FPGAs Caused Delays to Project
  - Drifting Software Requirements Impacted FPGA
  - Drifting System Requirements Impacted FPGA
- No Stable Specification

Simulators and Limitations

Reliance on Logic Simulators General Principles

- Run Time Limited
- Number of Vectors
- Vector Generation
- Number of Operating Modes
- Time for Modeling External Circuitry
- CAE S/W Limitations

Reliance on Logic Simulators Case Study 1

- Simulator Could Only Simulate 1 ms.
  - Instrument Had a 125 ms Cycle Time.
- Simulating All Inputs Not Practical
  - Too Many Combinations

→Failed to Find a Logic Error Which Caused an Arithmetic Error

Reliance on Logic Simulators Case Study 2

- FPGA Converted to ASIC
- No Gate Level Design Review Performed at Any Stage
- Test Vectors from FPGA Version Were Not Run on the ASIC Version
- Test Vectors Were Capable of Detecting the Design Error

Analysis vs. Simulation

From the Project documentation:

> All ... Actel designs were re-simulated using back-annotated timing data, to ensure that clock skews were within proper limits.

From Actel documentation:

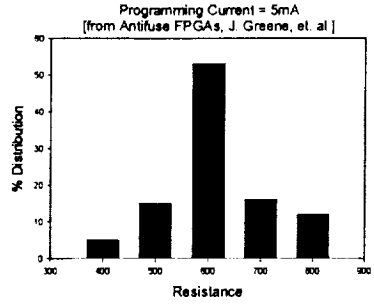
> To verify that a design works properly, both the design's functionality and its timing must be checked. Static timing analysis checks timing, but not the design's functionality. **Simulation checks the functionality of a design, but it may miss some timing problems**. Used together, static timing analysis and simulation complement each other to provide complete design verification

Analysis vs. Simulation (cont'd)

From Actel documentation:

> Both gate array and FPGA designs are susceptible to race conditions, which require careful analysis of setup and hold times, and clock skew across best-case and worst-case operating conditions. This application note describes how to use the Actel Timer to analyze accurately these types of potential timing problems. The Timer is a powerful static timing analysis tool that can be used successfully to check setup and hold times and clock skew.
>
> Since gate array devices are not production tested for setup and hold times, these parameters must be sufficiently guardbanded to guarantee they will never cause a failure. This is difficult when using backannotated timing simulation since simulation software does not allow best-case and worst-case timing analyses at the same time. Often such analysis is done be hand, if at all. In some cases, designers simply switch their data with the inactive edge of the clock to avoid such timing problems.

ONO Antifuse Resistance Distribution

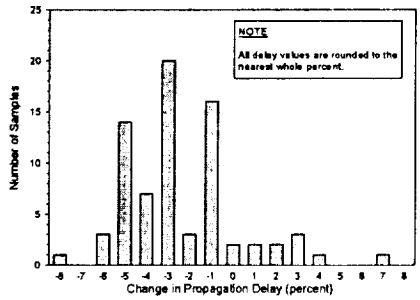


Qualification By Test

- Qualification by test is sometime acceptable
  - Ex., measured t_{PD} vs. data book worst-case values
- Qualification by test is limited
  - Can not simulate all effects of radiation, life
  - Not all changes in t_{PD}, for example, will track
- Qualification by test sometimes fails
  - Intel is recalling its 1.3 gigahertz Pentium III chip, which it has sold to only "a handful" of "power users" running advanced applications, because a certain combination of data, voltage, and temperature conditions may cause the chip to fail. The chip is expected to be back on the market in a couple of months.[1]

[1] Reuters, The Washington Post, "Intel Says ..." reported in computers

Change in t_{PD} Over Life



Data from RH1280-1_SEL report

Verification

Verification Issues (1)

- Macro generators fail
  - Expect them to be correct by construction
  - Working macro fails in later revisions
    - ex., modulo counter
- VHDL Synthesis
  - Simulated vs. Synthesized Results
    - Latch vs. Flip-flops.
  - Lockup states in FSMs
  - Introduction of static hazards
- No simulations or timing analysis.

Verification Issues (2)

- Detailed peer-review of the design is not performed
  - Designs "approved" at the CDR
  - FPGA designs not completed at the CDR
  - Management barriers to review
  - Simulation does not replace analysis
  - Testing does not replace analysis
- Complete worst-case analysis not performed
- Asynchronous design risks not identified, assessed and mitigated

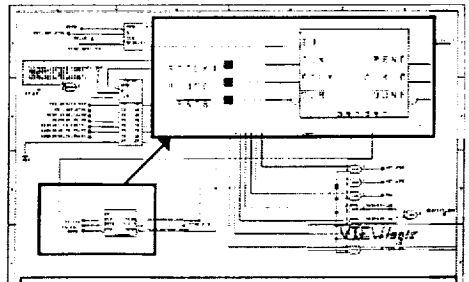
Verification Issues (3)

- Inadequate Reviews
  - Slide flipping
  - Unskilled reviewers
  - Insufficient time
  - Findings not enforced
- Unresolved problems
  - Glitches not fully understood

Review Samples

- Red Team Review
  - No Issues
  - Good FPGA design practices applied
- NASA Civil Servant Design Engineer
  - **"Oh my God!"**
- NASA On-Site Contractor Design Engineers
  - **"This circuit <expletive deleted>!"**
  - **"Oh, <expletive deleted>. <pause> Oh, <expletive deleted>!"**

Design Rule Compliance



Violation of project clock loading rule of no more than 5 flip-flops on a local clock. PB_OSC has 23 loads.

Two Opinions

For a successful technology, reality must take precedence over public relations, for nature cannot be fooled.

-- R. P. Feynman, Report of the PRESIDENTIAL COMMISSION on the Space Shuttle Challenger Accident, Volume 2: Appendix F -- Personal Observations on Reliability of Shuttle, June 6th, 1986

They are our gremlin hunters who are empowered to stalk the shop floor, look over our shoulders and take us to task when they sense something might be wrong. This is not the traditional 2 days of viewgraph watching.

-- Dan Goldin, April 27, 2000 on independent review teams

Conclusion (1)

One must understand not only the "how" but the "why."

Otherwise, failure is not a matter of 'if' but of 'when.'

Conclusion (2)

The key to developing engineering confidence is the rigorous identification of the cause for ALL failures encountered for ALL phases of testing ...

Dr. Joseph F. Shea, Deputy
Director of Manned Space Flight,
Spaceborne Computer Engineering Conference
October, 1962.