Autonomous Car

A New Driver for Resilient Computing and Design-for-Test

Nirmal Saxena
NVIDIA

NEPP Program
ETW 2016
June 15

Disclaimers

My Perspective and Not My Employer's

Attributions– Best Effort from Memory

Keynote Flow

Auto Safety Standard

- Driverless Car Model
- Resiliency & DFT Requirements

Machine Learning

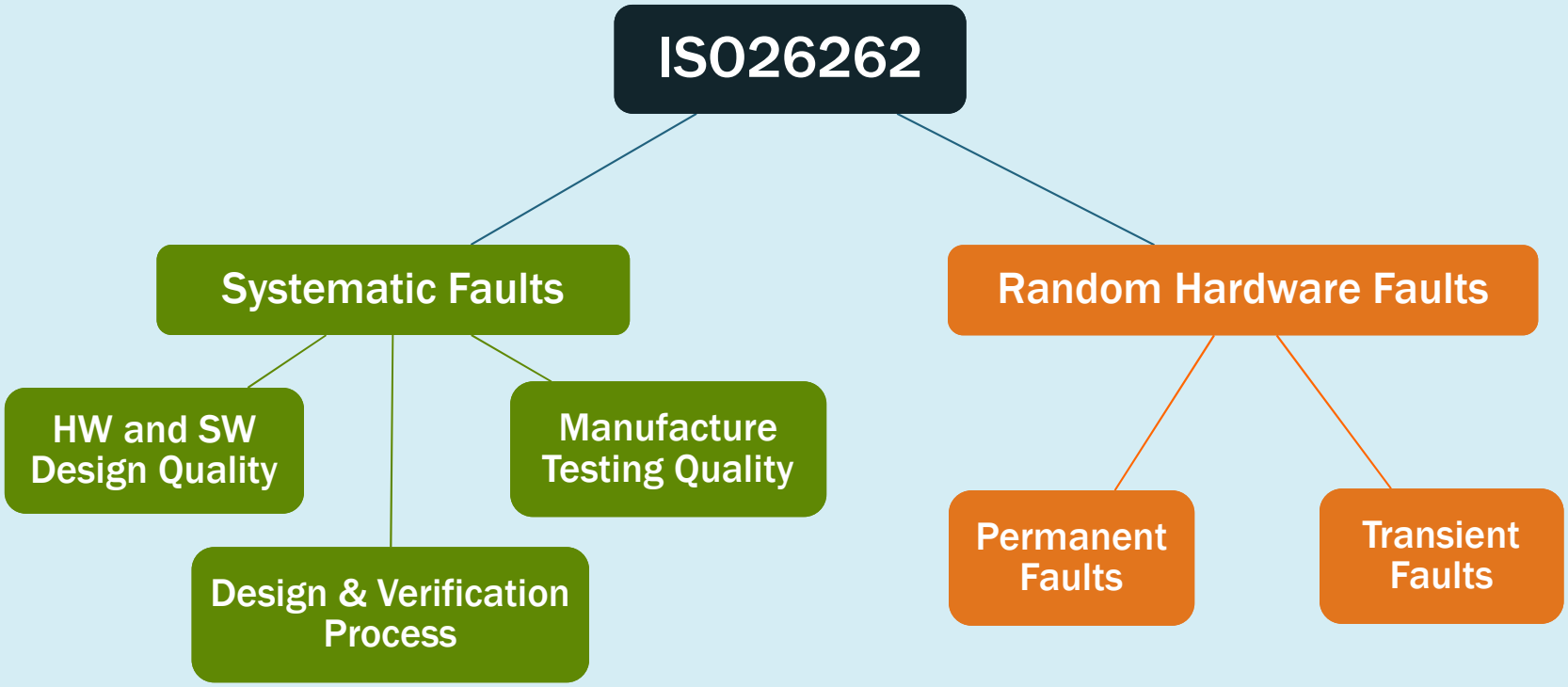
- A Short Tutorial
- Accuracy, Performance and Resiliency

Road to Resiliency

- Reliability Models
- Ordering Statistics
- Important Focus Areas

ISO26262 Auto Safety Standard Review

Auto Safety Requirement Diagram



References:
Daniel P. Siewiorek , The Theory and Practice of Reliable System Design, Digital Press, 1982.
Avizienis, A. “Architecture of Fault-Tolerant Computing Systems”, Digest 5th Intl. Symposium on FTCS, Paris, 1975.

Random Hardware Faults Requirements

Hardware Random Fault Metrics	ASIL B	ASIL C	ASIL D
Permanent Fault Coverage (SPFM)	90%	97%	99%
Transient Fault Coverage (SPFM)	90%	97%	99%
Latent Fault Coverage (LFM)	60%	80%	90%
Hardware Failure Probability (PMHF)	100FIT $\leq 10^{-7}/hr$	100FIT $\leq 10^{-7}/hr$	10FIT $\leq 10^{-8}/hr$

FIT = Failures in Time, Time = 10^9 Hours. 1 FIT = 10^{-9} failures/hr

ASIL	Automotive Safety Integrity Level
SPFM	Single Point Fault Metric
LFM	Latent Fault Metric
PMHF	Probabilistic Metric for Hardware Failures

2013 Accident Statistics– US

Reference: National Highway Traffic Safety Administration (NHTSA): www.nhtsa.gov

Description	Statistics
Fatal Crashes	30,057
Driver Related Fatal Crashes	10,076
Non-Fatal Crashes	5,657,000
Number of Registered Vehicles	269,294,000
Licensed Drivers	212,160,000
Vehicle Miles Travelled	2,988,000,000,000
Driver Related Fatal Crashes per 100Million VMT	0.34
Fatal Crash FIT Rate (Assuming Average Speed 25-50 MPH)	84 to 168
Non-Fatal Crash FIT Rate	28582

ASIL-D 10FIT Target

9x to 17x Reduction in Fatal Crash FIT Rate

3000x Reduction in Non-Fatal Crash FIT Rate

Control System Model– Autonomous Car

30
Frames/Sec

Maps

RADAR
LIDAR

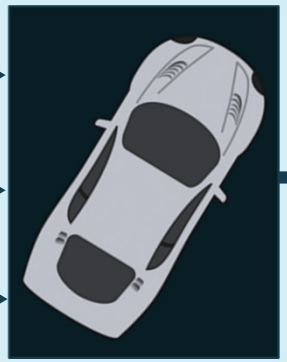
Cameras

Path Planning
Object Detection

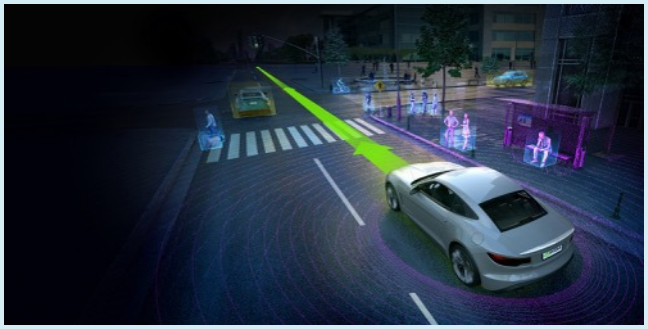
Steer

Accel

Brake



Car Velocity & Position



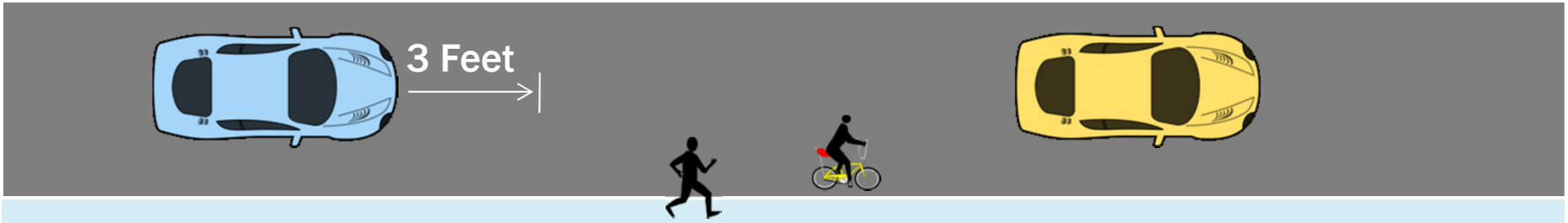
Object Detection & Path Planning

Fault Tolerant Time Interval (FTTI)

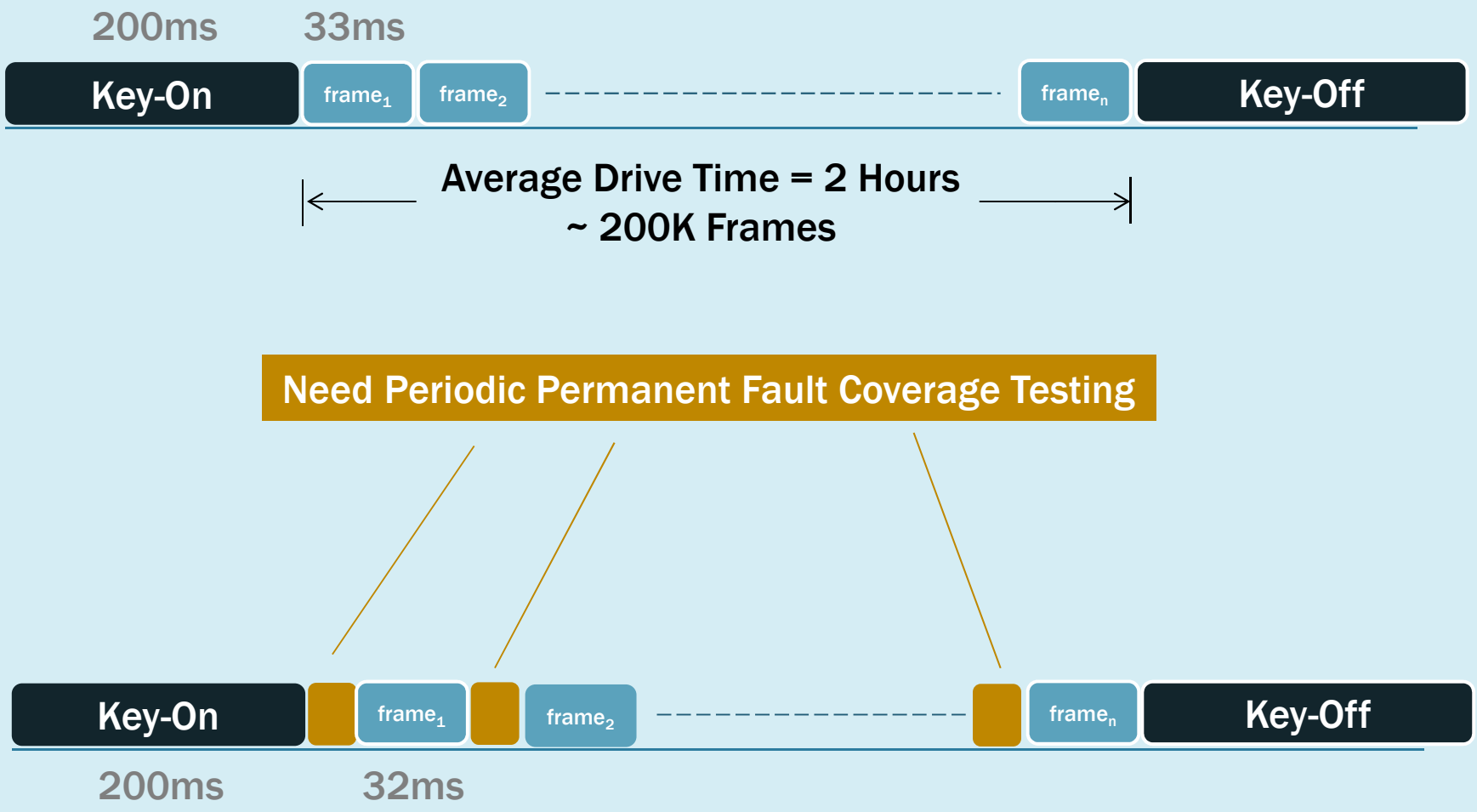
ISO26262 does not Quantify FTTI

Highway Driving 75 MPH

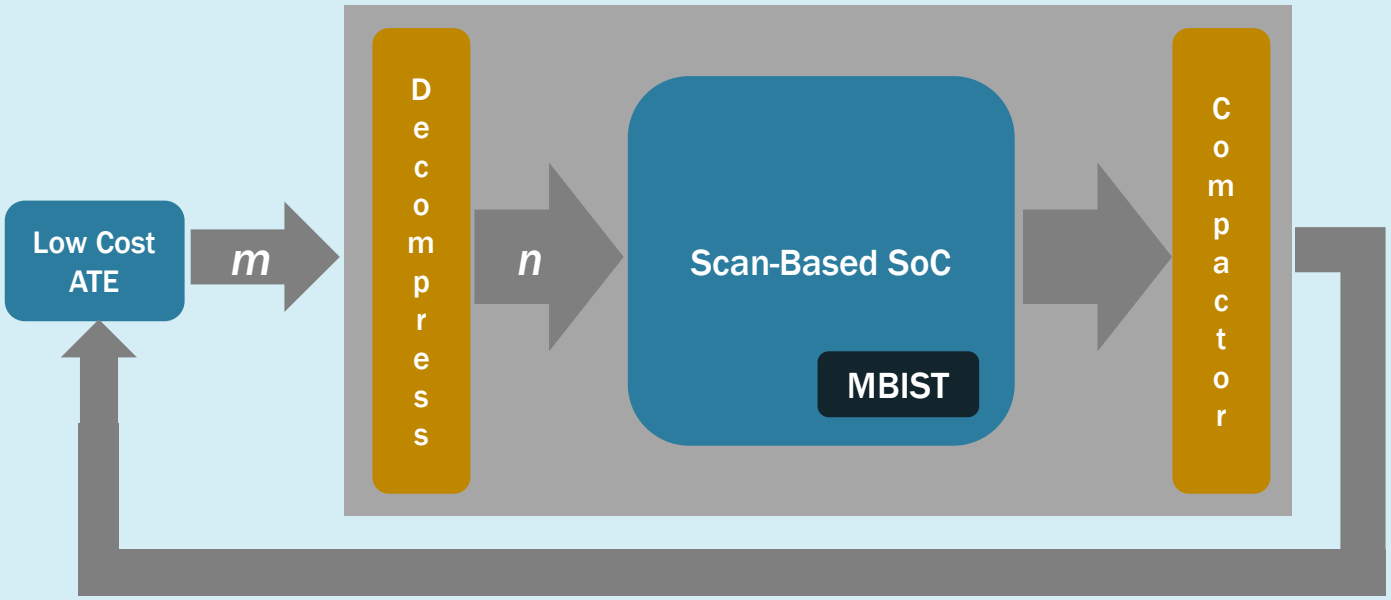
Urban Driving 25 MPH



Key-On, Drive-Time, Key-Off

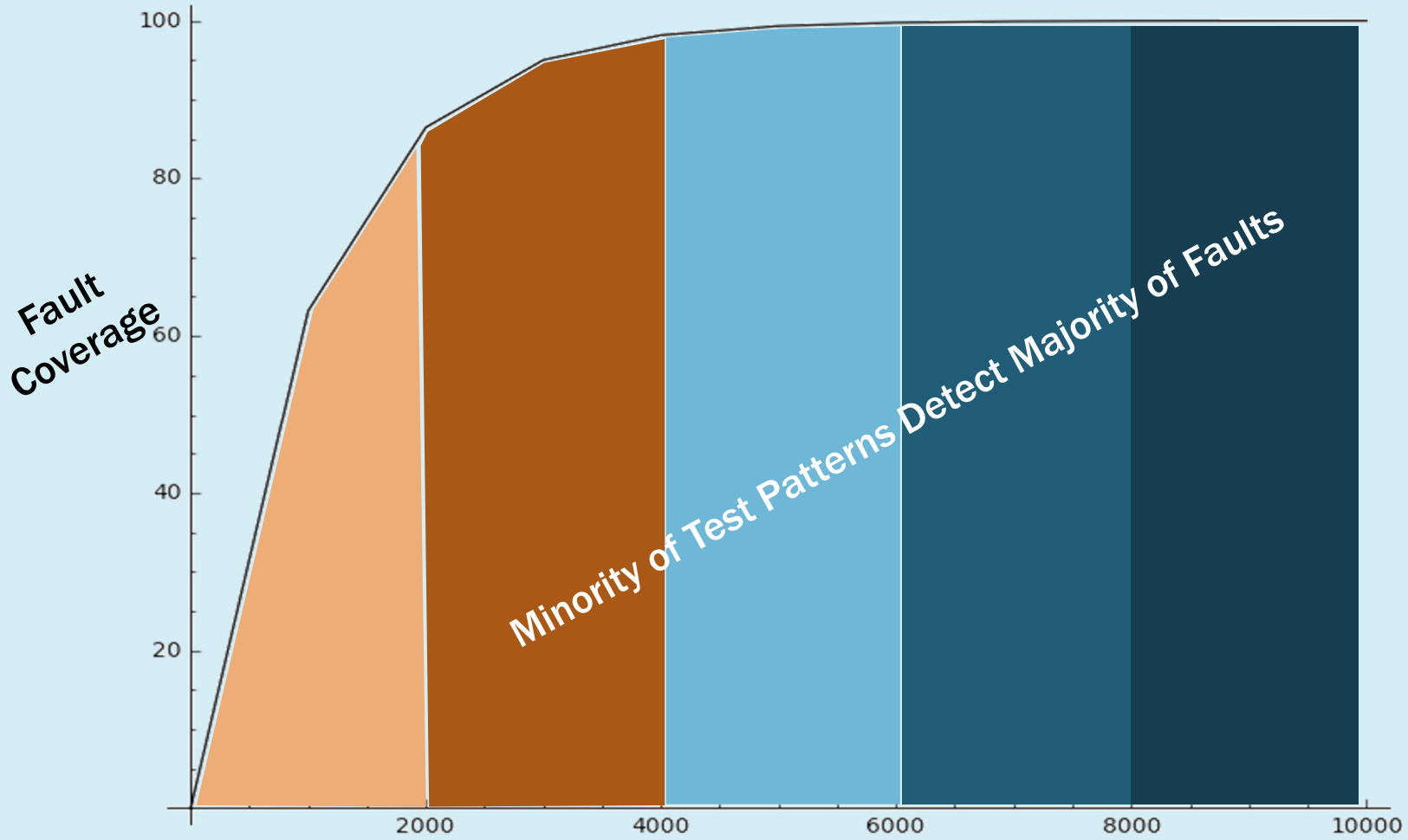


Leveraging Test Compression

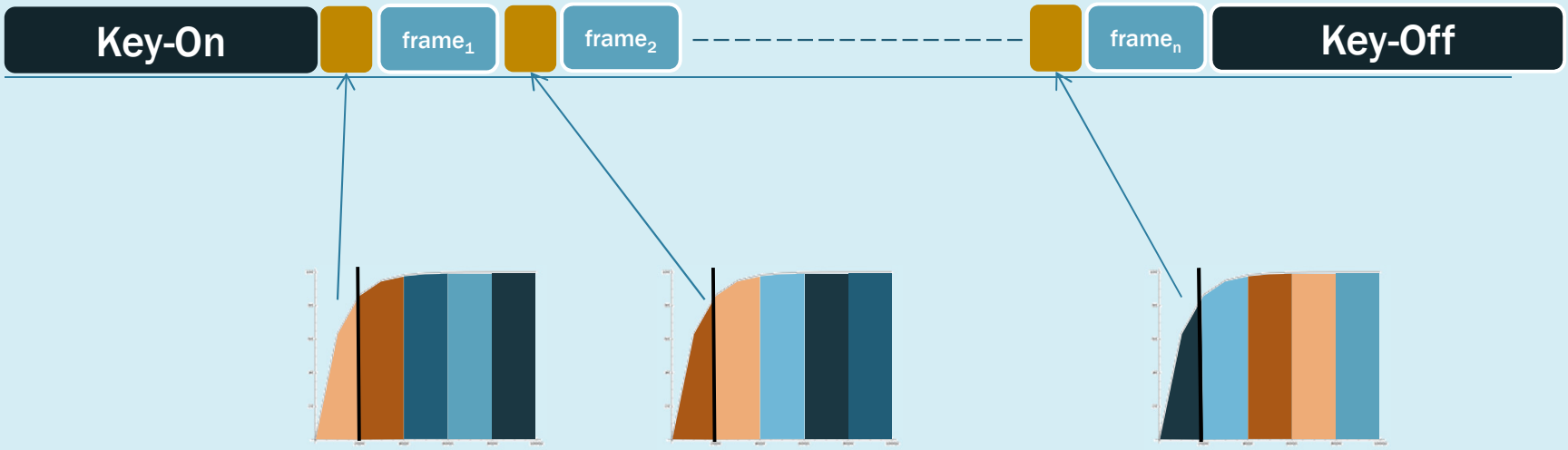


VLSI Test Principles and Architectures, 2006, Edited by: L-T Wang et. al.
Chapter 6 [X. Li, K-J Lee, Nur Touba]
[Reddy et. al. 2002] [Wûrtenberger 2004][[Jas 2003][Reda 2002][Han 2005b]
[Chandra 2001][Krishna 2003][Rajski 2004][Hamzaoglu and Patel 1999][Li 2004]
[Wang 2004][Wohl 2001][Das 2003][Mitra 2004]

Permanent Fault Coverage– Power Law



Permanent Fault Coverage Tests



How Many Test Patterns Are Useless?
IEEE VLSI Test Symposium [Ferhani, Saxena, McCluskey, Nigh 2008]

Permanent Fault Coverage Challenges

Test Time < 1 millisecond

Periodic Test Power Usage

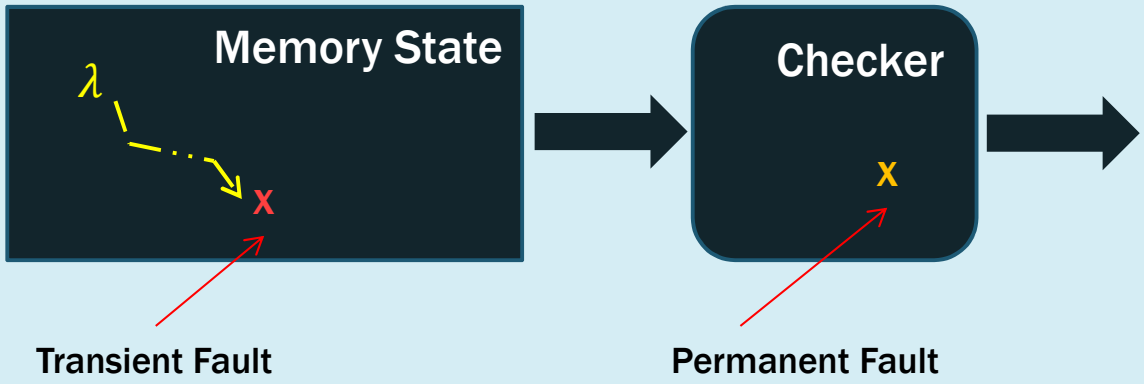
Fast Context Switch

- Run-Time Process and Structural Test

Periodic Software Test as an Alternative

- Solves the Context-Switch Problem
- Coverage Evaluation Still an Issue

Latent Fault Coverage



Percentage of **Fault-Secure** Permanent Faults in the Checker

How to Detect Latent Fault?

- Use Permanent Fault Tests– Works Only During Periodic Tests
- Self-Checking Checker– Works During Run-Time
- If Checker is Software use Algorithm Based Fault Tolerance (ABFT)

Totally Self-Checking Circuits [Andersen & Metze 1973]
[Ashjaee & Reddy 1976] and ABFT [Huang & Abraham 1984]

Machine Learning (ML)

Handwritten Digit Recognition Dataset

5000 Training Examples

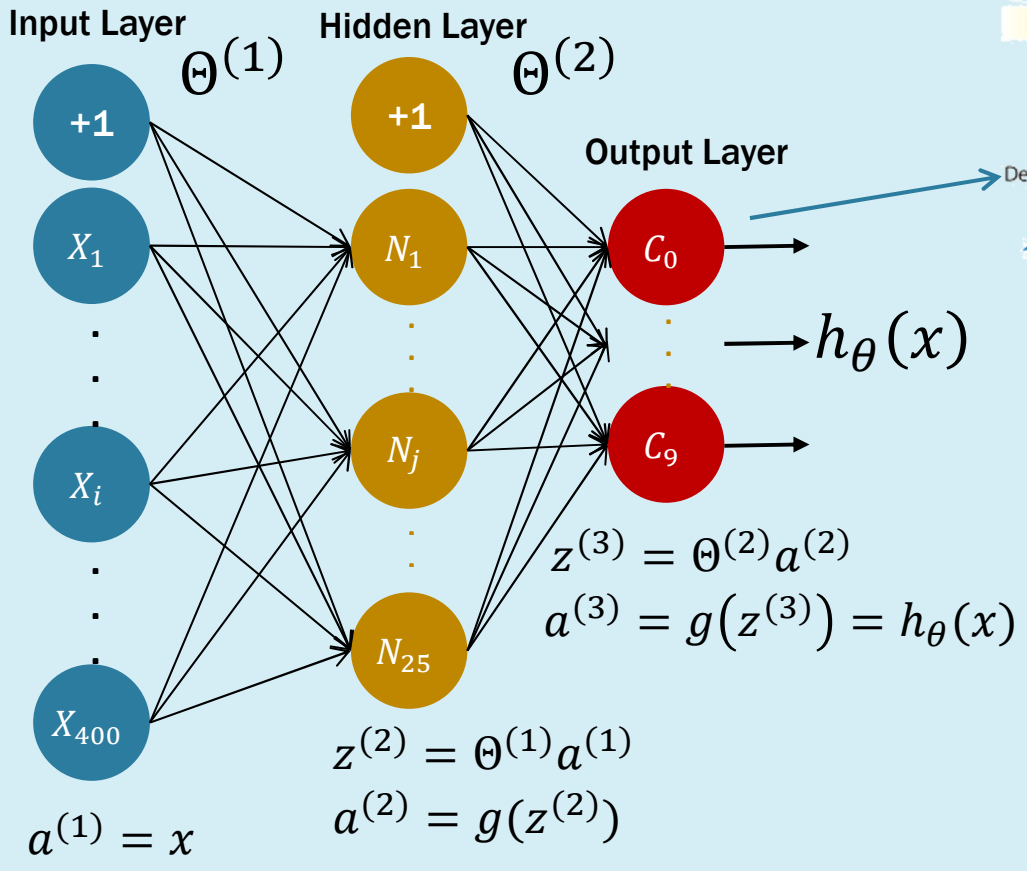
Each Digit 20 x 20 Pixels
- Flattened to 400 Elements

Each Pixel Greyscale Shading
- Floating Point Number

Supervised Learning

Credits: Coursera Machine Learning Class, Prof. Andrew Ng, Stanford University

Handwritten Digit Recognition Neural Network



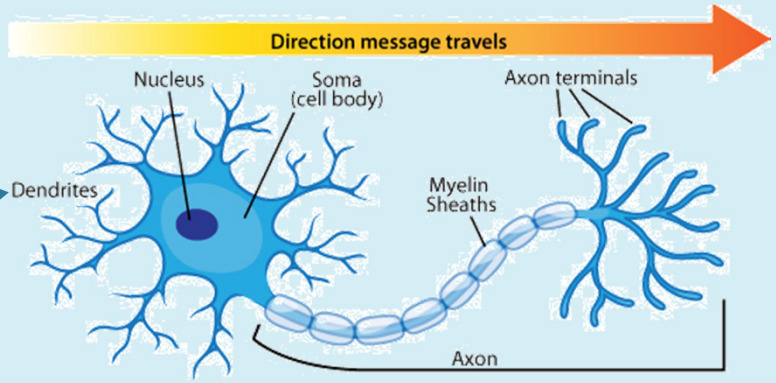
$$a^{(1)} = x$$

$$z^{(2)} = \Theta^{(1)} a^{(1)}$$

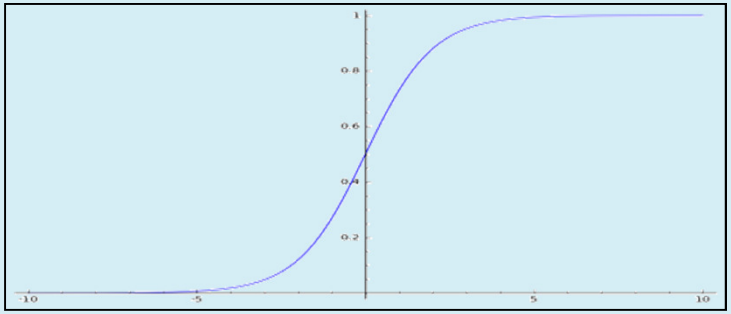
$$a^{(2)} = g(z^{(2)})$$

$$z^{(3)} = \Theta^{(2)} a^{(2)}$$

$$a^{(3)} = g(z^{(3)}) = h_\theta(x)$$



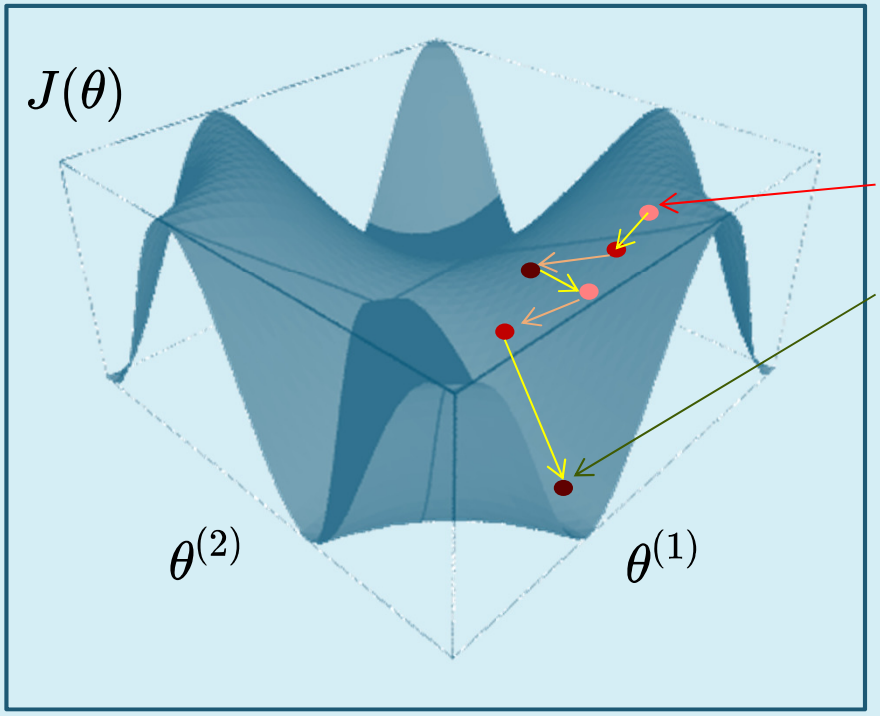
$$sigmoid(z) = g(z) = \frac{1}{1 + e^{-z}}$$



Gradient Descent Algorithm

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m=5000} [-y^{(i)} log(h_\theta(x^{(i)})) - (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))]$$

Cost Function



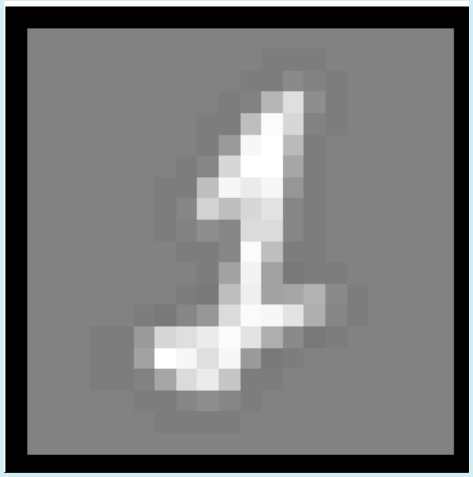
$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$h_\theta(x)$ *far away from* y

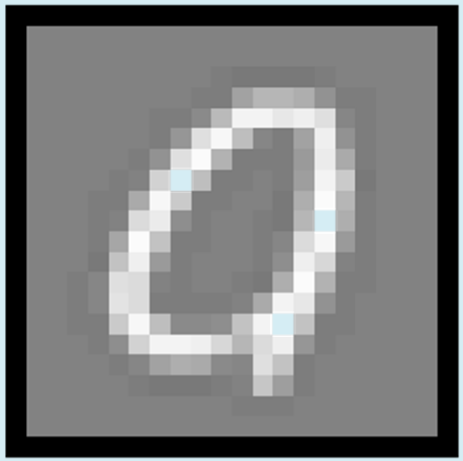
$h_\theta(x)$ *close to* y

50 Iterations, 20mins > 95% Accuracy
400 Iterations, 3hrs > 99% Accuracy

Test Examples– Resilient Learning

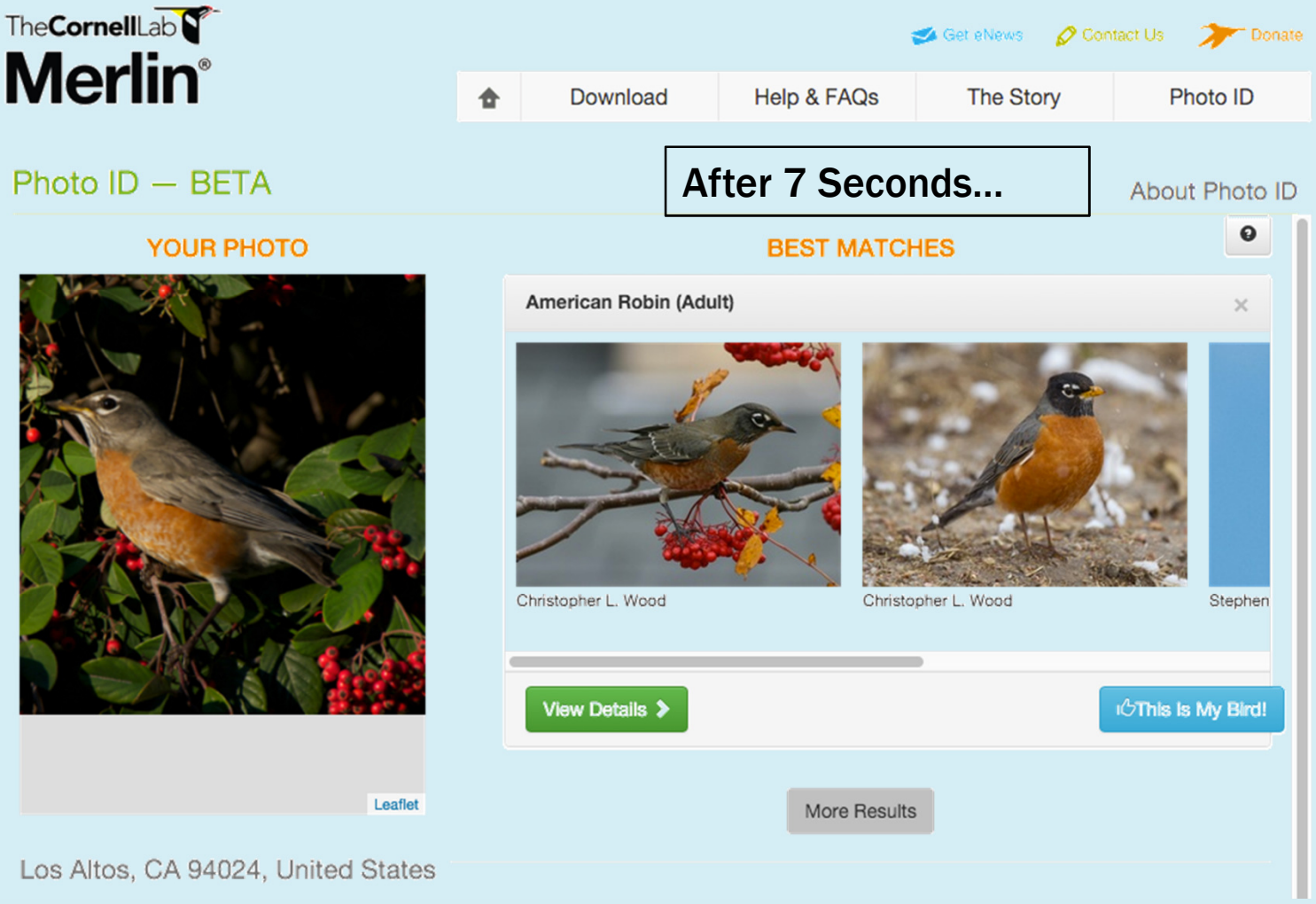


Classified as 2 but Detected as 1

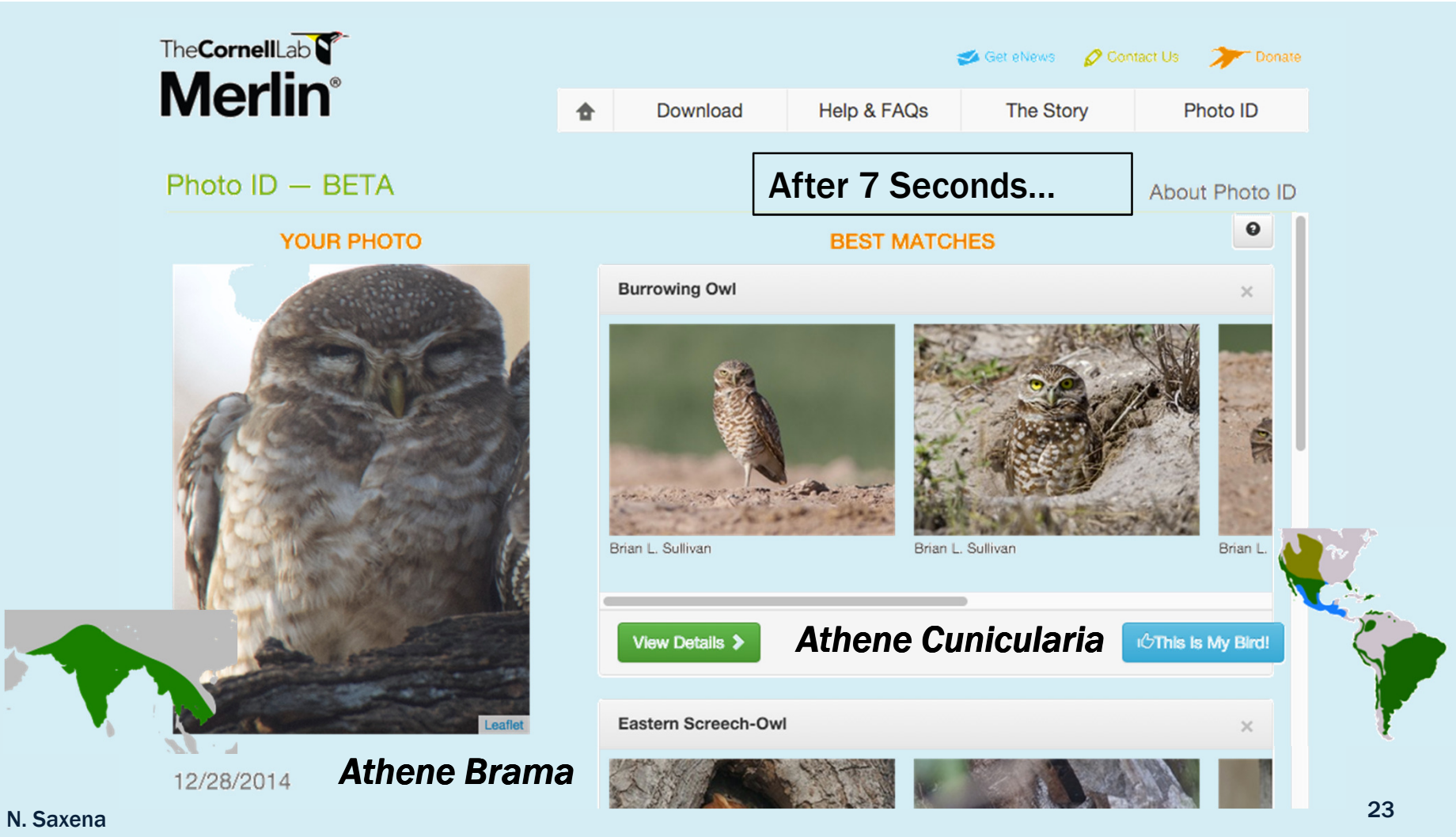


Classified as 9 but Detected as 0

Machine Learning– North American Bird-ID



Spotted-Owlet in Rajasthan– North India



N. Saxena

What Processing Power Per Frame is Needed?

Discounting Network Time

- Image Classification Takes 6 Secs

Merlin Bird-ID Hosted on AWS

- Possibly uses Single Xeon Server

To Classify Image in 33ms

- Need 6000ms/33ms = 180 Xeons…

Supercomputer in a Car

TechRepublic. CXO Innovation Cloud Security Big Data More Newsletters Forums Resource Library Tech Pro Free Trial

HARDWARE

CES 2016: NVIDIA Drive PX 2 supercomputer for self-driving cars like having 150 MacBook Pros in your trunk

NVIDIA plans to put a supercomputer and deep-learning neural network in the truck of every self-driving car.

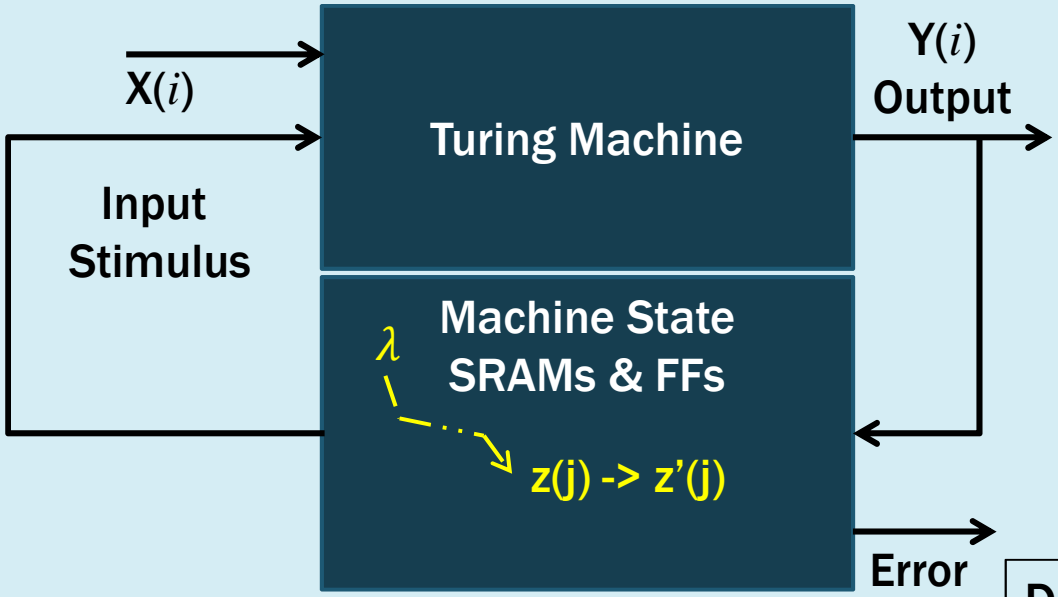
By Bill Detwiler | January 5, 2016, 12:42 AM PST

150 MACBOOK PROS IN YOUR TRUNK

6 TITAN X = 42 TFLOPS, Core i7 = 280 GFLOPS, 42 / 0.28 = 150 MacBook Pros

Road to Resiliency

Architectural Vulnerability Factor (AVF)



SDC– Silent Data Corruption

AVF_{sdc} *if Y corrupt and Error = 0*

AVF_{due} *if Error = 1*

DUE– Detected and Uncorrected Error

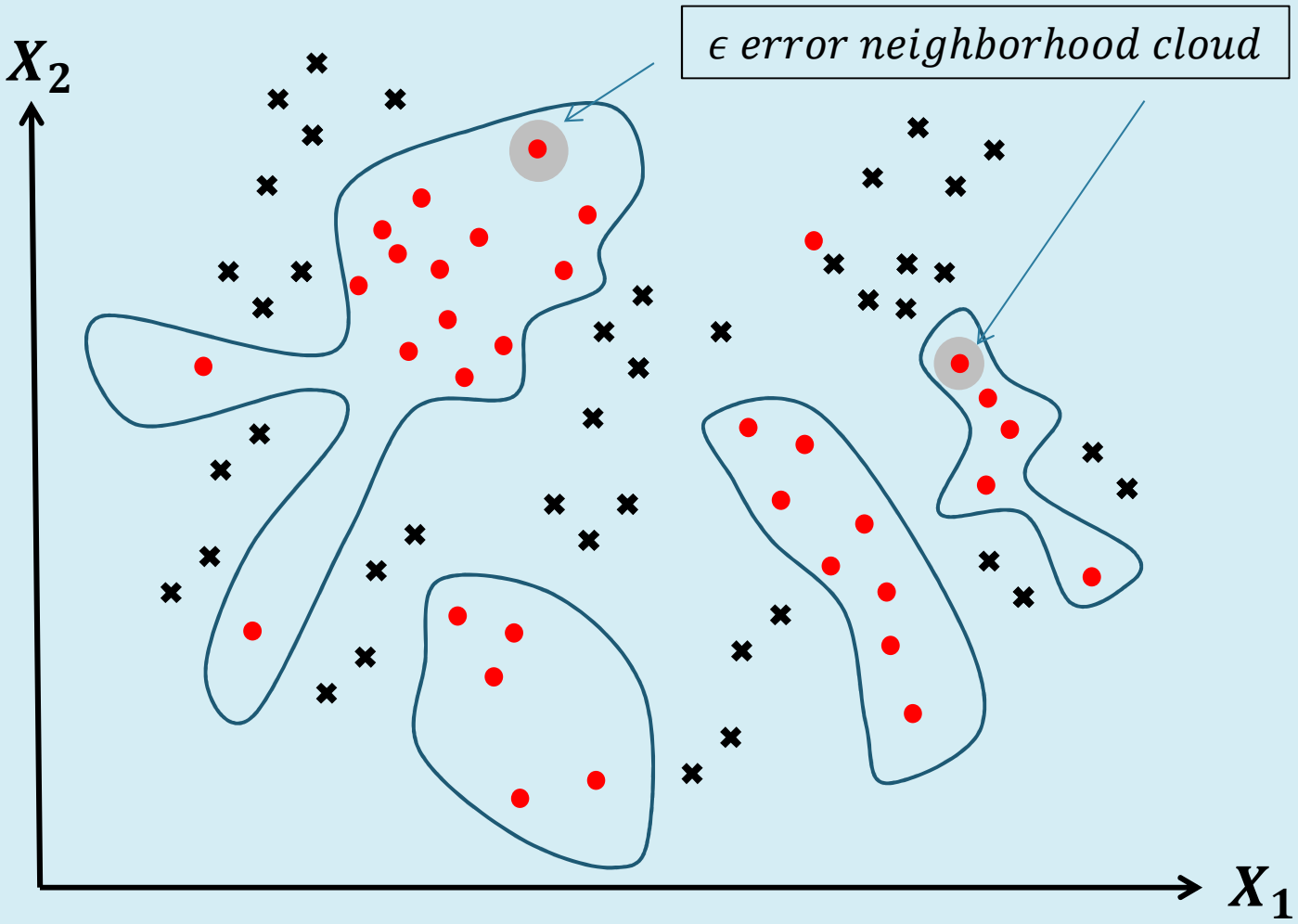
$$AVF \text{ is the Probability of } Y\big(X(i), z(j)\big) \neq Y\big(X(i), z'(j)\big)$$

AVF was first defined by Mukherjee et. al. [MICRO-36 2003]

Related to Signal Probability Problem

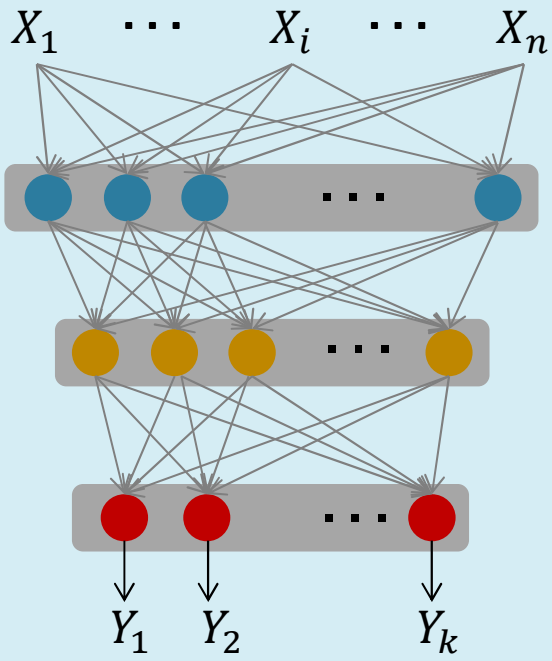
K. P. Parker, E. J. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Trans. Computers*, vol. C-24, no 6,pp. 668-670, June 1975.

AVF for Feature X_i Error– Very Low

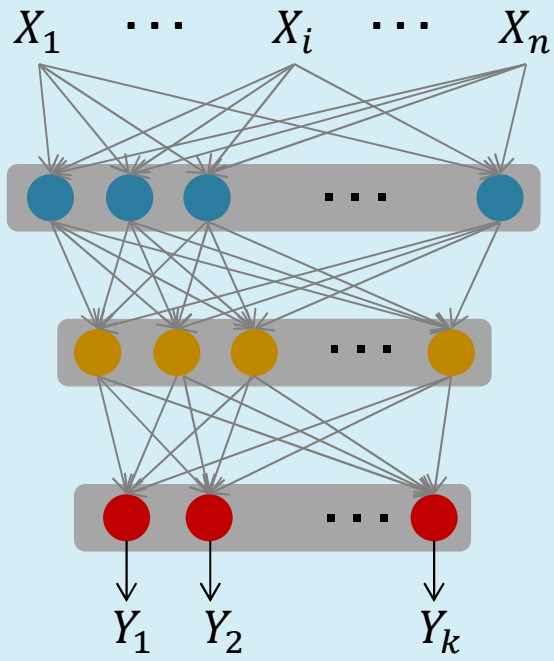


Higher ML Performance through Reduced Precision

$32 - Bit\ X_i\ Features, \Theta\ Weights\ and\ Y_j\ Outputs$



$16 - Bit\ X_i\ Features, \Theta\ Weights\ and\ Y_j\ Outputs$

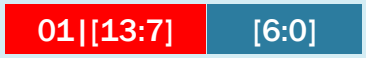


2X Performance Improvement!!

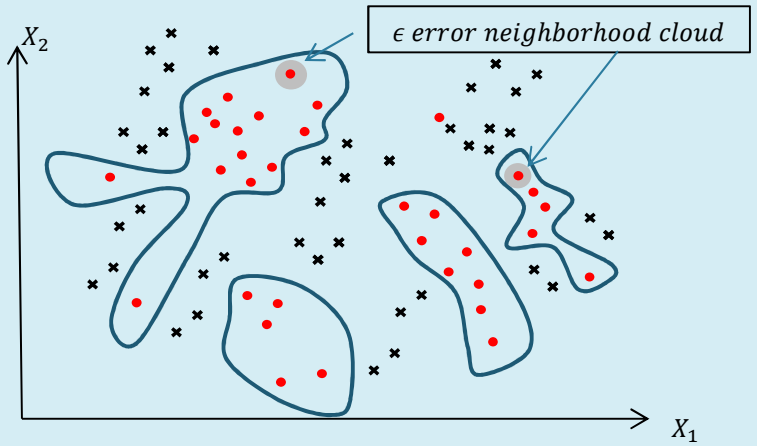
What about ML Resilience when Features and Weights are Scaled?

$32 - bit\ Precision\ and\ X_i > 2^{30}$

$16 - bit\ Precision\ and\ X_i > 2^{14}$



$$\frac{\Delta X_i}{X_i} = \epsilon < 0.01$$



Precision	Vulnerable Bits (Average)	Vulnerable Fraction (Average)	Raw FITs/Word (Relative)	Effective FITs/Word (Relative)
int32	22	68.75%	2	1.375
int16	14	87.50%	1	0.875
fp32	21	65.63%	2	1.313
fp16	12	75.00%	1	0.750

Resiliency Gets Better with Scaled Features & Weights

ML Resilience for Control-Flow Faults?

Neural Networks Implemented as Program Code

Errors in Control-Flow

- Program Counter, Instruction Bits

SDC-AVF in the Range 20% to 40%

Requires Parity Protection & Self-Checking Code

Recovery Strategy– Detect and Retry

- Works for Transient Errors

Redundant Execution Needed (Internal Redundancy)

Detect & Retry Does Not Work for Permanent Faults

Error Signals Still Needed

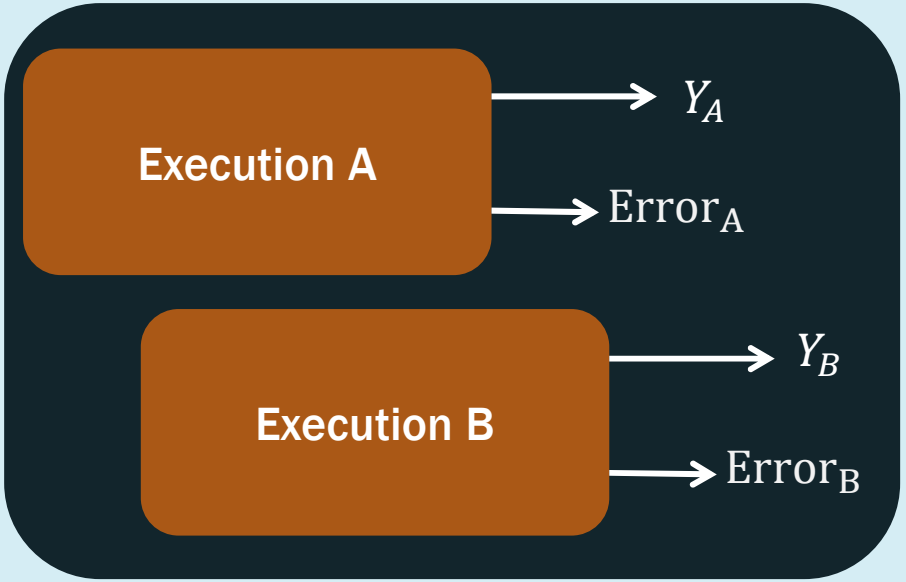
- Single-Point Fault Tolerance

Similar to Erasure Codes

- Mirrored RAID
- Identify Correct Copy

Execution Instances

- On Non-Overlapping Hardware



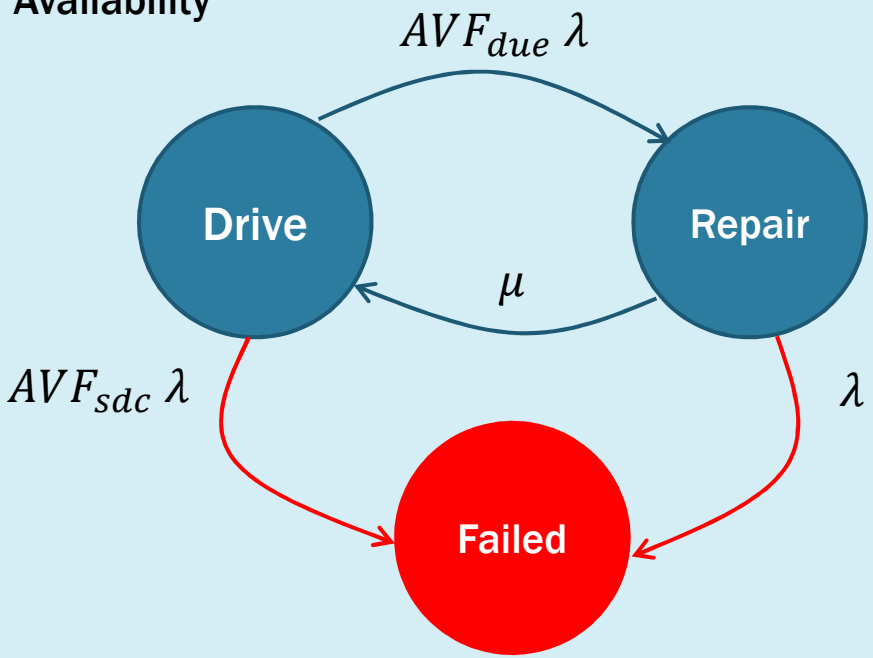
Markov Chain Analysis (Need External Redundancy)

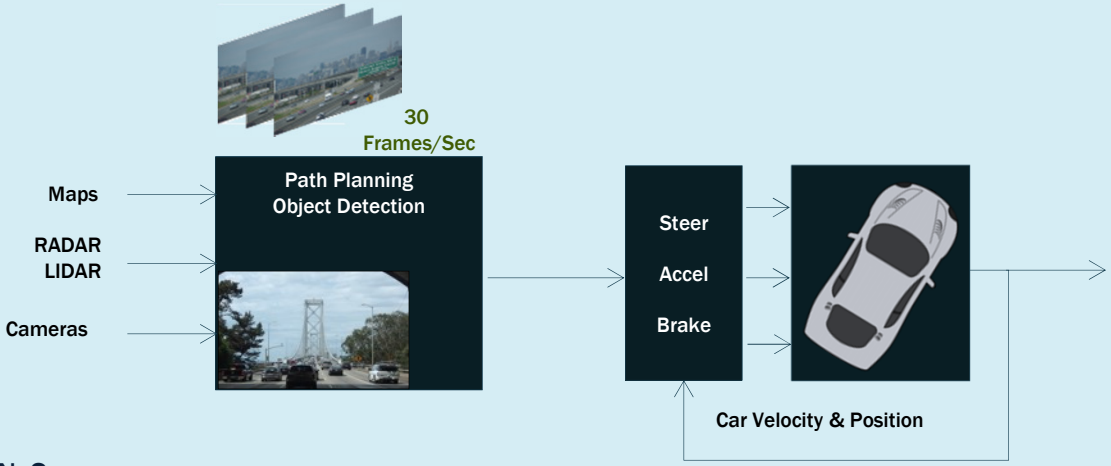
ISO26262 Does Not Have Any Requirements on Availability

For Driverless Car
Loss of Frames => Loss of Life

For 3 Frame-Tolerance, Need

$$\frac{1}{\mu} < 100ms$$





Prof. K. S. Trivedi

N. Saxena

Dual Redundant System

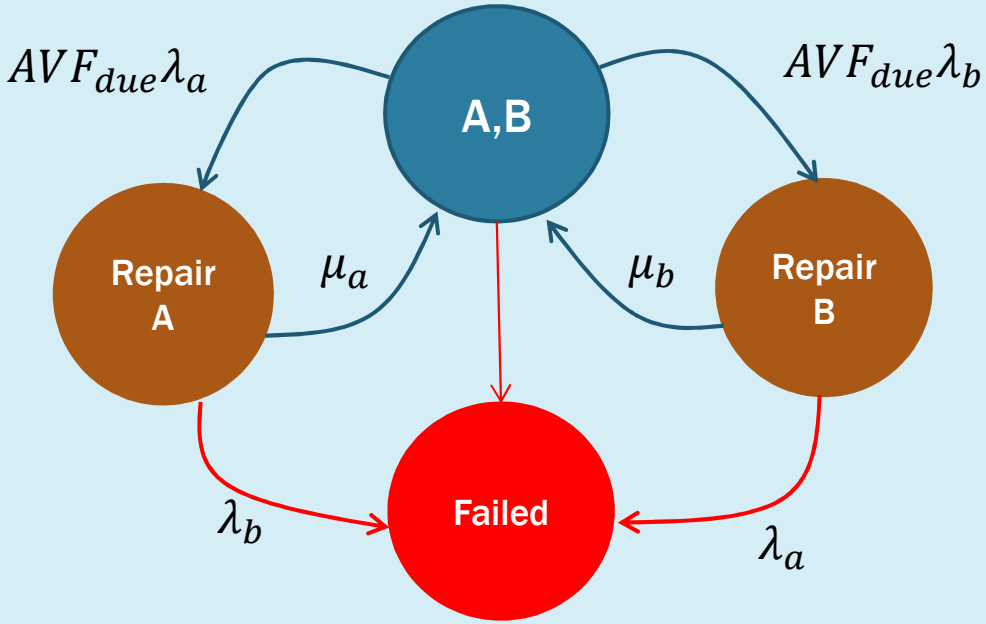
Relaxed Constraints on Repair Rate

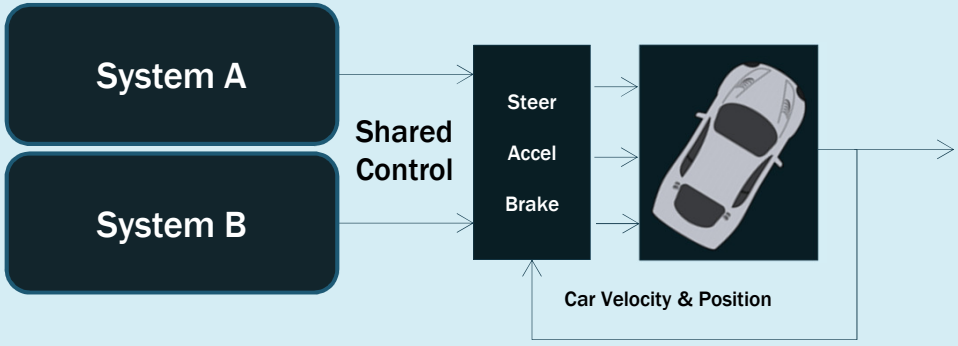
$$\frac{1}{\mu_a} < \frac{1}{\lambda_b}$$

$$\frac{1}{\mu_b} < \frac{1}{\lambda_a}$$

$\frac{1}{\lambda_a}$ or $\frac{1}{\lambda_b}$ **in the order 1000's of hours**

Repair can wait till the next Key-Off Event





What is the Current FIT Rate for Systematic Faults?

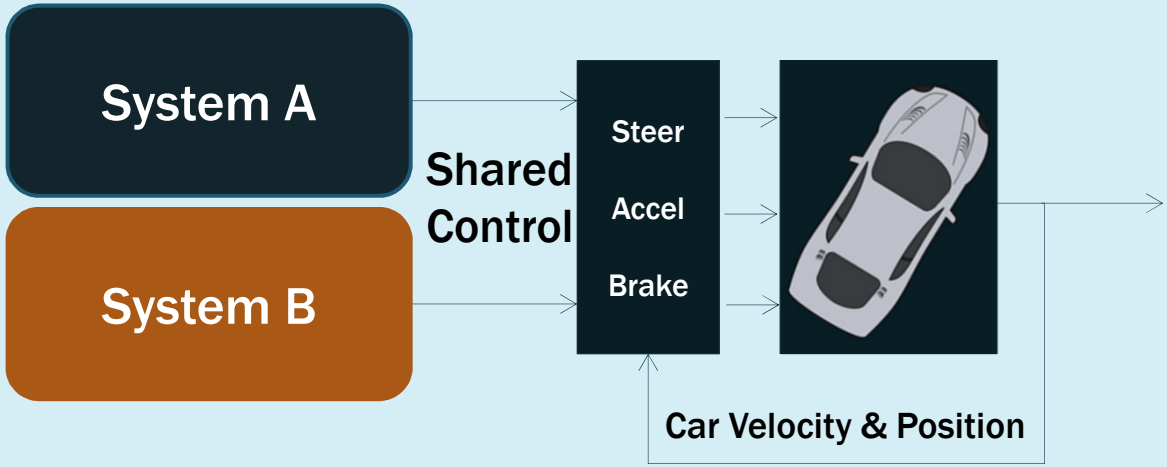
Systematic Faults	Observed Bug Rate	FIT Rate
Hardware Design Faults	3 Bugs in 48 Years	7000
Software Design Faults	1 Bug Every Year	100000

Mitigating Factors

Automotive Environment is More Constrained

- Hardware Design Quality– Need **Three Orders** of Improvement
- Software Design Quality– Need **Four Orders** of Improvement

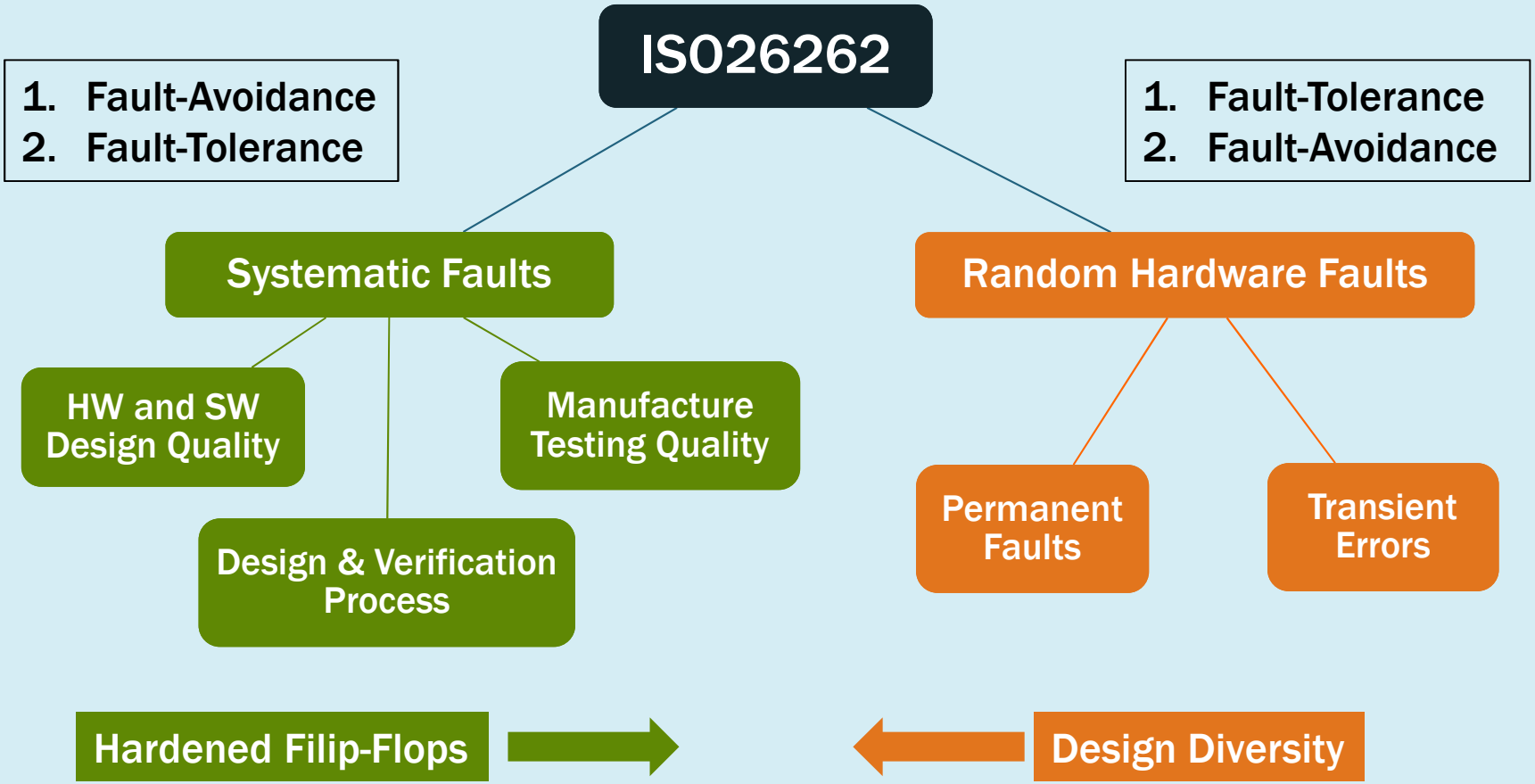
Design Diversity



Coping with Systematic Hardware and Software Design Errors

[Siewiorek et. al. 1978] (byte reversal copies C.mmp processor)
[Sedmak and Liebergot 1980] (complementary function diversity in VLSI)
[Chen and Avizienis 1978] (N-version programming, SIFT software implemented fault-tolerance)
[Horning et. al 1974] (Recovery Blocks) [Patel] RESO Technique
[Amman and Knight 1987] (Data Diversity)
[McCluskey, Saxena, Mitra 1998] Diversity for Reconfigurable Logic & Quantifying Diversity

My Perspective on ISO26262



Conclusions

Road to Resiliency ⇒ Dual Redundancy

- Mitigates Permanent Fault Testing
- Relaxes Repair Time Requirements

Systematic Faults

- Rigorous Testing and Validation
  - Need 3-to-4 Orders of Improvement
- Design Diversity