# A Look at the Hardware:
## what can approximation buy us, and how can we cash in?

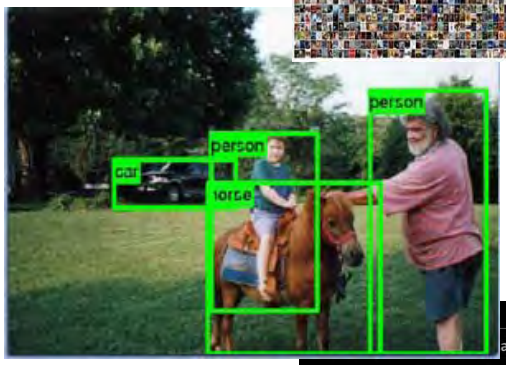

**Naveen Verma**
nverma@princeton.edu

**April 3, 2016**

# Why are we talking about approximation?

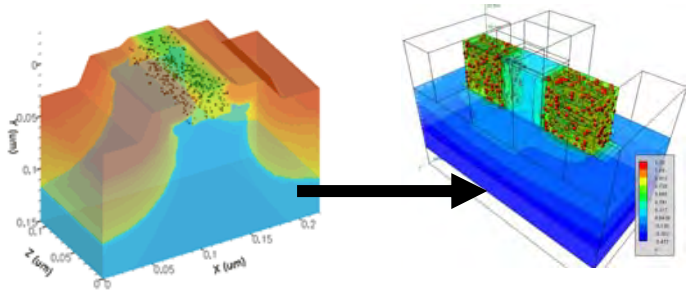
## A CONVERGENCE:

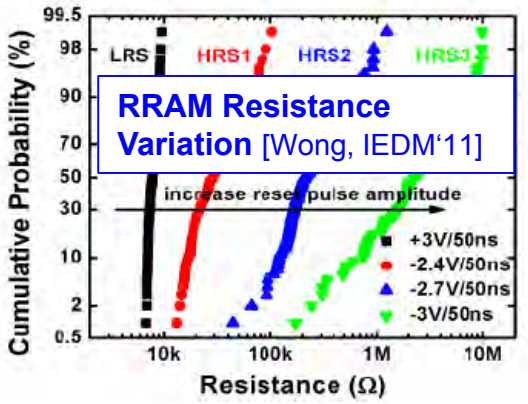
**Applications are statistical…**

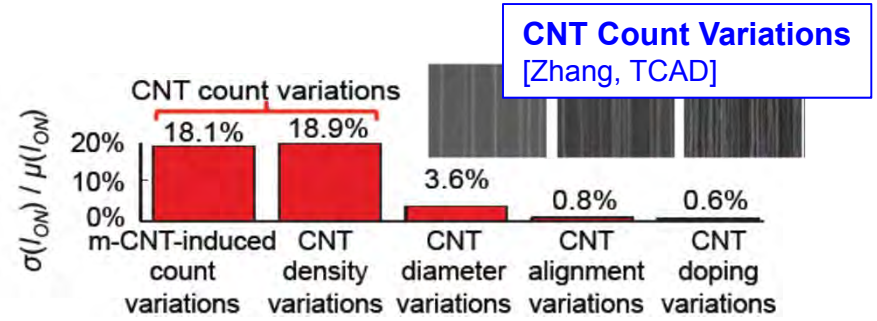





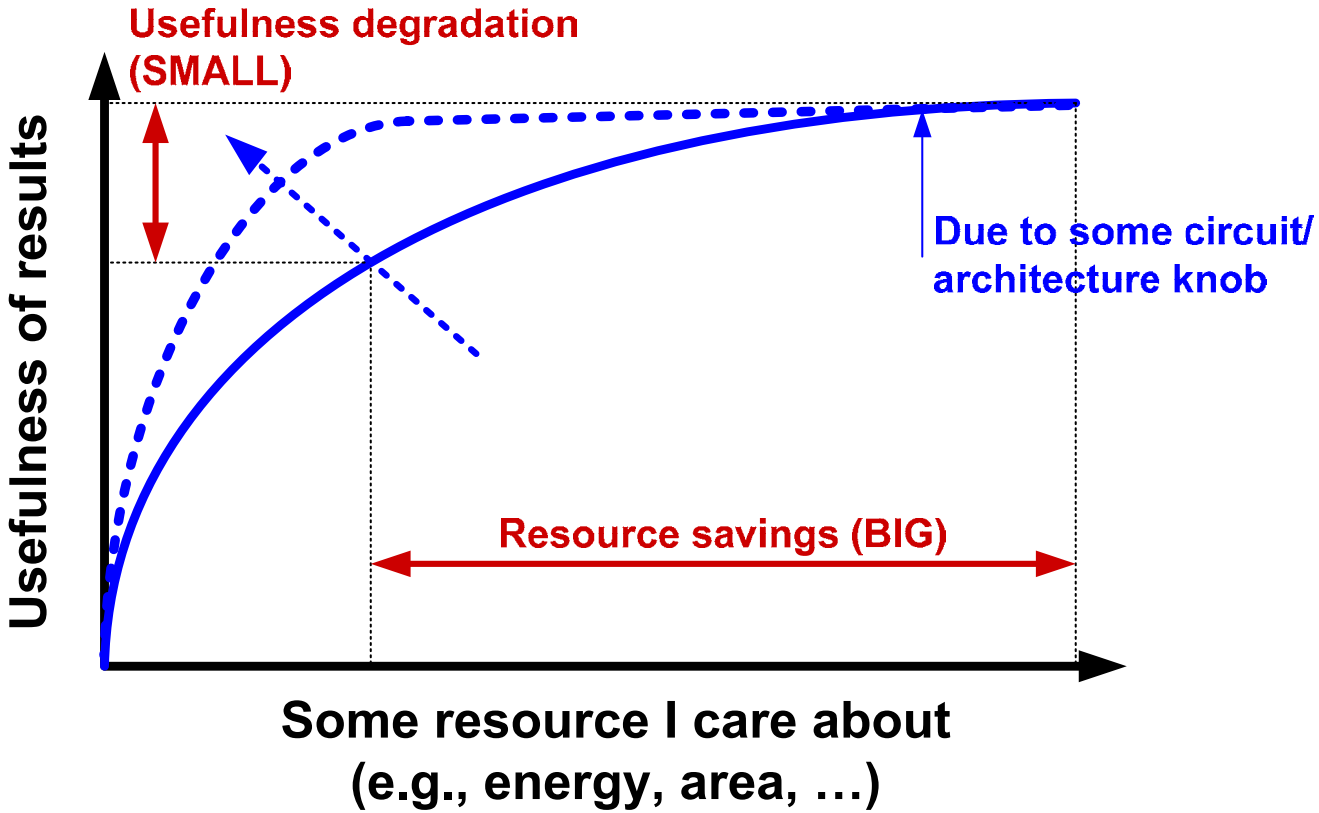
**…Devices are statistical**





# Why are we <u>really</u> talking about approximation?




**<u>Research questions:</u>**

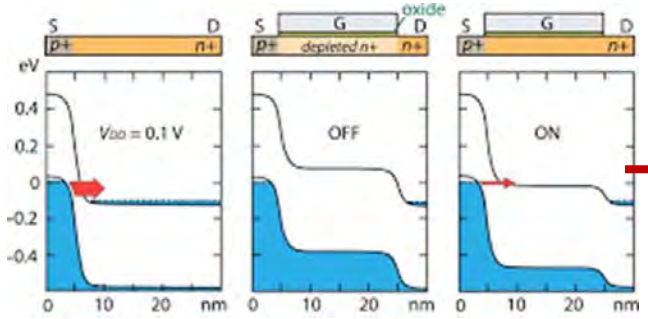
**0. What knobs/mechanisms actually give us such tradeoffs?**

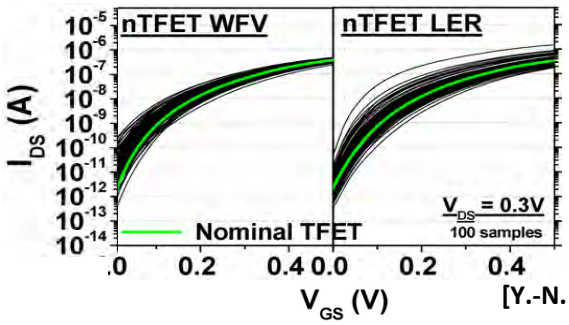
**1. How can we enhance the tradeoffs, by exploiting attributes of applications?**

**2. What architectures derive maximum leverage from such tradeoffs**

# There are knobs at all levels

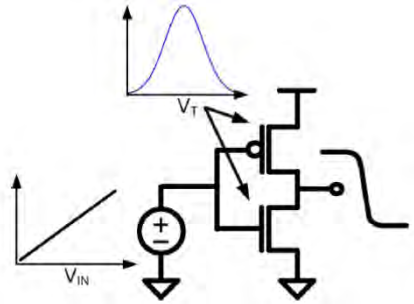
- **Device level (e.g., TFET)**

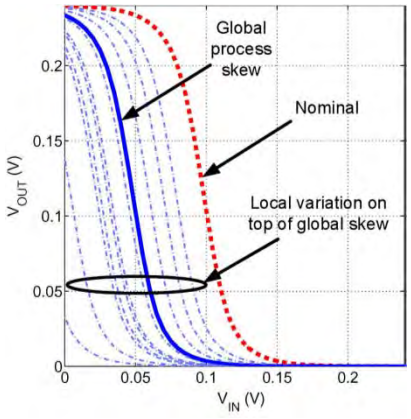


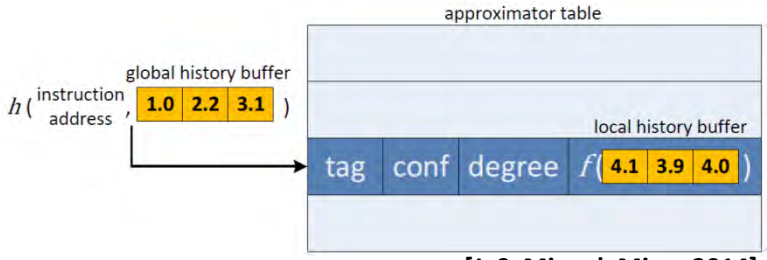

[Y.-N. Chen, JLPEA 2015]

- **Circuit level (e.g., $V_{DD}$ scaling)**

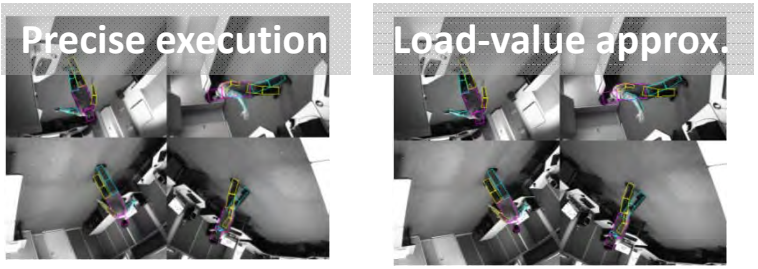



- **Algorithmic level**

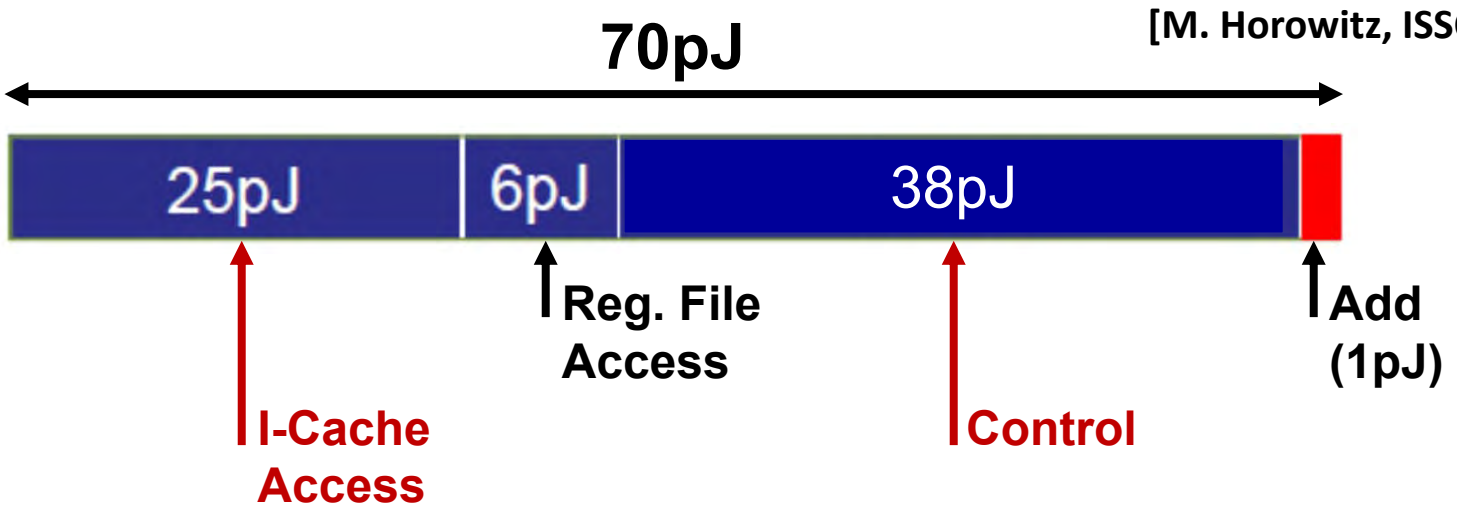



[J .S. Miguel, Micro 2014]



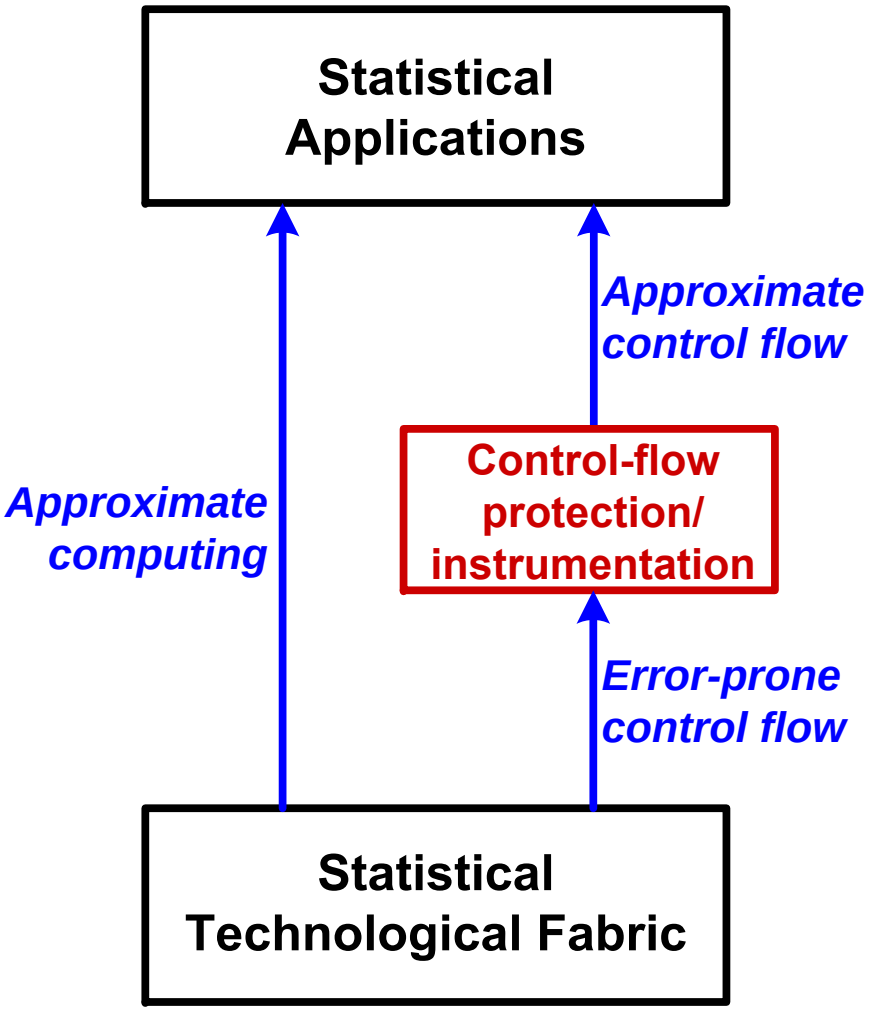
# Before going any further…

## Ex. Energy breakdown of ADD instr (45nm):

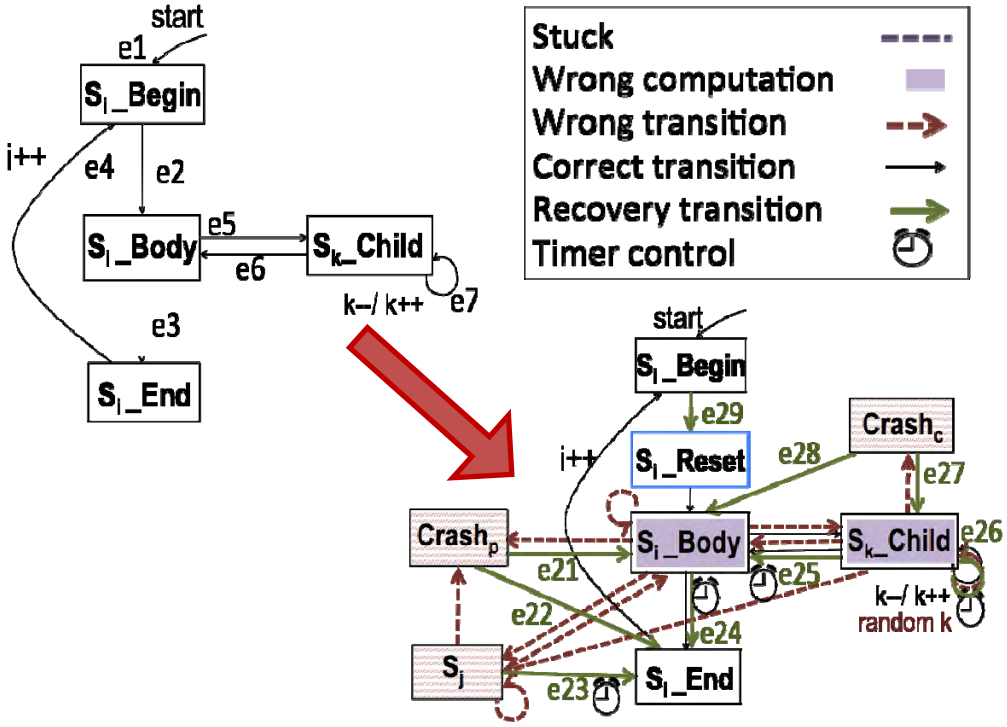



- Instruction fetch and control consumes 90% of energy
- Programmable control does NOT benefit from statistical applications (directly)
- Think accelerators (fine-/coarse-grained?)…

# Aside: approximate control flow




**E.g., YMM, DATE'2013:**

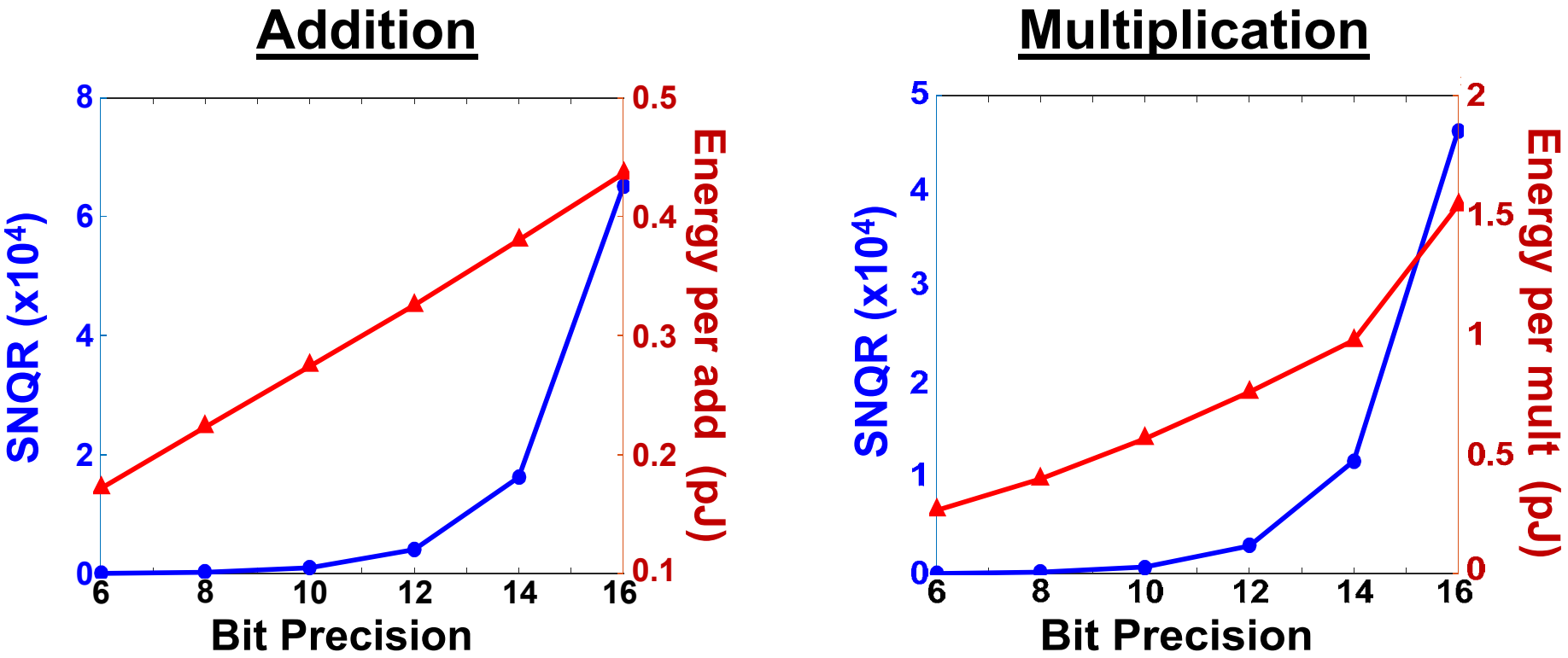



**Formal guarantee of:**

1. **Progress**
2. **Ephemeral effect of errors**
3. **Execution of essential states**

(courtesy S. Malik, A. Golnari)

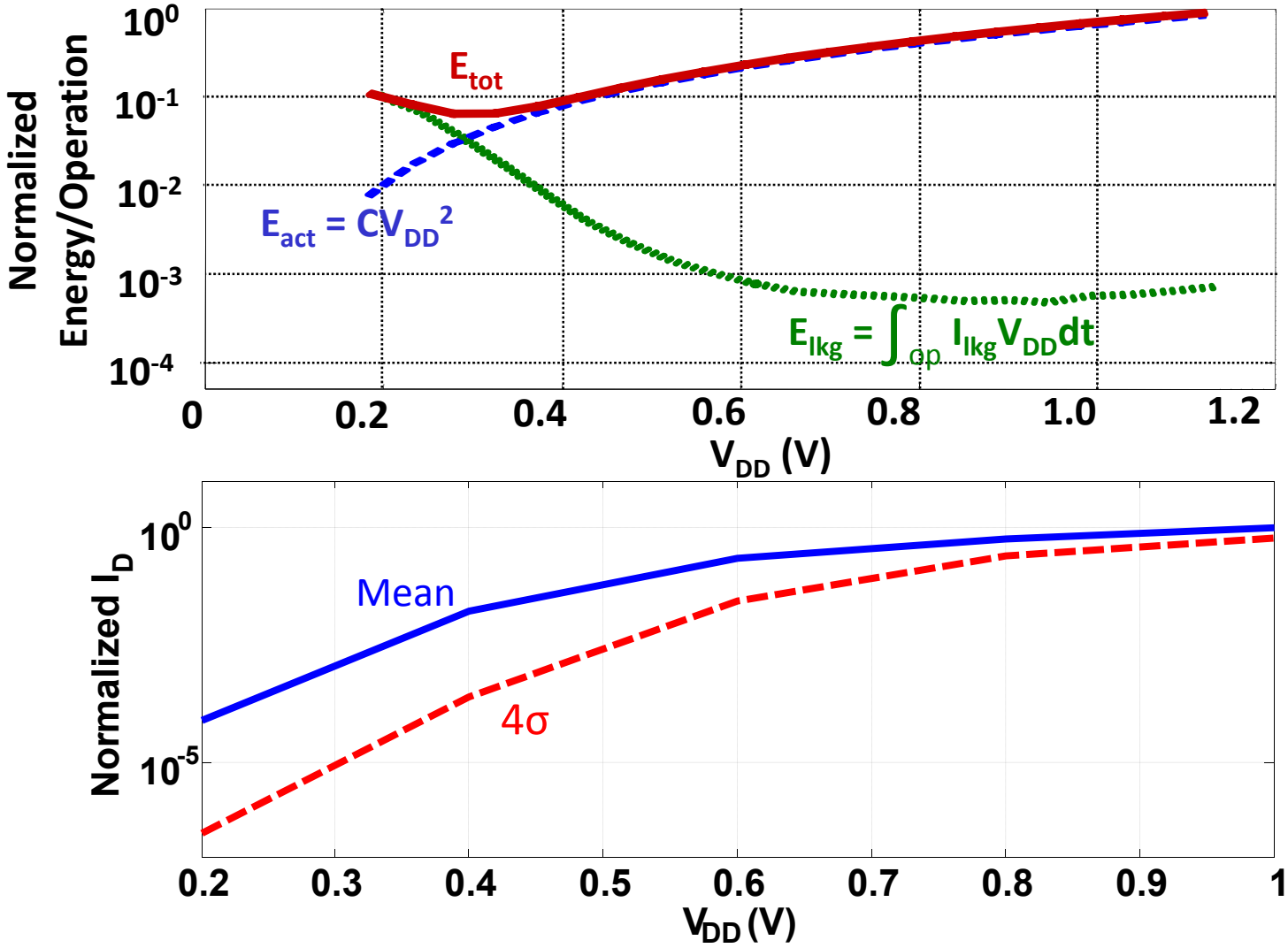
# Where is the beef: bit precision?




(Energy from 32nm CMOS, SQNR from uniformly distributed inputs)

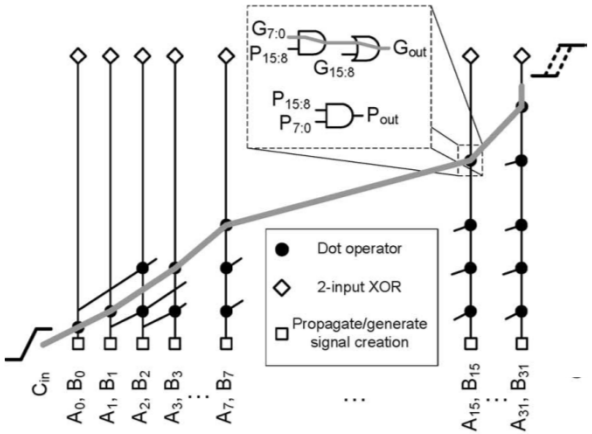
# Where is the beef: $V_{DD}$ scaling?

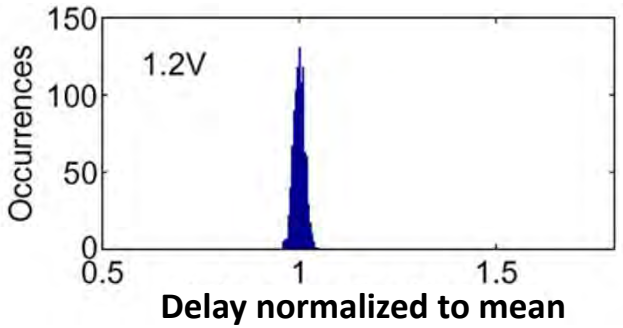
## Ex. Carry-lookahead adder (65nm):

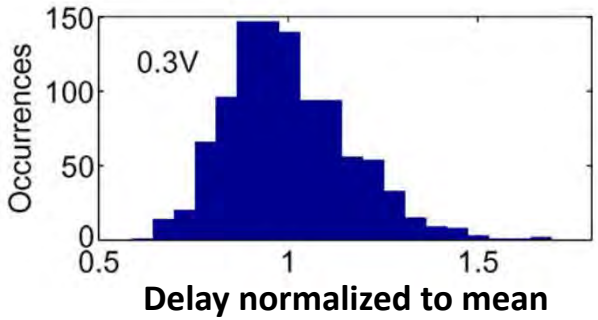


# $V_{DD}$ scaling

## 1. Timing-path variation (65nm)

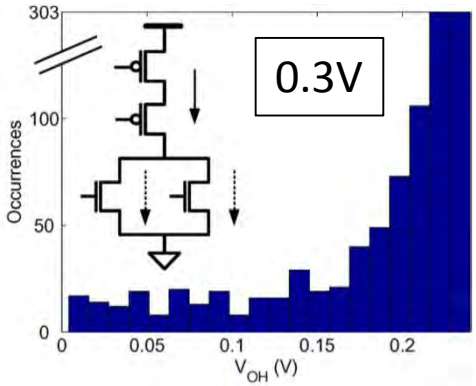


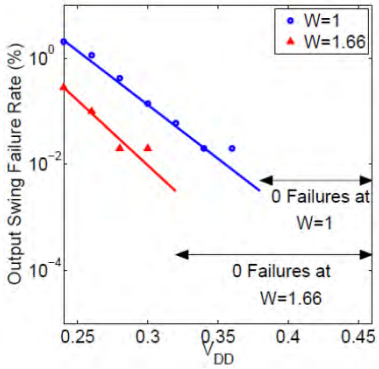




## 2. Noise-margin violations

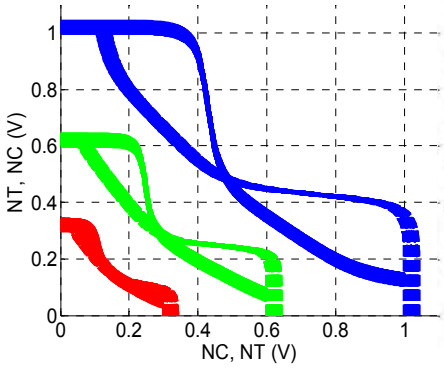
### Logic (65nm)

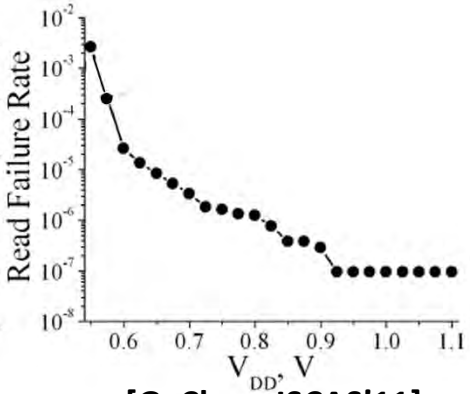



[J. Kwong, ISLPED'06]

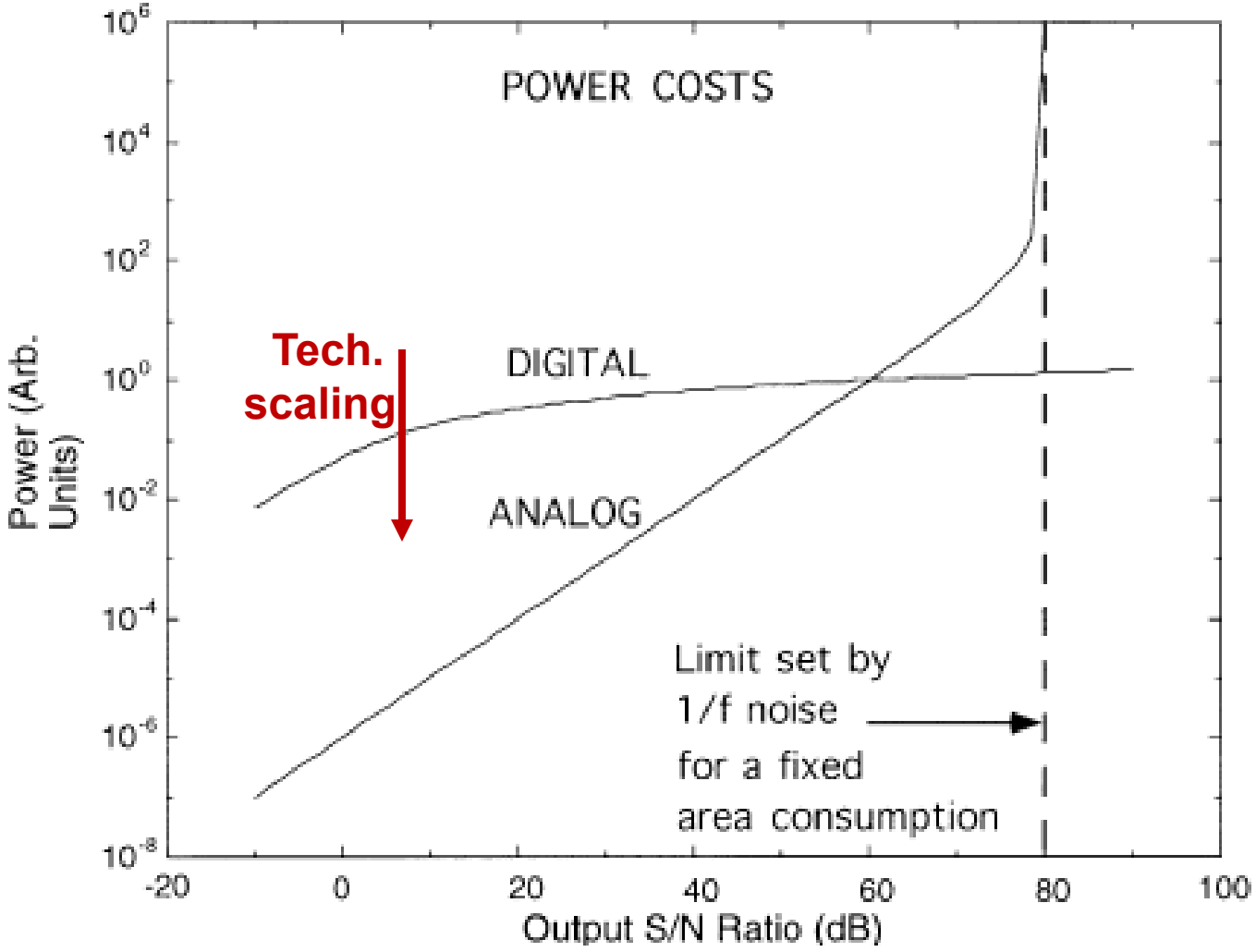
### SRAM (45nm)




[G. Chen, ISCAS'11]

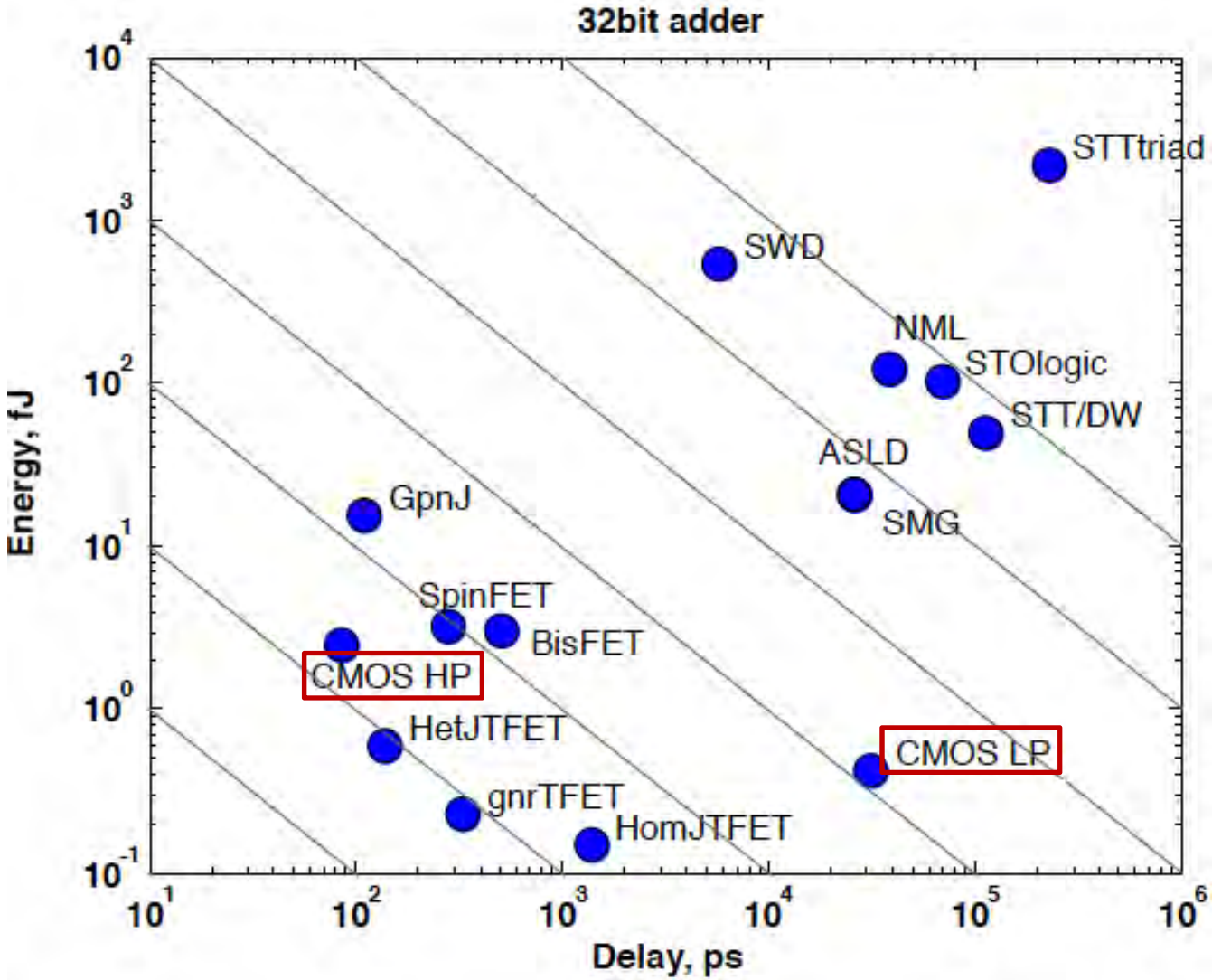
# Where is the beef: analog computation?






R. Sarpeshkar, *Analog vs. Digital: extrapolating from electronics to neurobiology,* 1998

# Where is the beef: beyond Si CMOS?




[D. Nikonov, IEEE Proc.'13]

# So, where *is* the beef?

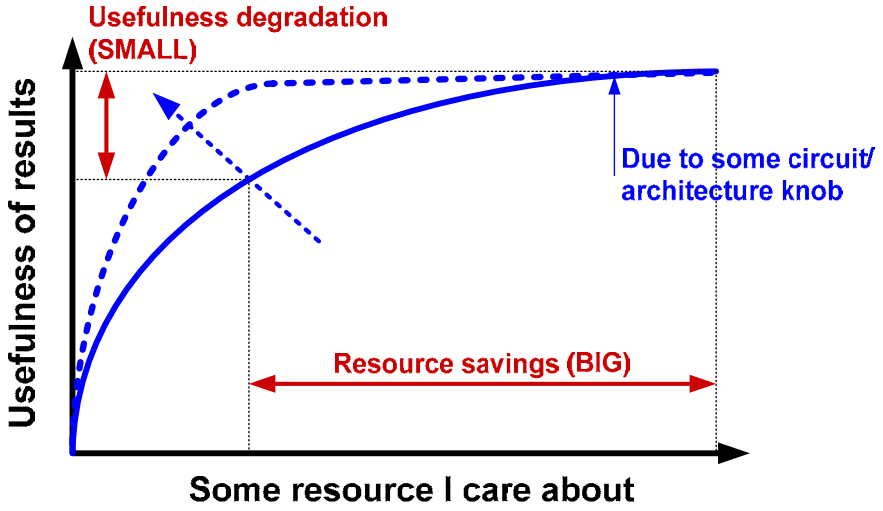
**A: at the *SYSTEM* level**

**E.g., leverage points in system design:**

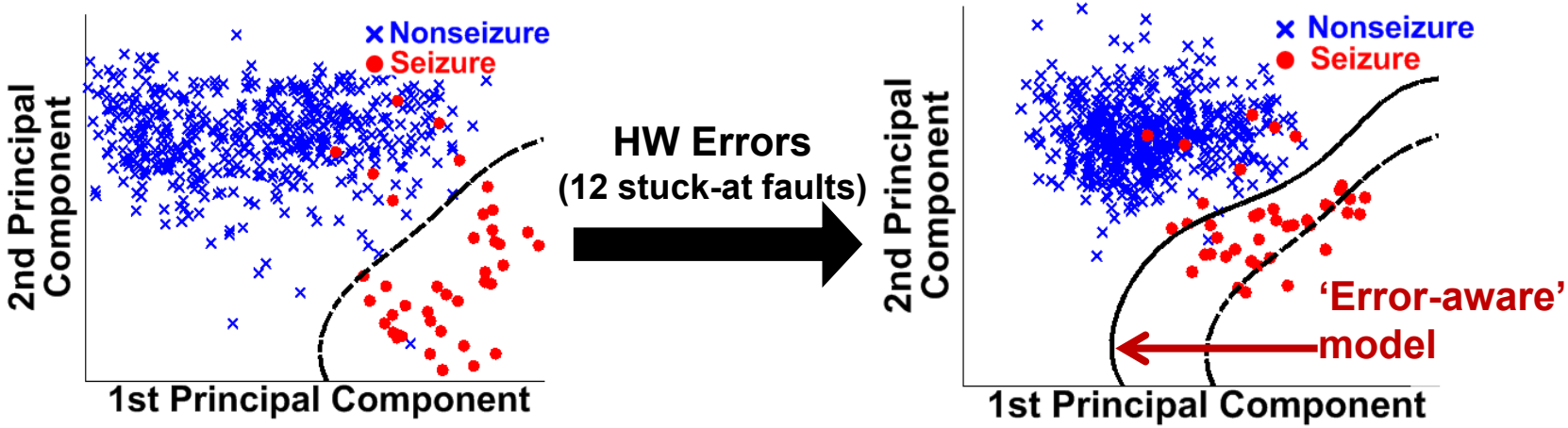
1. **Instrumentation & data conversion**
   - E/ADC (12b, 45nm): ~200pJ
   - E/MAC (12b, 45nm): ~1pJ
2. **Memory accessing**
   - E/SRAM (32kB, 45nm): ~20pJ
3. **General-purpose (vs. heterogeneous) computing**
   - 100-1000× computational energy reduction

**The job of architecture design is to get the most out of device/application attributes, at the *system level***

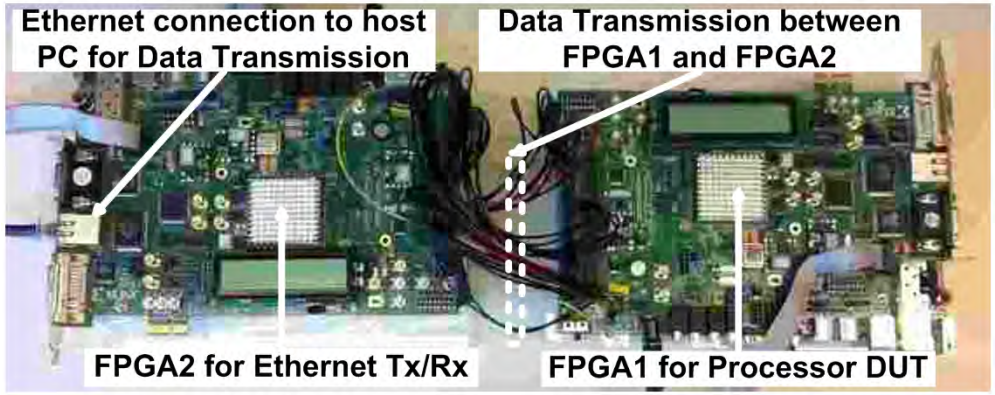
# Before thinking about architectures…

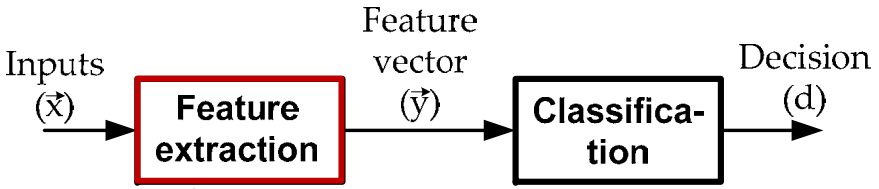



**Data-Driven Hardware Resilience (DDHR)** [Z. Wang, IEEE TVLSI'14]



# Analyzing DDHR: enhancing the tradeoff

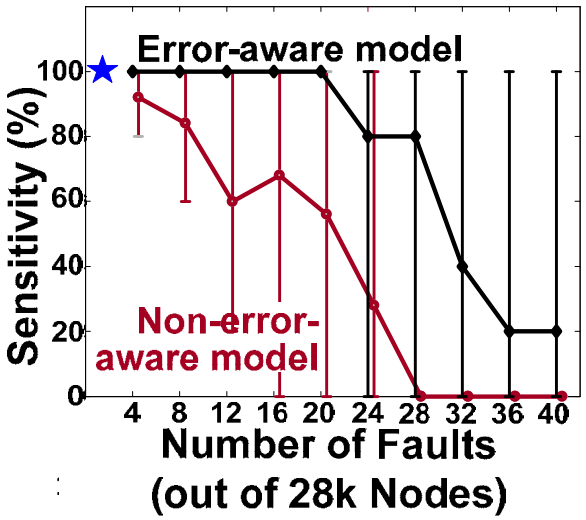


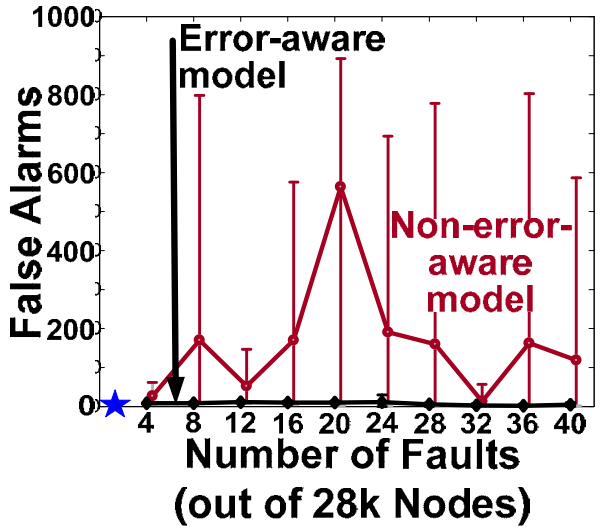



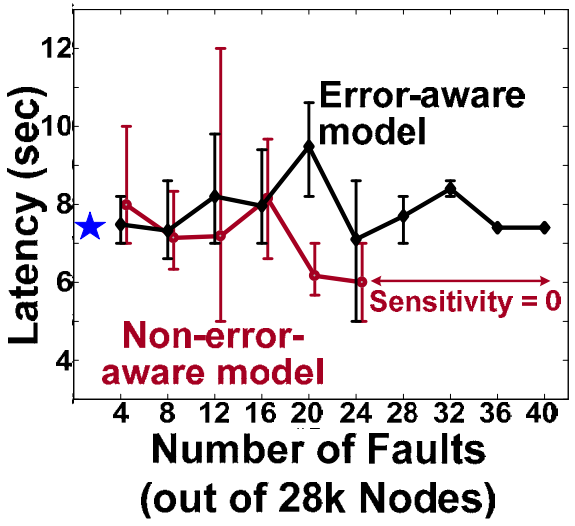

**Inject 'stuck-at' faults randomly on nodes, and…**

1. **scale error rate**
2. **perform multiple runs at each rate**

## Ex. EEG-based Seizure Detector



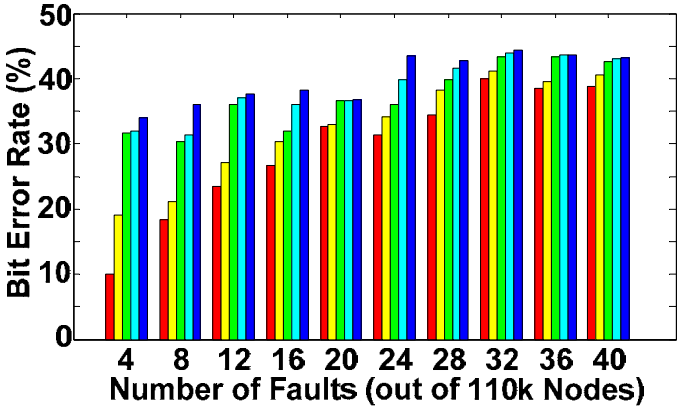



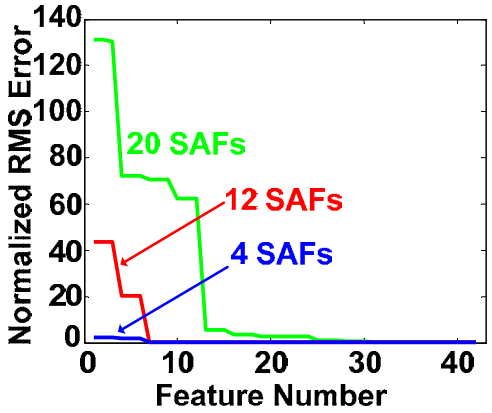



# Analyzing DDHR: enhancing the tradeoff

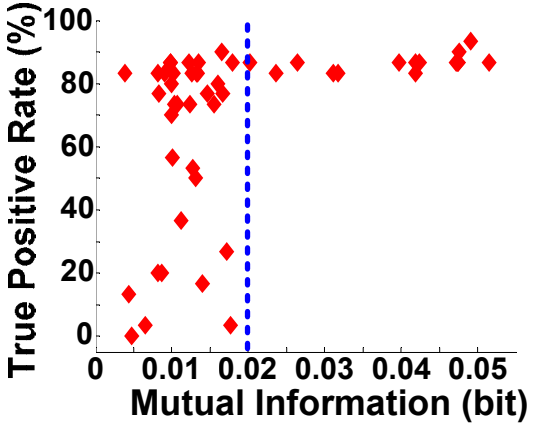
## Ex. EEG-based Seizure Detector:

### Bit-Error Metrics



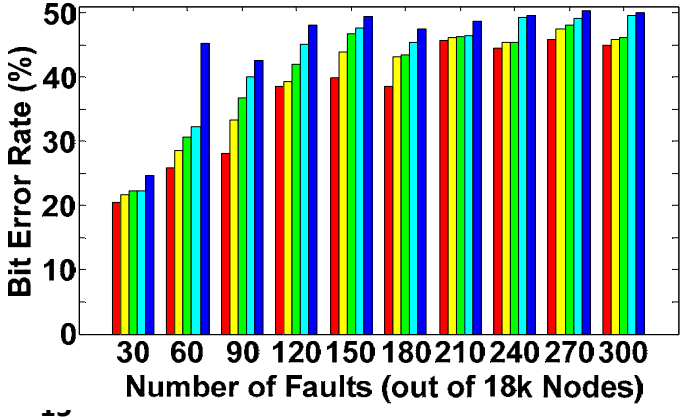



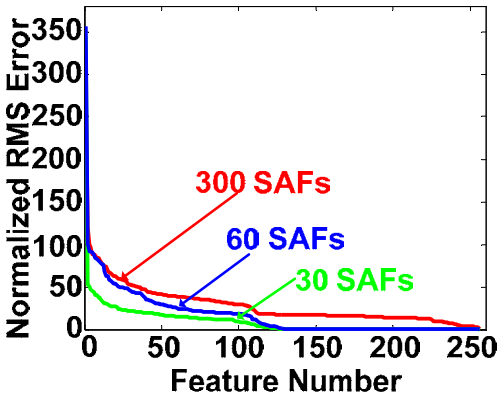

### Information Metric



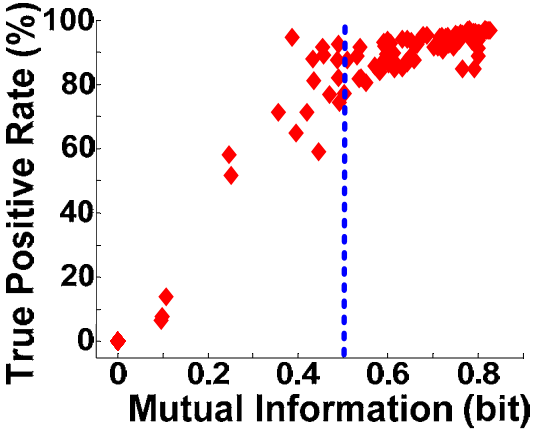

## Ex. ECG-based Arrhythmia Detector:

### Bit-Error Metrics

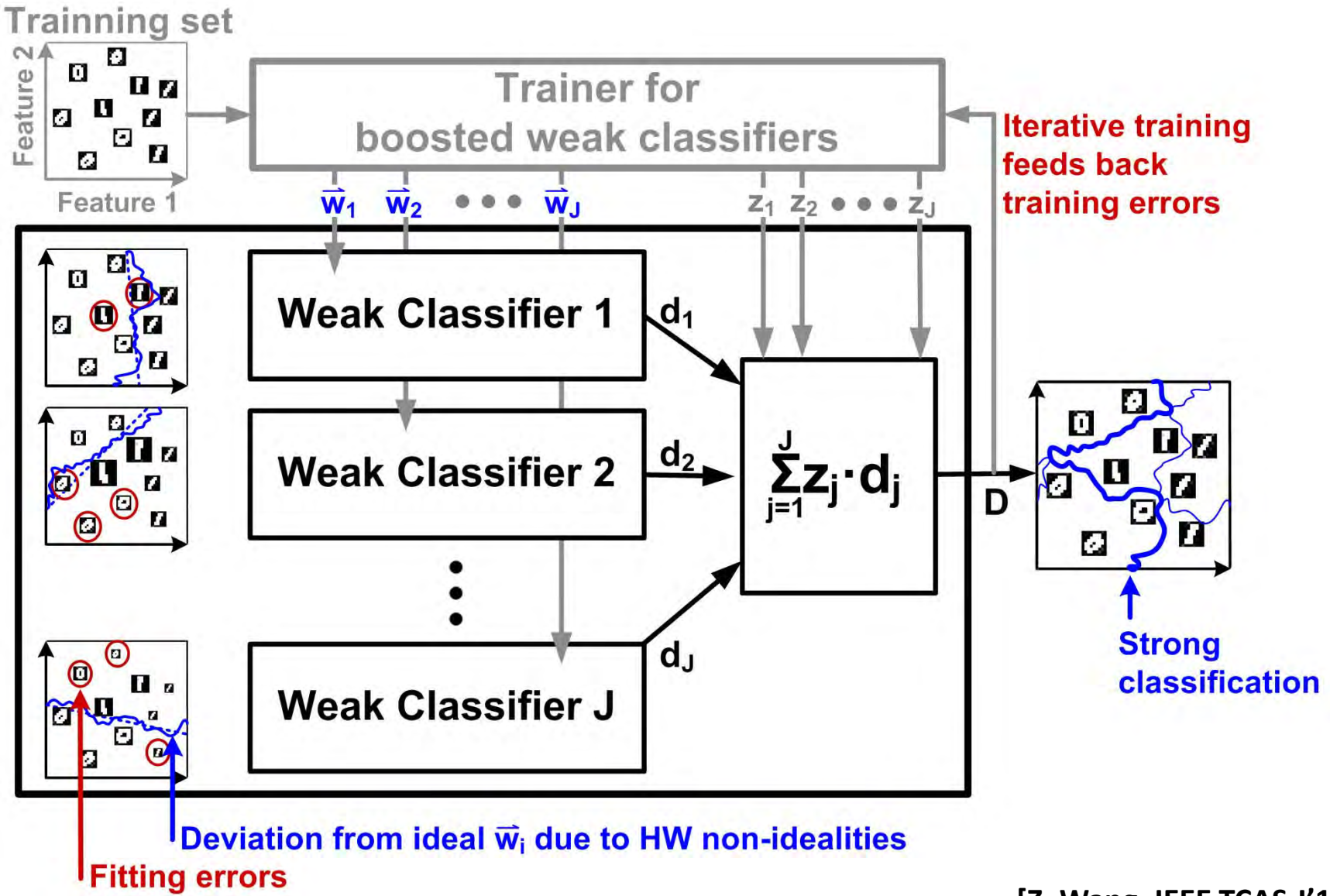



### Information Metric



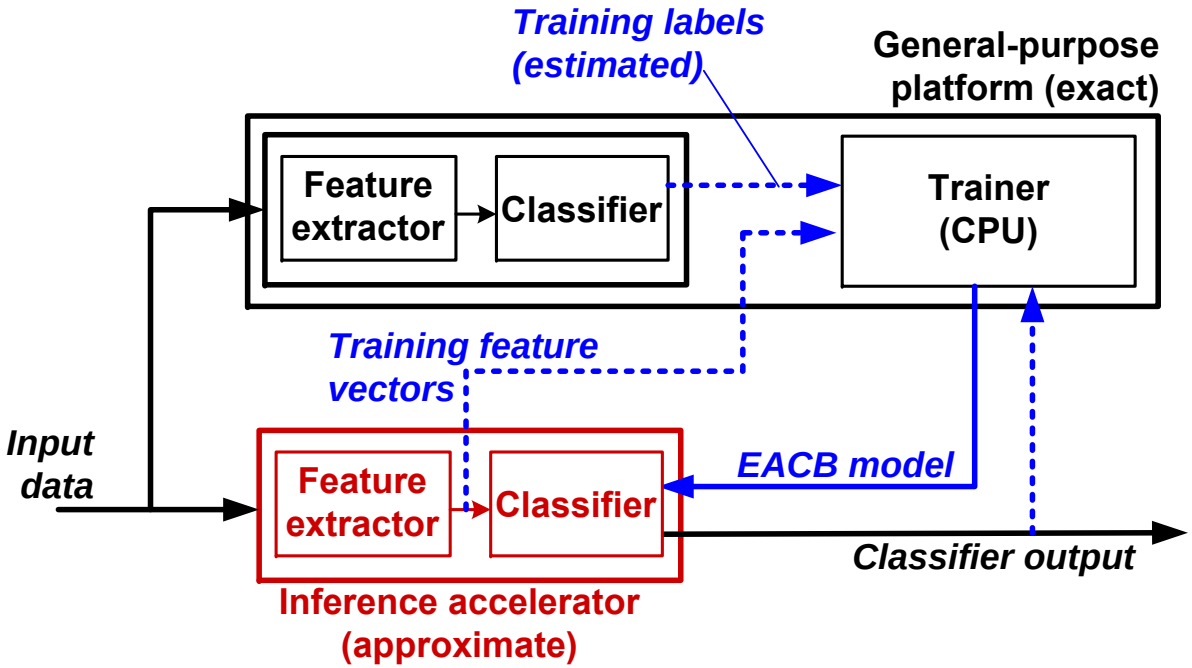
# Error-adaptive classifier boosting (EACB)



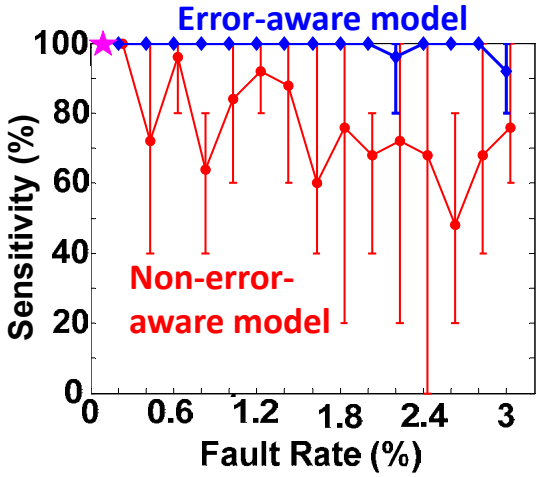


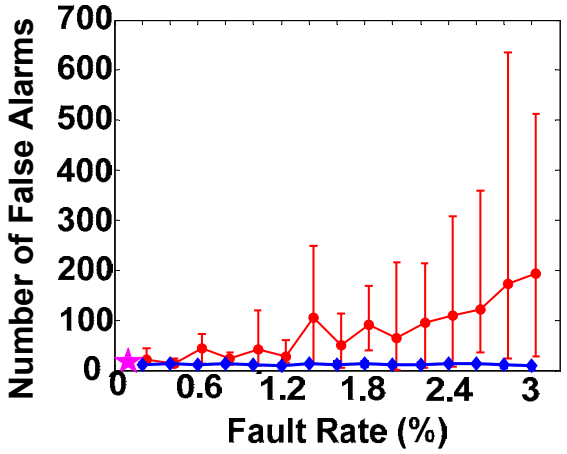

[Z. Wang, IEEE TCAS-I'15]

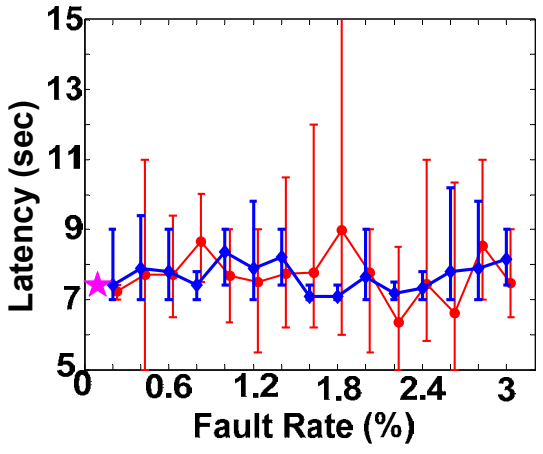
# Training the error-aware model




## Ex. EEG-based Seizure Detector (FPGA emulation): [Z. Wang, IEEE TCAS-I'15]

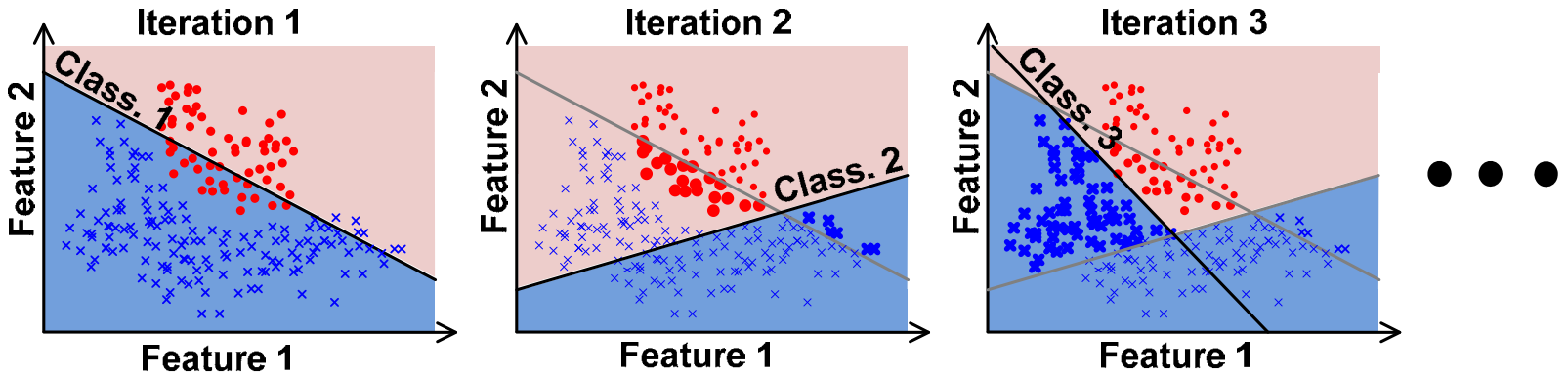


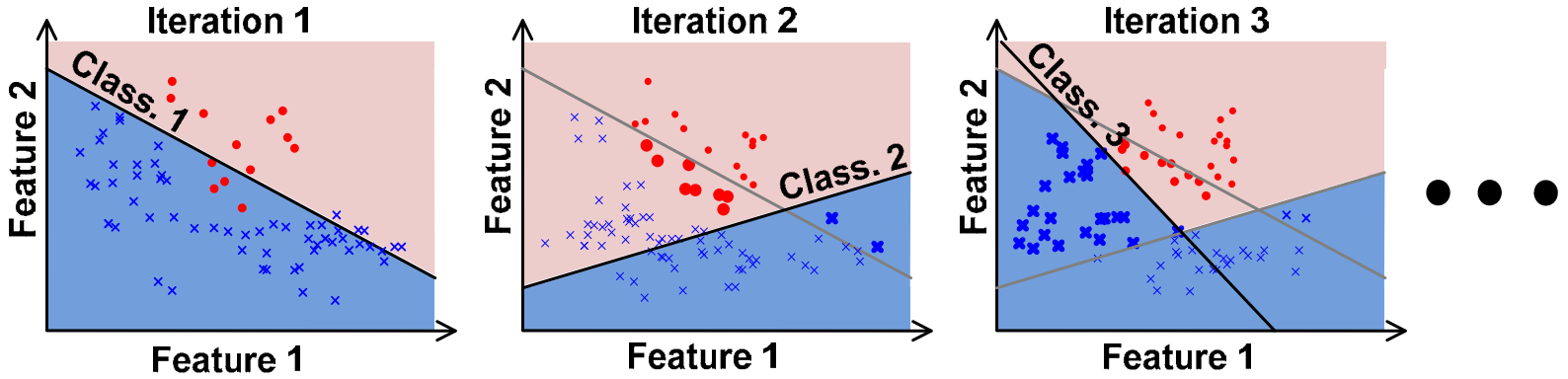



# Low-memory, embedded trainer

**AdaBoost Training – Same training data used for each iteration**




**Proposed Training – Different training data used for each iteration**



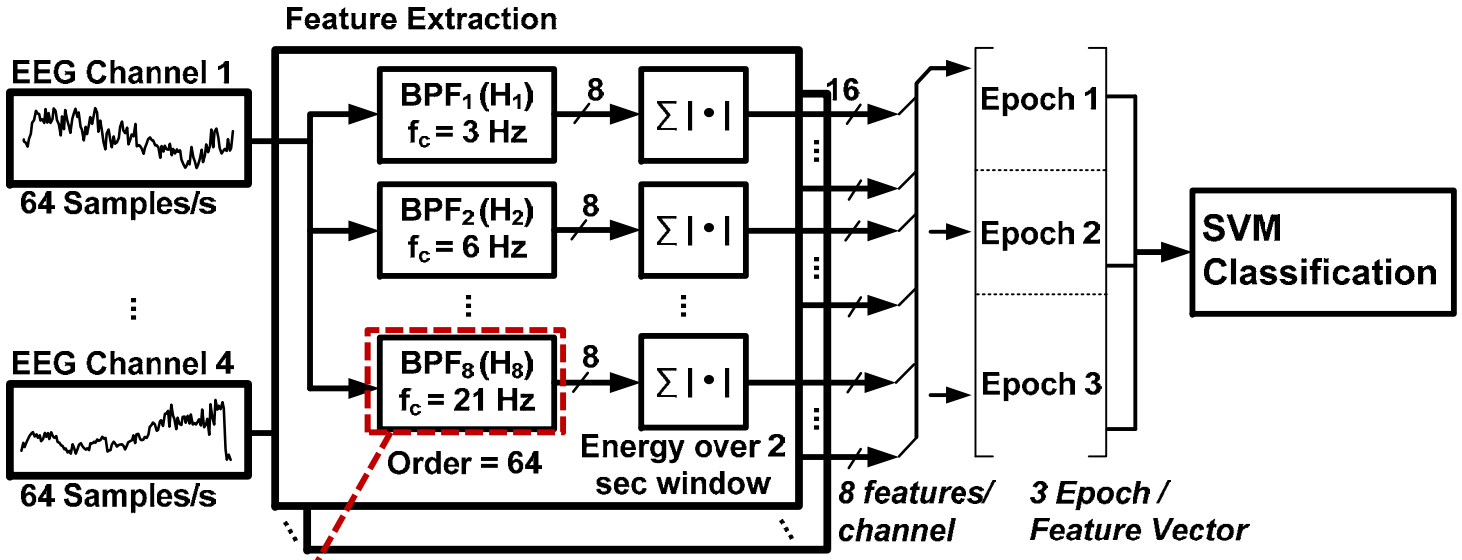

**Ex.: Seizure detector trained on OpenMSP core:**

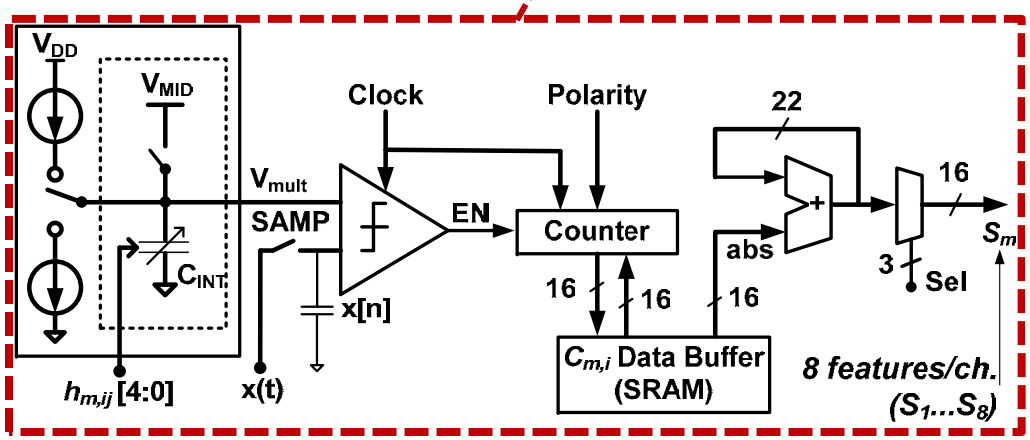
**65x memory reduction (to 7kB); 10x energy reduction (to 5M clock cycles)**

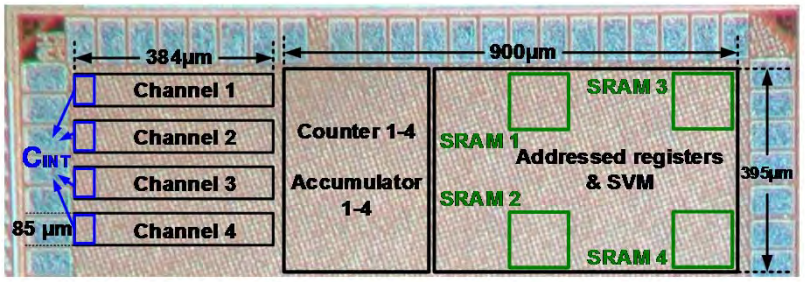
# Architecture design: instrumentation & ADC

**Motivation: E/ADC≈200pJ, E/MAC≈1pJ**

## EEG-based Seizure Detection

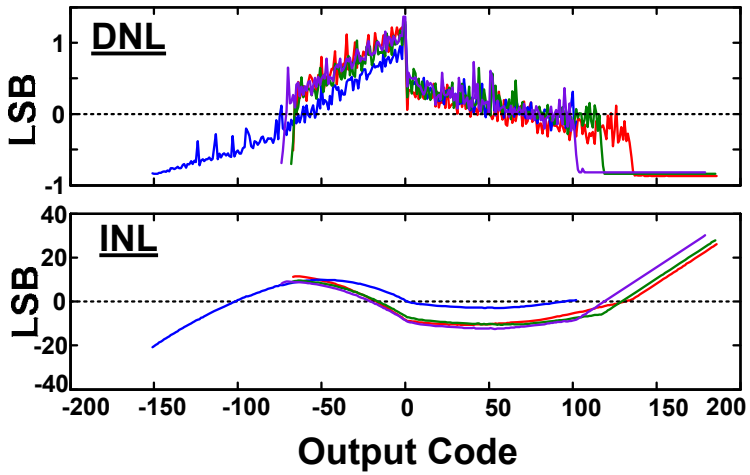


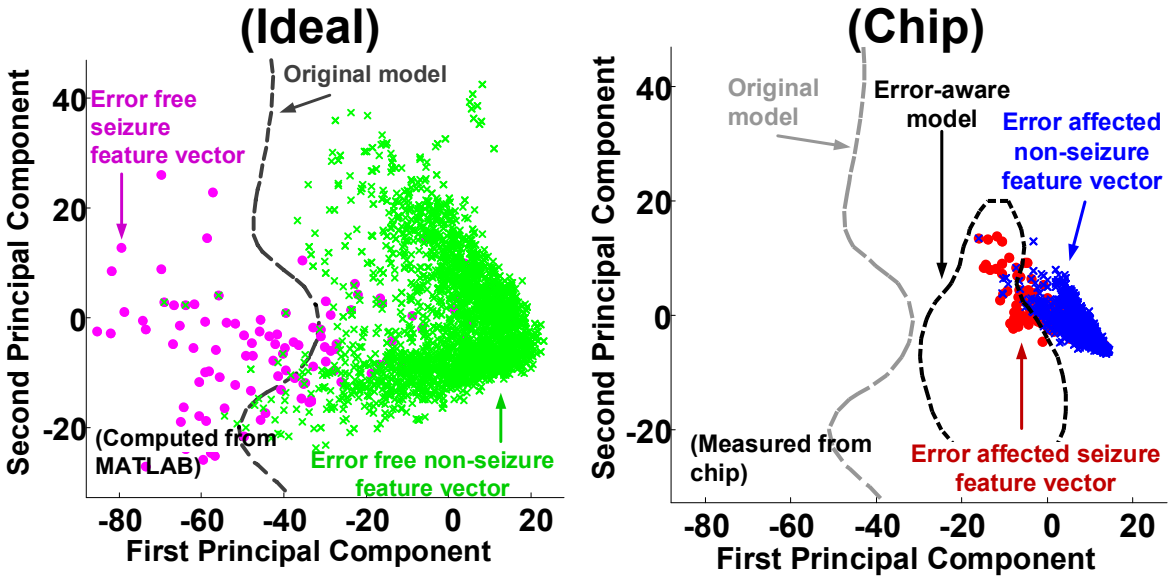

**Mixed-signal Seizure Detector (32nm CMOS SOI)**

# In-ADC feature extraction

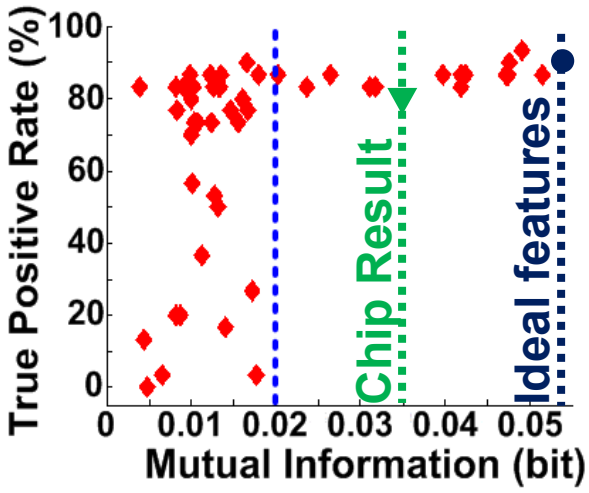



| | Ideal | Chip: baseline model | Chip: error-aware model |
|---|---|---|---|
| Sensitivity | 5/5 | 5/5 | 5/5 |
| Latency | 2.0 sec. | 3.6 sec. | 3.4 sec. |
| False alarms | 8 | 443 | 4 |

## Feature Distributions



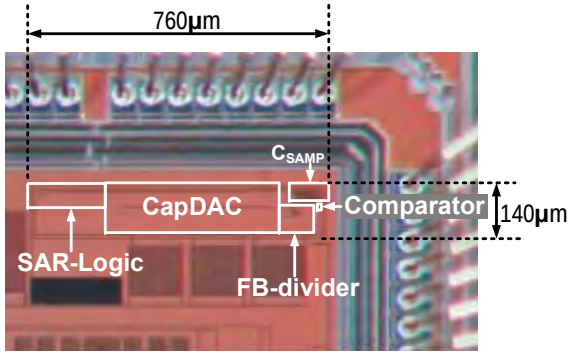

## Mutual Information




[J. Zhang, CICC'15]

# Related work

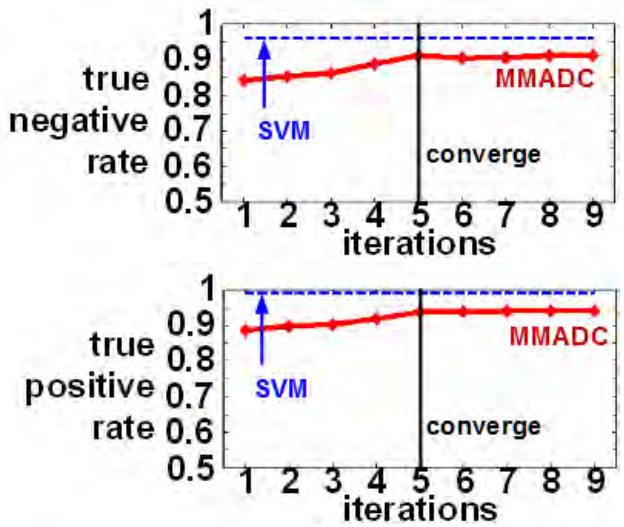
## Matrix-multiplying Successive Approximation ADC:




| Applications | Conventional System | Proposed System | Savings |
|---|---|---|---|
| 1. ECG arrhythmia detector (N=256, J=256, S=170, K=5) | Mults: 109,056 Adds: 108,630 | Mults: 1,280 Adds: 1,279 | Mults: 85× Adds: 85× |
| 2. Image-pixel gender detector (N=19200, J=200, S=180, K=1) | Mults: 3,876,000 Adds: 3,859,620 | Mults: 19,200 Adds: 19,199 | Mults: 202× Adds: 201× |

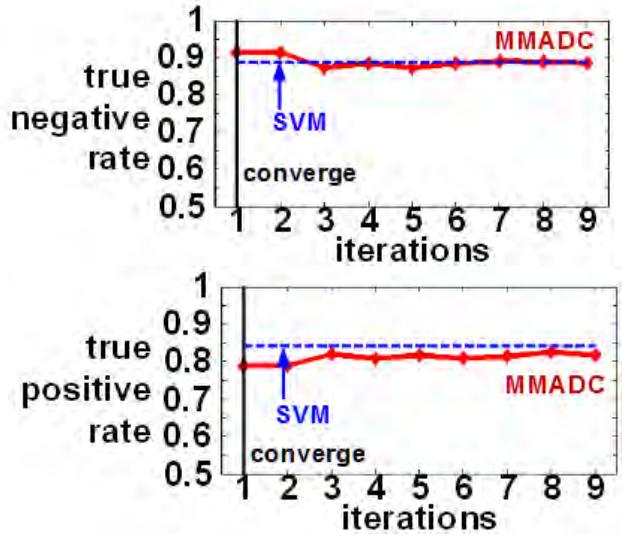
[J. Zhang, ISSCC'15]

### Demo. 1: ECG-based Arrhythmia Detector




9× energy reduction

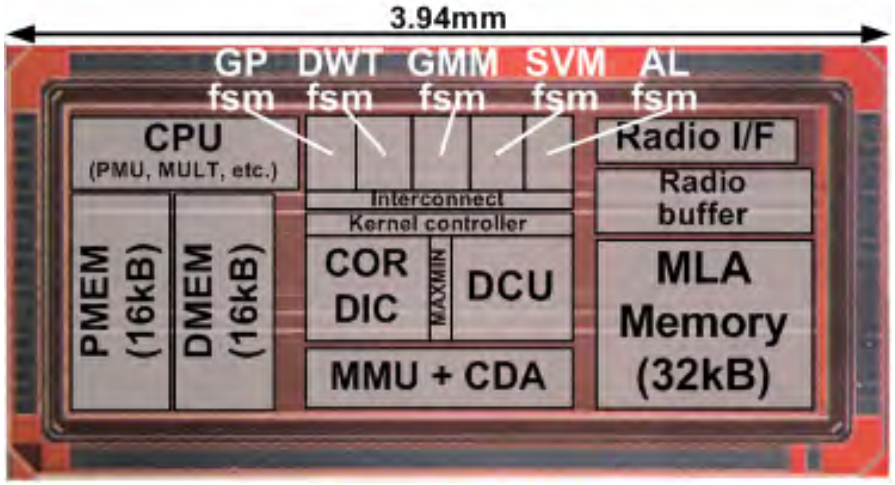
### Demo. 2: Image-pixel-based gender detector




23× energy reduction

# Architecture design: memory accessing

**Motivation:** E/SRAM≈20pJ, E/MAC≈1pJ




- DCU energy: ~10pJ/clock
- SRAM energy: 37pJ/clock
- CDA energy: 8.4pJ/clock

[K. H. Lee, VLSI Symp'13]

**Compression/Decompression Accelerator (CDA)**

+) Simple algorithm (ADPCM) with low energy overhead
→ 8.4pJ per comp./decomp. (memory access energy: 36.4pJ)
→ 4x compression guaranteed

-) Lossy; errors in low-order bits
→ Low impact thanks to the resilience of machine-learning frameworks against noise in low-order bits

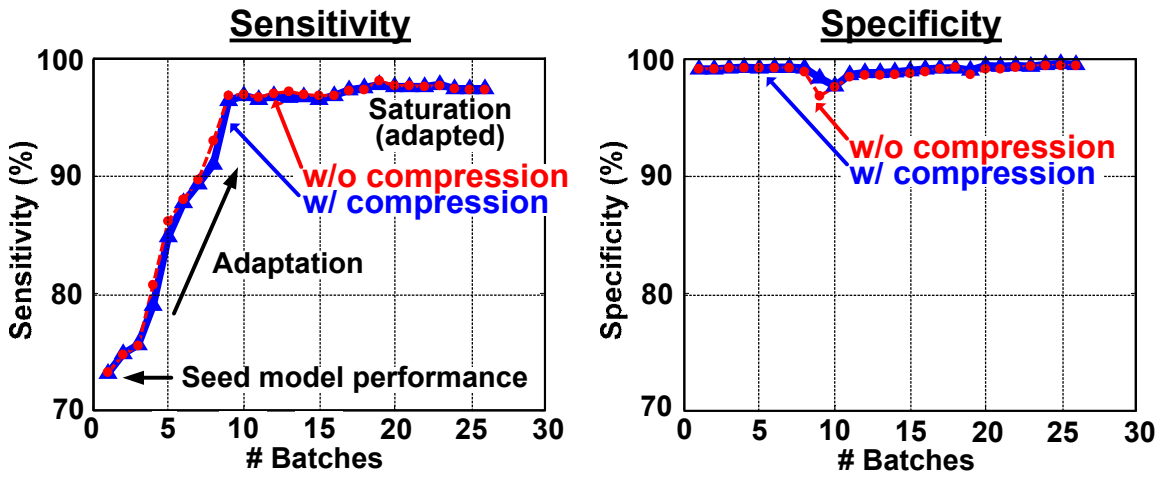
# Compression/decompression SRAM interface

## Ex. 1: EEG-based seizure detection

| | w/o compression | | w/ compression | |
|---|---|---|---|---|
| | True Pos. | True Neg. | True Pos. | True Neg. |
| Patient #1 | 96.1% | 98.1% | 94.1% | 98.9% |
| Patient #2 | 93.8% | 99.7% | 93.8% | 99.9% |
| Patient #3 | 91.7% | 98.7% | 90.4% | 99.4% |

## Ex. 2: Patient-adaptive arrhythmia detection

| Memory requirement | |
|---|---|
| w/o comp. | w/ comp. |
| 70.3kB | 17.6kB |

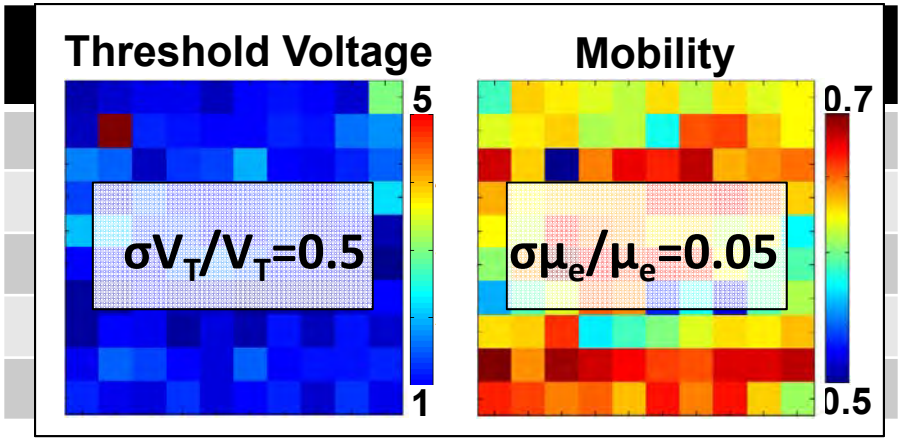


# Architecture design: sensing (looking ahead)

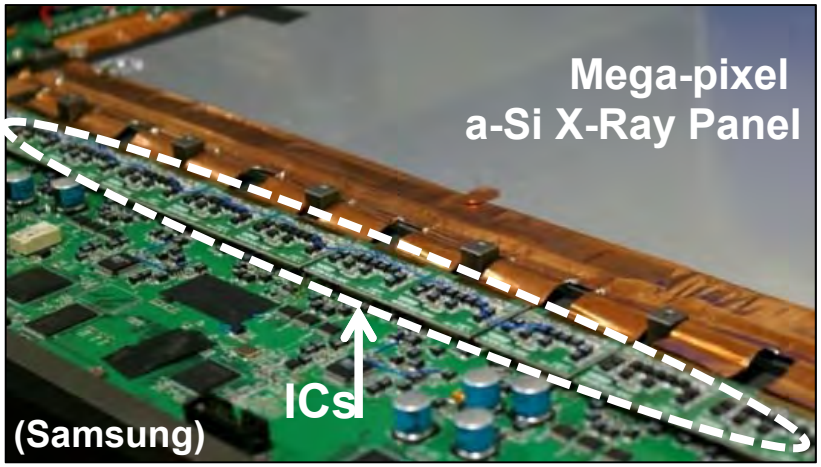
## Large-Area Electronics (LAE): a technology for *SENSING*

### Thin-film Sensors


### Thin-film Transistors

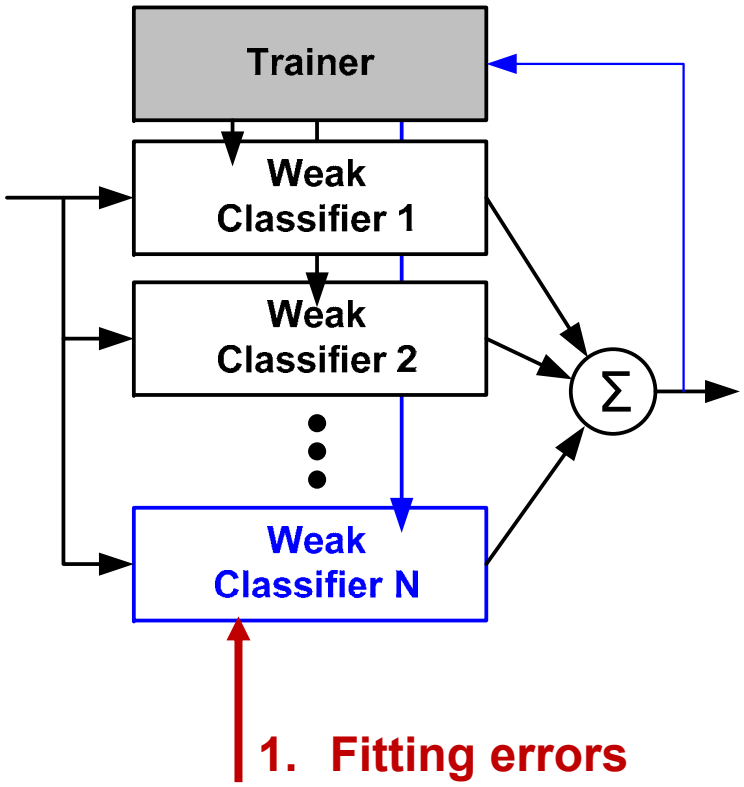



## Hybrid Systems (LAE/Si-CMOS)



# Embedded thin-film classifiers

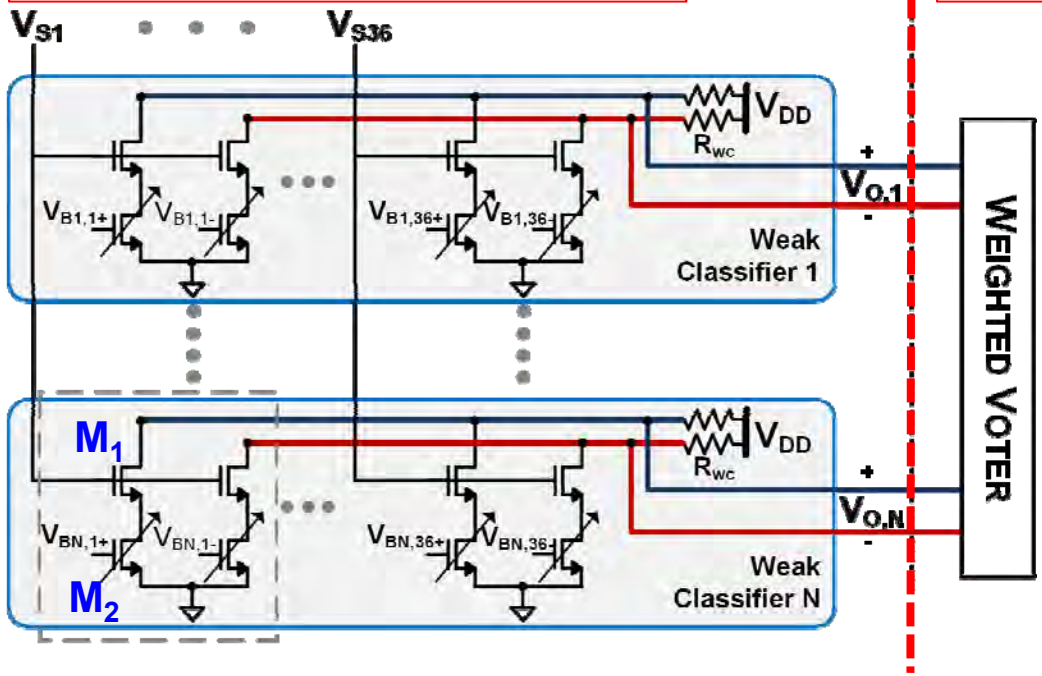
**Error-Adaptive Classifier Boosting**

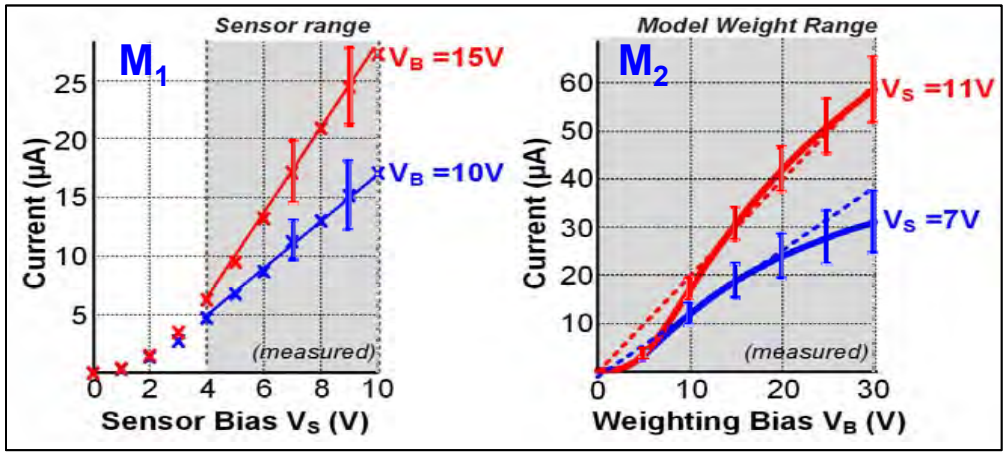



1. Fitting errors
2. Non-idealities

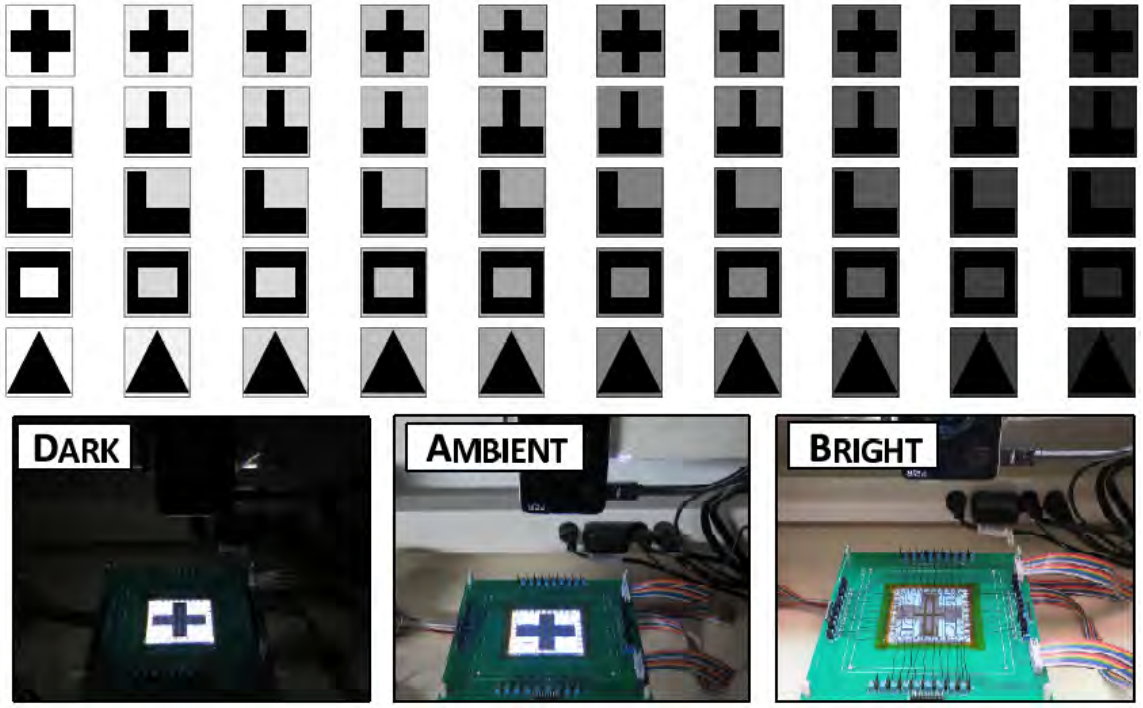
Large-area image-sensing pixels

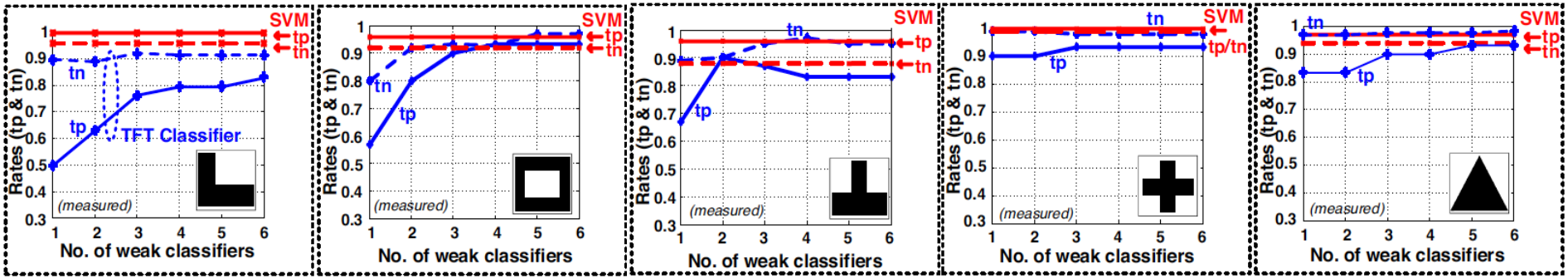
CMOS



# Image-sensing/-detection system



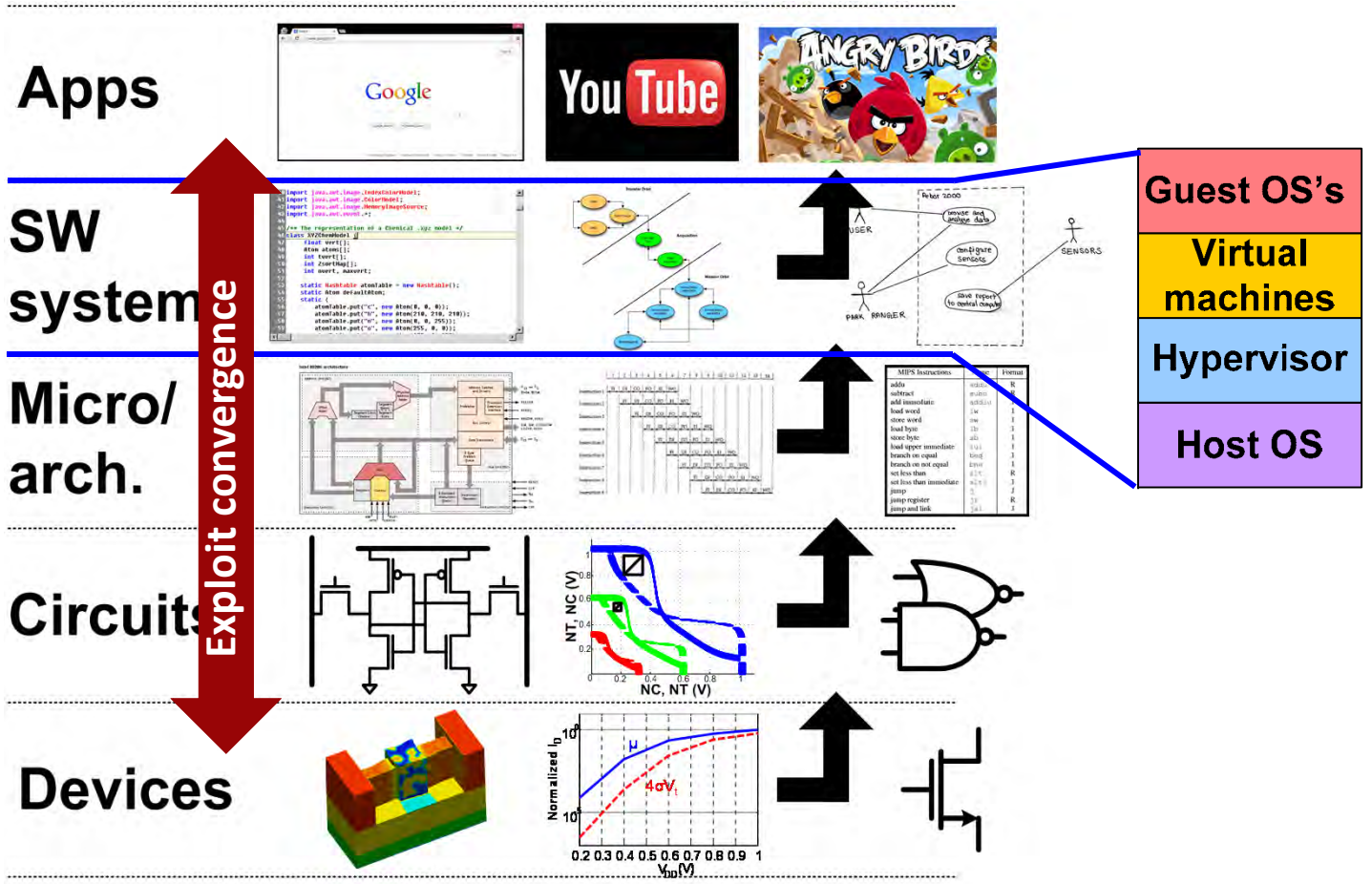






[W. Rieutort-Louis, ISSCC'15]

# But, we're at WAX…

**WHOLE POINT:**
**abstract away the dirty reality**



# Summary and conclusions

**There is a (powerful?) convergence between attributes on APPLICATION level and TECHNOLOGICAL level**

**There are many mechanisms in systems by which we have a direct tradeoff between 'usefulness of results' and 'resources consumed'**
**➔ *BUT, direct tradeoff usually does not have much leverage***

**Top-down architectural choices can often be enabled by approximate computing**
**(b/c statistical behaviors are so prominent/pervasive in system physics)**

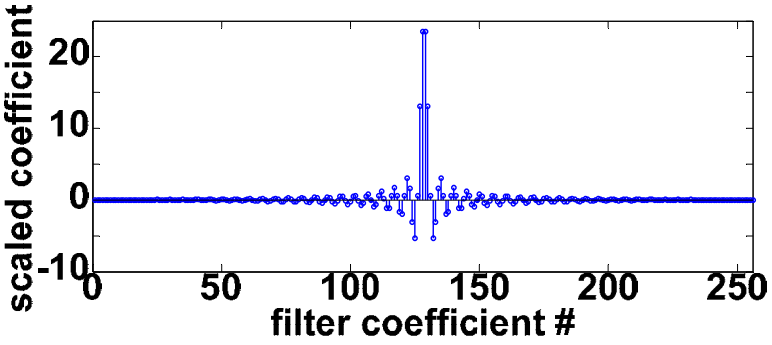
**But, now application designers must consider statistics of applications AND systems**

Note: All work is done by students (K. H. Lee, J. Liu, W. Rieutort-Louis, Z. Wang, J. Zhang)
Acknowledgements: FCRP (GSRC), STARnet (SONIC, C-FAR), NSF, SRC, AFOSR, MOSIS.

# EXTRAS

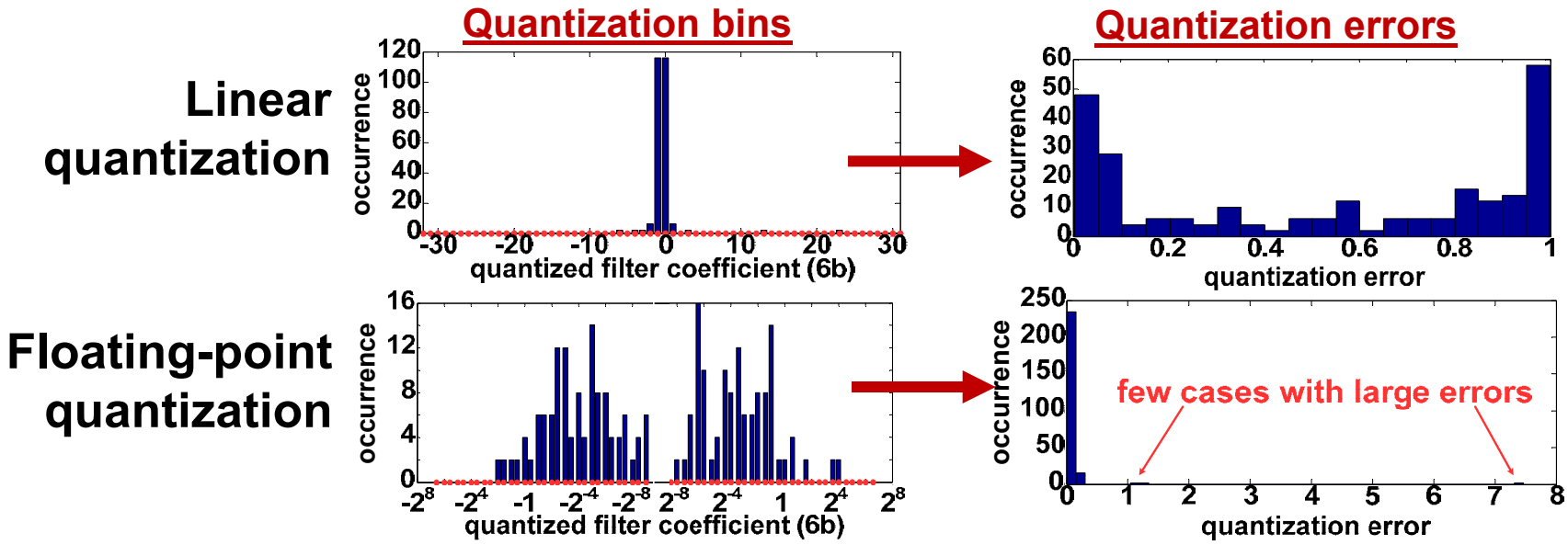
# Playing with bit precision

## E.g.: optimizing FIR filter SQNR

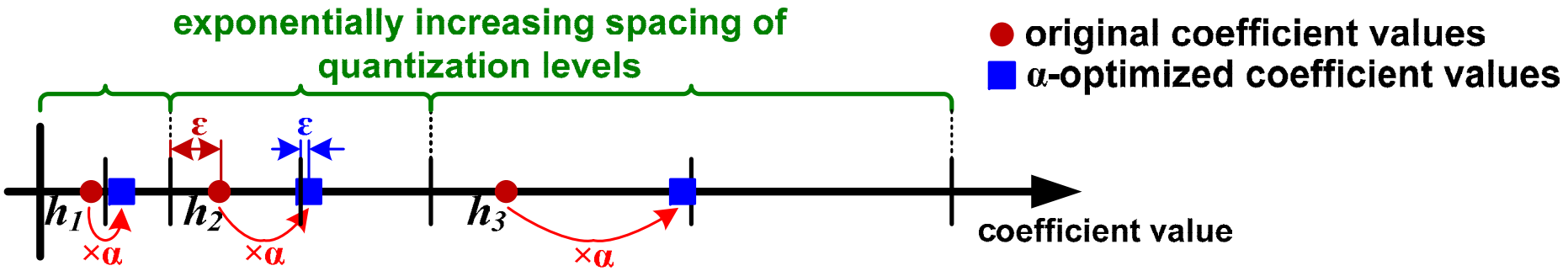



- **Represent filter coefficients as**

$$h_l = l_l \times 2^{s_l} \times (1 + m_l)$$



# Statistical optimization (I)

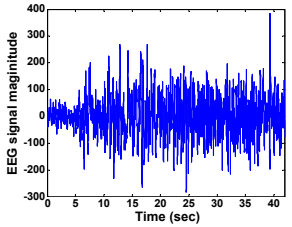



**Objective: Find optimal $\alpha$, to minimize error $\varepsilon_{\hat{y}} = \frac{|\sum_t (\alpha h_t) x_t - \sum_t (\widehat{\alpha h_t}) x_t|}{\alpha}$.**

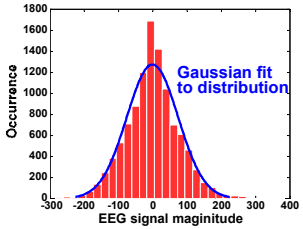
**Approach:**

1. Assume $x_i$ drawn from random variable $X_i$ (having some statistical distribution).

E.g., segment of EEG ($x_i$)



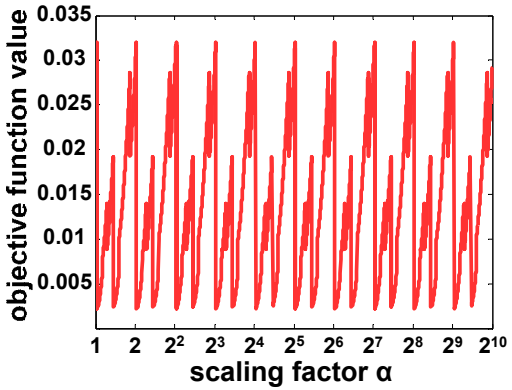

Estimated statistics of $X_i$

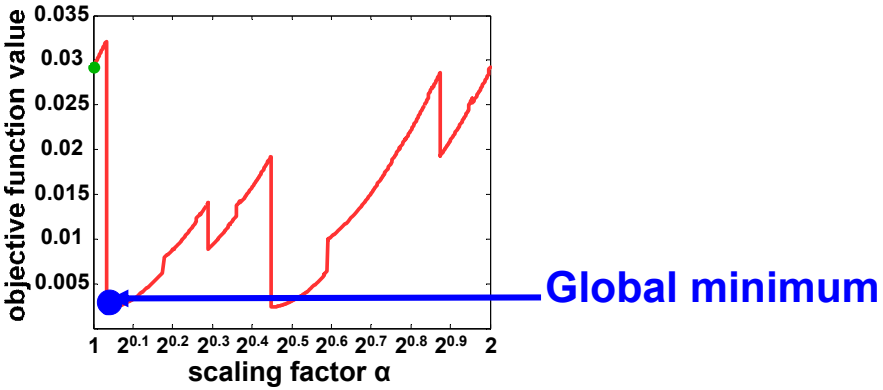



2. Compute distribution of output error $E_{\hat{Y}}$, and define cost function for minimization.
3. Minimize cost function to find optimal $\alpha$.

# Statistical optimization (II)

Cost function is found to be 'periodic'




## Ex.: FIR filtering EEG feature extraction

[Z. Wang, *ICASSP* 2015]