Learning on Silicon: Overview

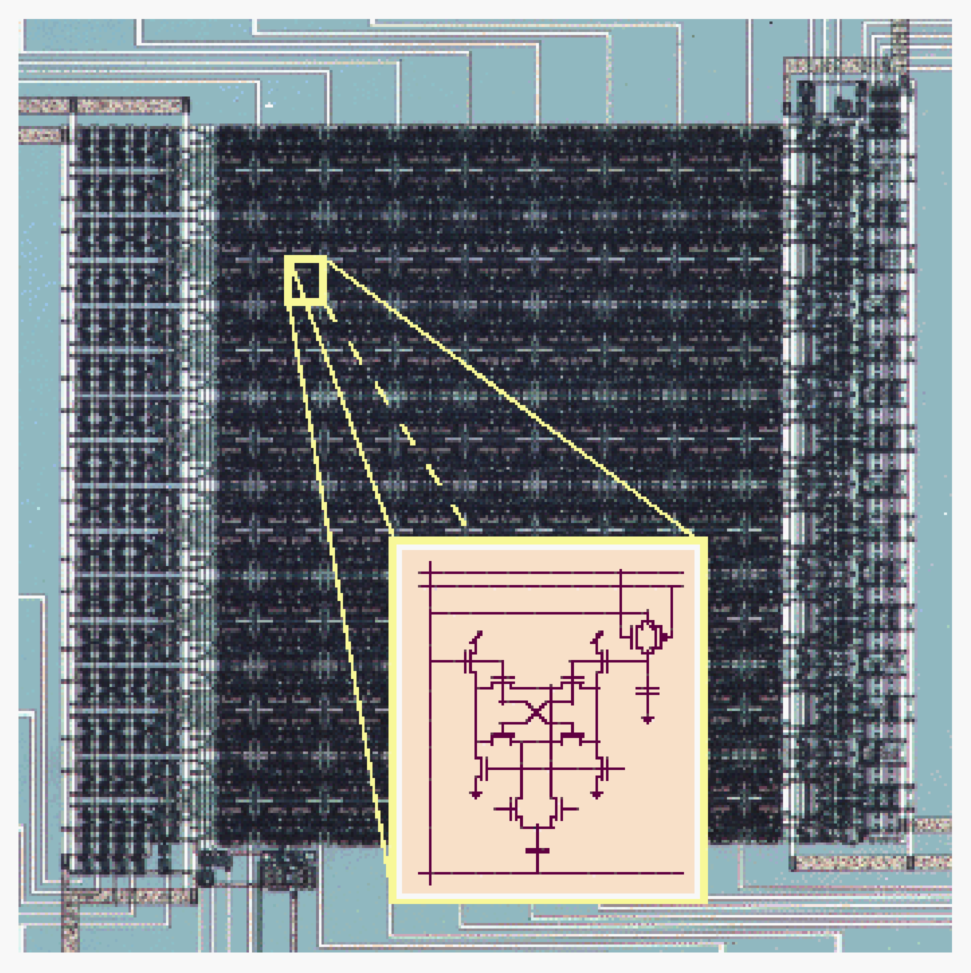
Gert Cauwenberghs
Johns Hopkins University
gert@jhu.edu

520.776 Learning on Silicon
http://bach.ece.jhu.edu/gert/courses/776

Learning on Silicon: Overview

- **Adaptive Microsystems**
  - Mixed-signal parallel VLSI
  - Kernel machines
- **Learning Architecture**
  - Adaptation, learning and generalization
  - Outer-product incremental learning
- **Technology**
  - Memory and adaptation
    - *Dynamic analog memory*
    - *Floating gate memory*
  - Technology directions
    - *Silicon on Sapphire*
- **System Examples**

Massively Parallel Distributed VLSI Computation



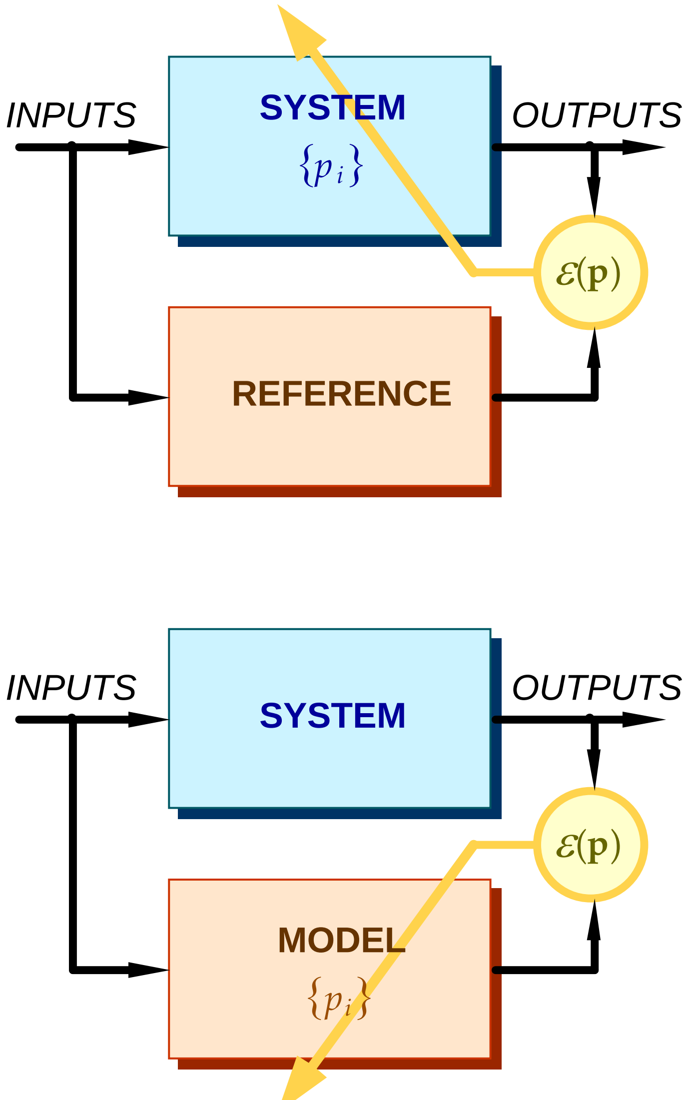
Example: VLSI Analog-to-digital vector quantizer (Cauwenberghs and Pedroni, 1997)

- **Neuromorphic**
  - distributed representation
  - local memory and adaptation
  - sensory interface
  - physical computation
  - internally analog, externally digital
- **Scalable**

  throughput scales linearly with silicon area
- **Ultra Low-Power**

  factor 100 to 10,000 less energy than CPU or DSP

Learning on Silicon



Adaptation:

- necessary for robust performance under variable and unpredictable conditions
- also compensates for imprecisions in the computation
- avoids ad-hoc programming, tuning, and manual parameter adjustment

Learning:

- generalization of output to previously unknown, although similar, stimuli
- system identification to extract relevant environmental parameters

Adaptive Elements

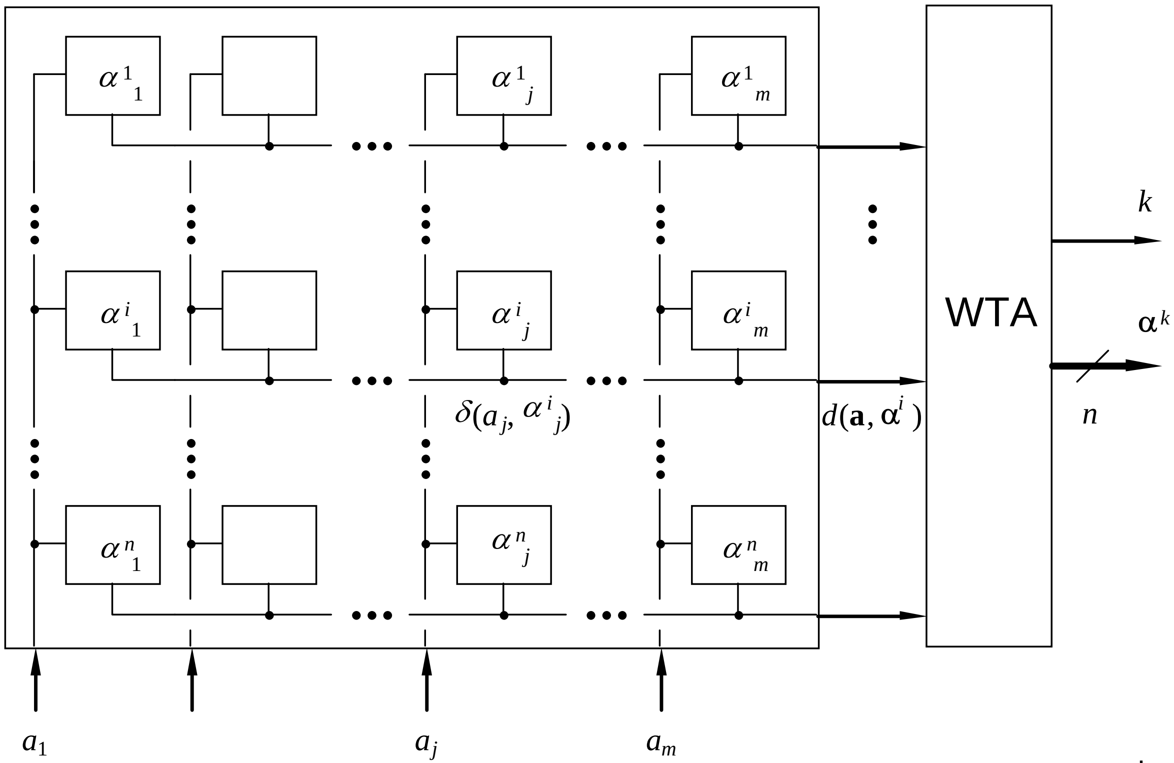
Adaptation:*

Autozeroing (high-pass filtering)	*outputs*
Offset Correction	*outputs*
e.g. Image Non-Uniformity Correction	
Equalization /Deconvolution	*inputs, outputs*
e.g. Source Separation; Adaptive Beamforming	

Learning:

Unsupervised Learning	*inputs, outputs*
e.g. Adaptive Resonance; LVQ; Kohonen	
Supervised Learning	*inputs, outputs, targets*
e.g. Least Mean Squares; Backprop	
Reinforcement Learning	*reward/punishment*

Example: Learning Vector Quantization (LVQ)

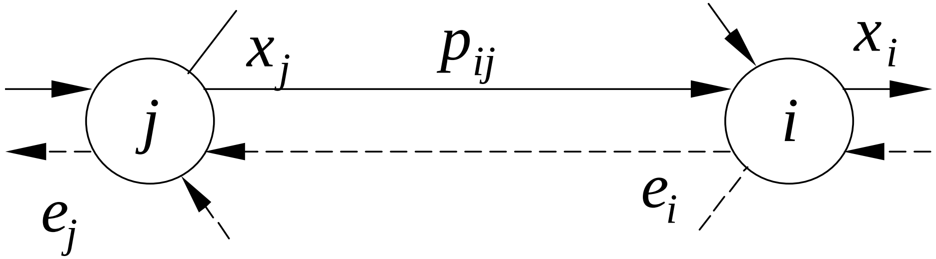


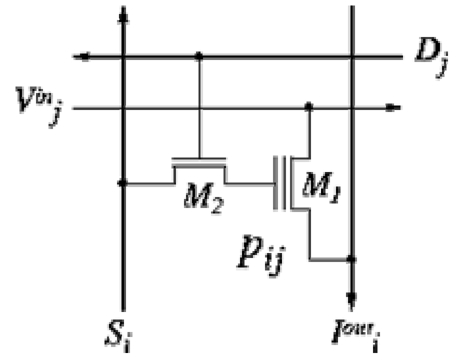
Distance Calculation: $d(\mathbf{a}, \alpha^i) = \sum_j \delta(a_j, \alpha_j^i) = \sum_j \left| a_j - \alpha_j^i \right|^\nu$

Winner-Take-All Selection: $k = \mathrm{argmin}_i \, d(\mathbf{a}, \alpha^i)$

Training: $\alpha_j^k \leftarrow (1 - \lambda)\, \alpha_j^k + \lambda\, a_j$

Incremental Outer-Product Learning in Neural Nets





Multi-Layer Perceptron:

$$x_i = f\left(\sum_j p_{ij}\, x_j\right)$$

Outer-Product Learning Update:

$$\Delta p_{ij} = \eta \; x_j \cdot e_i$$

- Hebbian *(Hebb, 1949)*: $e_i = x_i$
- LMS Rule *(Widrow-Hoff, 1960)*: $e_i = f'_i \cdot \left(x_i^{\text{target}} - x_i\right)$
- Backpropagation *(Werbos, Rumelhart, LeCun)*: $e_j = f'_j \cdot \sum_i p_{ij}\, e_i$

Technology

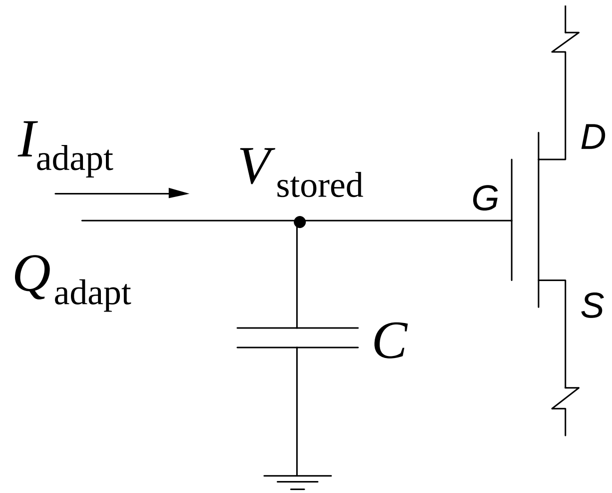
Incremental Adaptation:

– Continuous-Time:

$$C \frac{\mathrm{d}}{\mathrm{d}t} V_{\mathrm{stored}} = I_{\mathrm{adapt}}$$

– Discrete-Time:

$$C \, \Delta V_{\mathrm{stored}} = Q_{\mathrm{adapt}}$$



Storage:

– Volatile capacitive storage (incremental refresh)

– Non-volatile storage (floating gate)

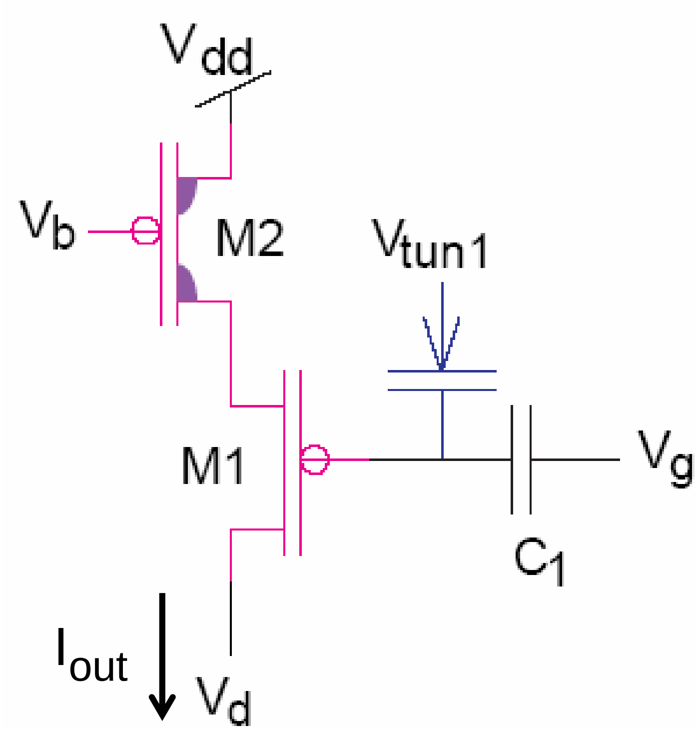
Precision:

– Only polarity of the increments is critical (not amplitude).

– Adaptation compensates for inaccuracies in the analog implementation of the system.

Floating-Gate Non-Volatile Memory and Adaptation

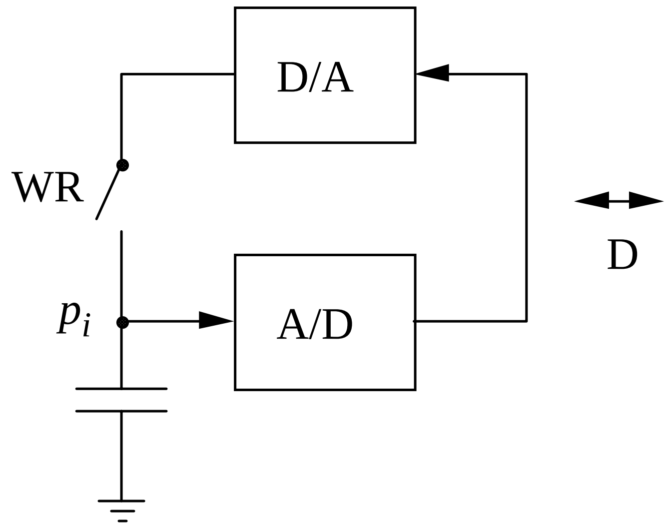
Paul Hasler, Chris Diorio, Carver Mead, …



- **Hot electron injection**
  - 'Hot' electrons injected from drain onto floating gate of M1.
  - Injection current is proportional to drain current and exponential in floating-gate to drain voltage (~5V).
- **Tunneling**
  - Electrons tunnel through thin gate oxide from floating gate onto high-voltage (~30V) n-well.
  - Tunneling voltage decreases with decreasing gate oxide thickness.
- **Source degeneration**
  - Short-channel M2 improves stability of closed-loop adaptation (Vd open-circuit).
  - M2 is not required if adaptation is regulated (Vd driven).
- **Current scaling**
  - In subthreshold, Iout is exponential both in the floating gate charge, and in control voltage Vg.

Dynamic Analog Memory Using Quantization and Refresh

Autonomous Active Refresh Using A/D/A Quantization:



- Allows for an excursion margin around discrete quantization levels, provided the rate of refresh is sufficiently fast.
- Supports digital format for external access
- Trades analog depth for storage stability

Binary Quantization and Partial Incremental Refresh

Problems with Standard Refresh Schemes:

- Systematic offsets in the A/D/A loop
- Switch charge injection (clock feedthrough) during refresh
- Random errors in the A/D/A quantization

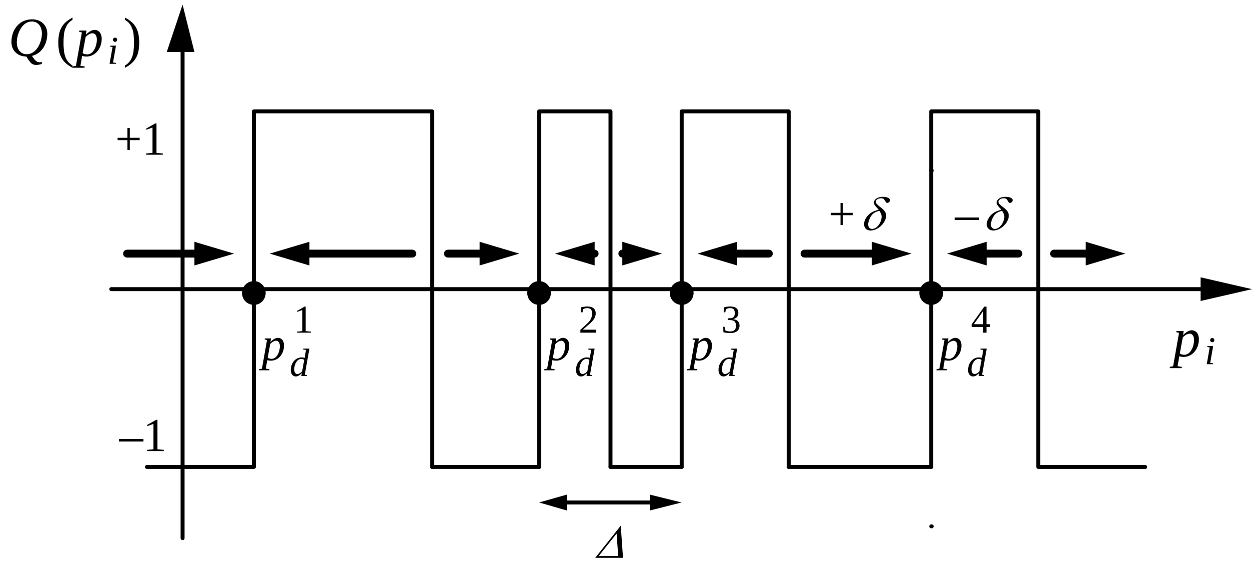
Binary Quantization:

- Avoids errors due to analog refresh
- Uses a charge pump with precisely controlled *polarity* of increments

Partial Incremental Refresh:

- Partial increments avoid catastrophic loss of information in the presence of random errors and noise in the quantization
- Robustness to noise and errors increases with smaller increment amplitudes

Binary Quantization and Partial Incremental Refresh

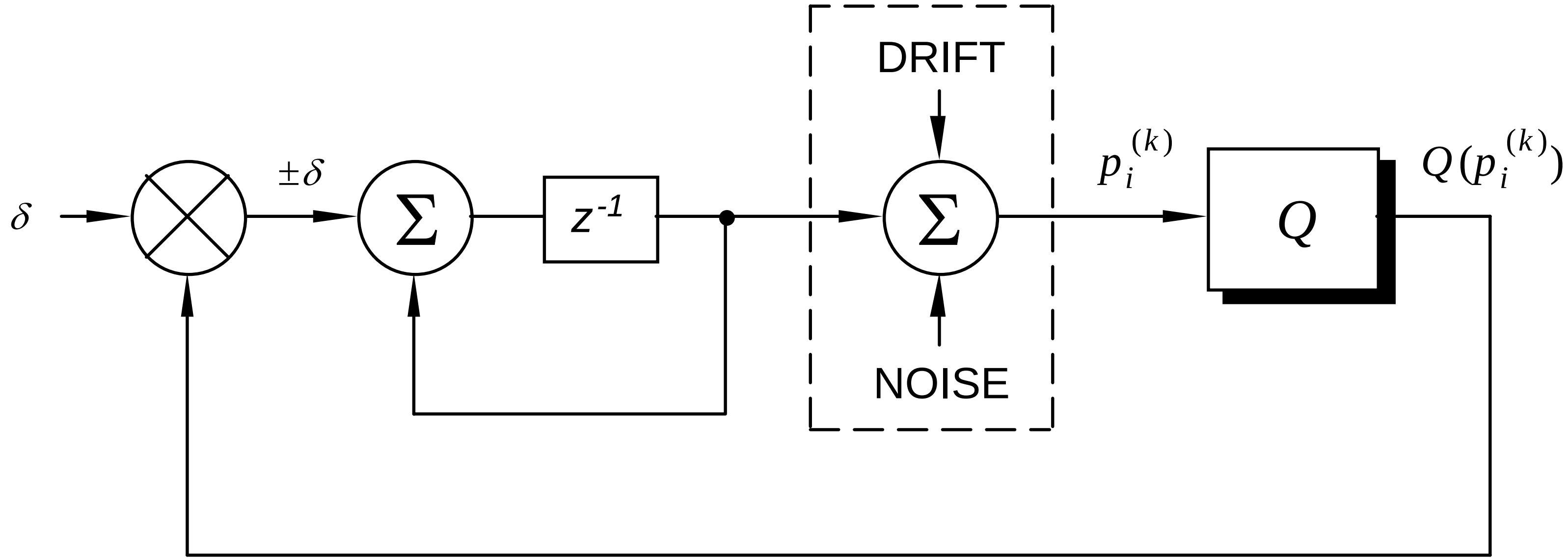


$$p_i^{(k+1)} = p_i^{(k)} - \delta\, Q(p_i^{(k)})$$

- Resolution Δ
- Increment size δ
- Worst-case drift rate ($|\mathrm{d}p/\mathrm{d}t|$) r
- Period of refresh cycle T

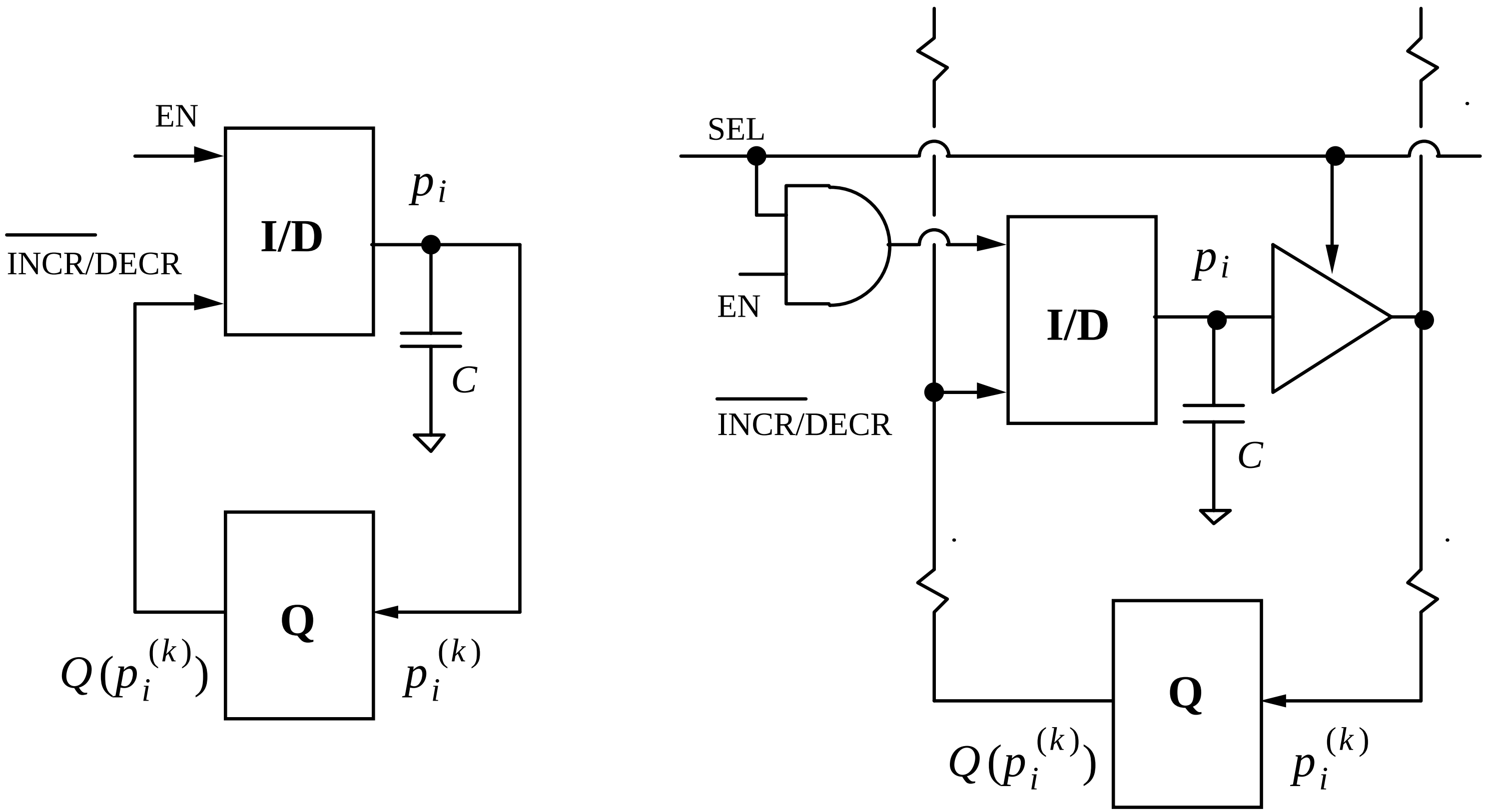
$$r\,T < \delta \ll \Delta$$

Functional Diagram of Partial Incremental Refresh



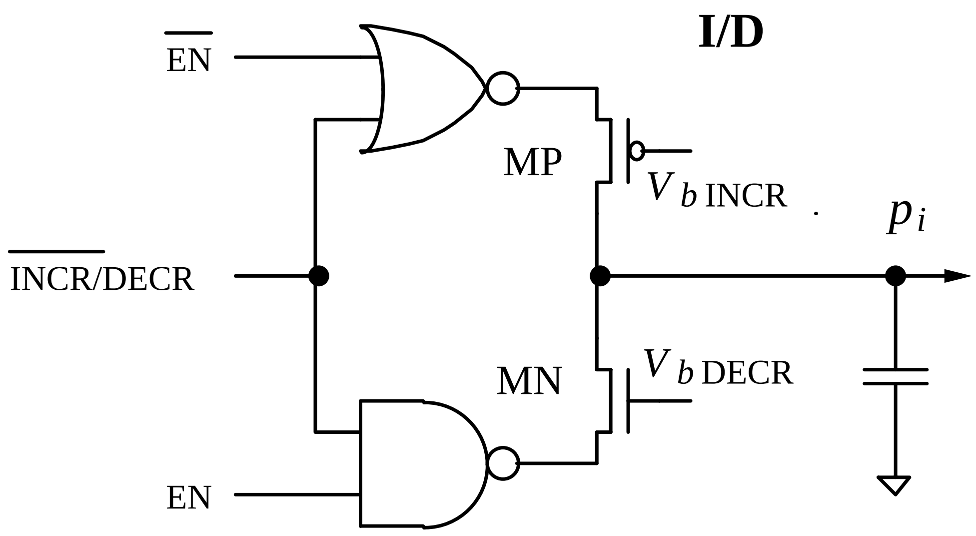
- *Similar in function and structure to the technique of delta-sigma modulation*
- *Supports efficient and robust analog VLSI implementation, using binary controlled charge pump*

Analog VLSI Implementation Architectures



- *An increment/decrement device* I/D *is provided for every memory cell, serving refresh increments locally.*
- *The binary quantizer* Q *is more elaborate to implement, and one instance can be time-multiplexed among several memory cells*

Charge Pump Implementation of the I/D Device



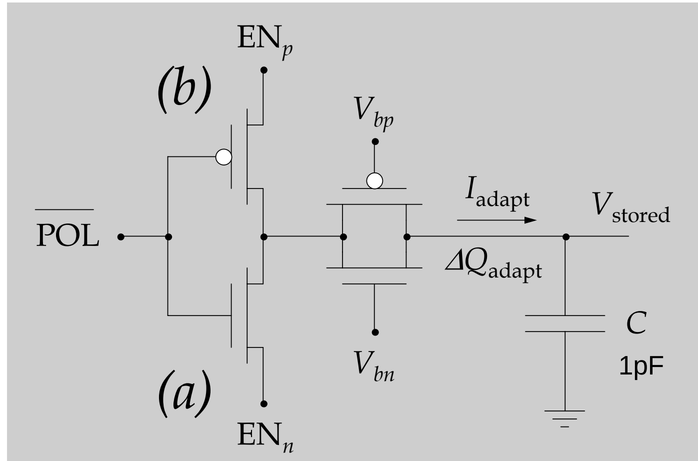
Binary controlled polarity of increment/decrement

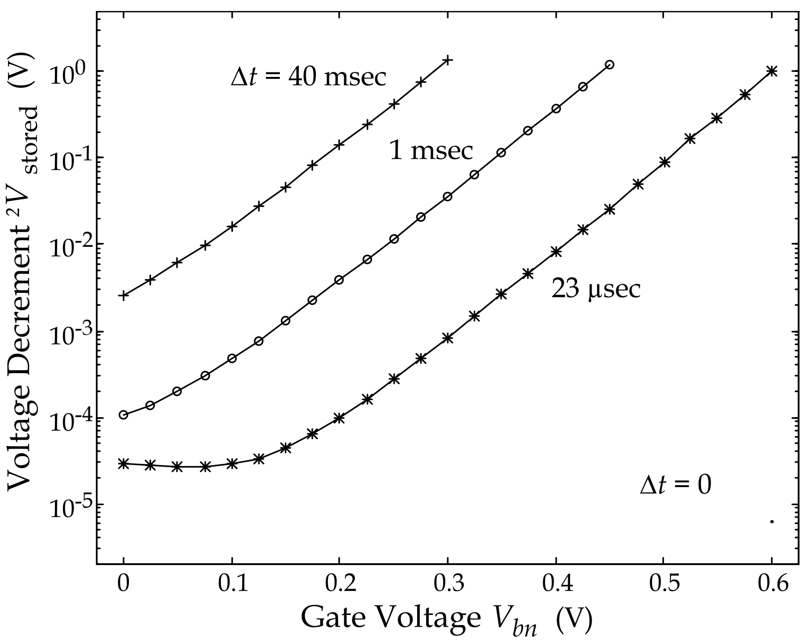
- INCR/DECR controls polarity of current

Accurate amplitude over wide dynamic range of increments

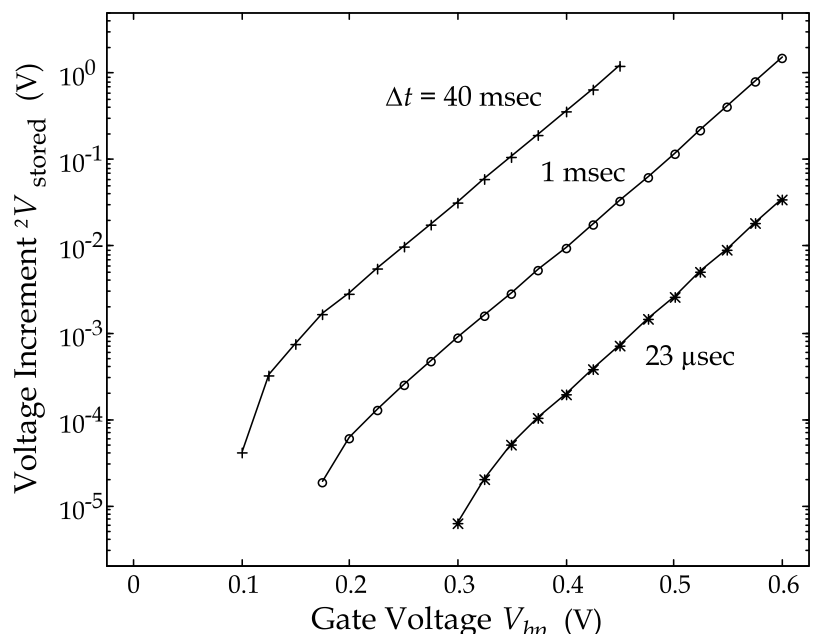
- EN controls duration of current
- $V_{b\,\mathrm{INCR}}$ and $V_{b\,\mathrm{DECR}}$ control amplitude of subthreshold current
- No clock feedthrough charge injection (gates at constant potentials)

Dynamic Memory and Incremental Adaptation



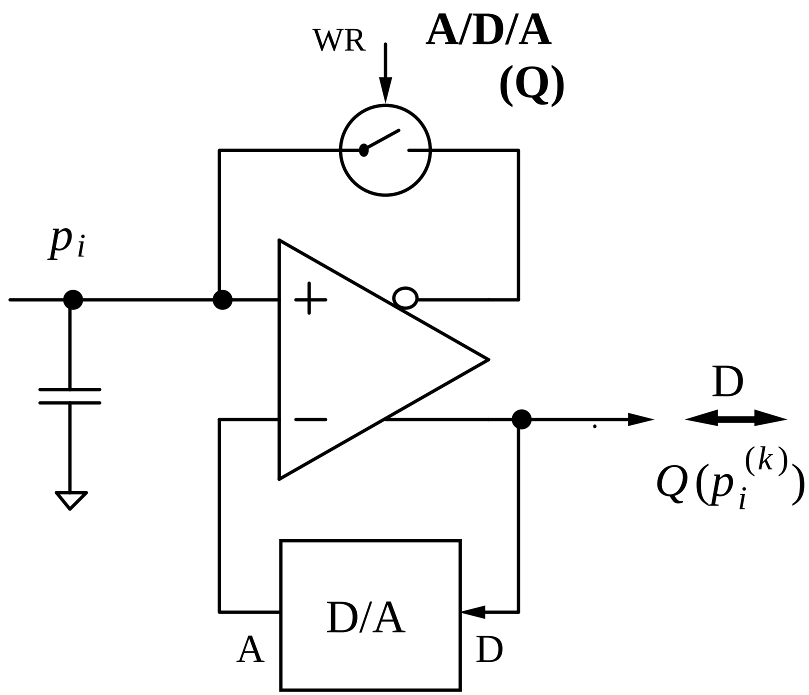


(a)



(b)

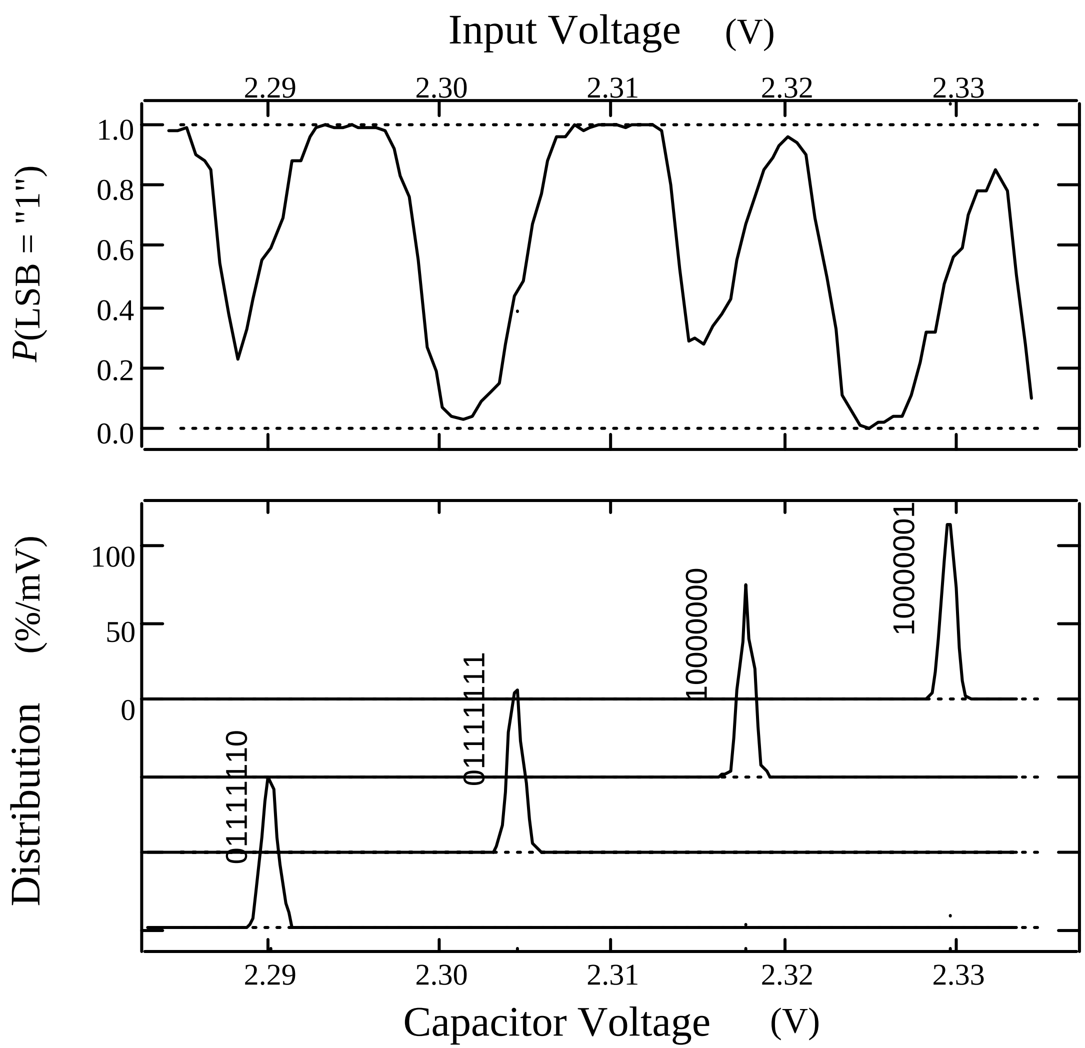
A/D/A Quantizer for Digital Write and Read Access



Integrated bit-serial (MSB-first) D/A and SA A/D converter:

- **Partial Refresh:** Q(.) from LSB of (n+1)-bit A/D conv.
- **Digital Read Access:** n-bit A/D conv.
- **Digital Write Access:** n-bit D/A ; WR ; Q(.) from COMP

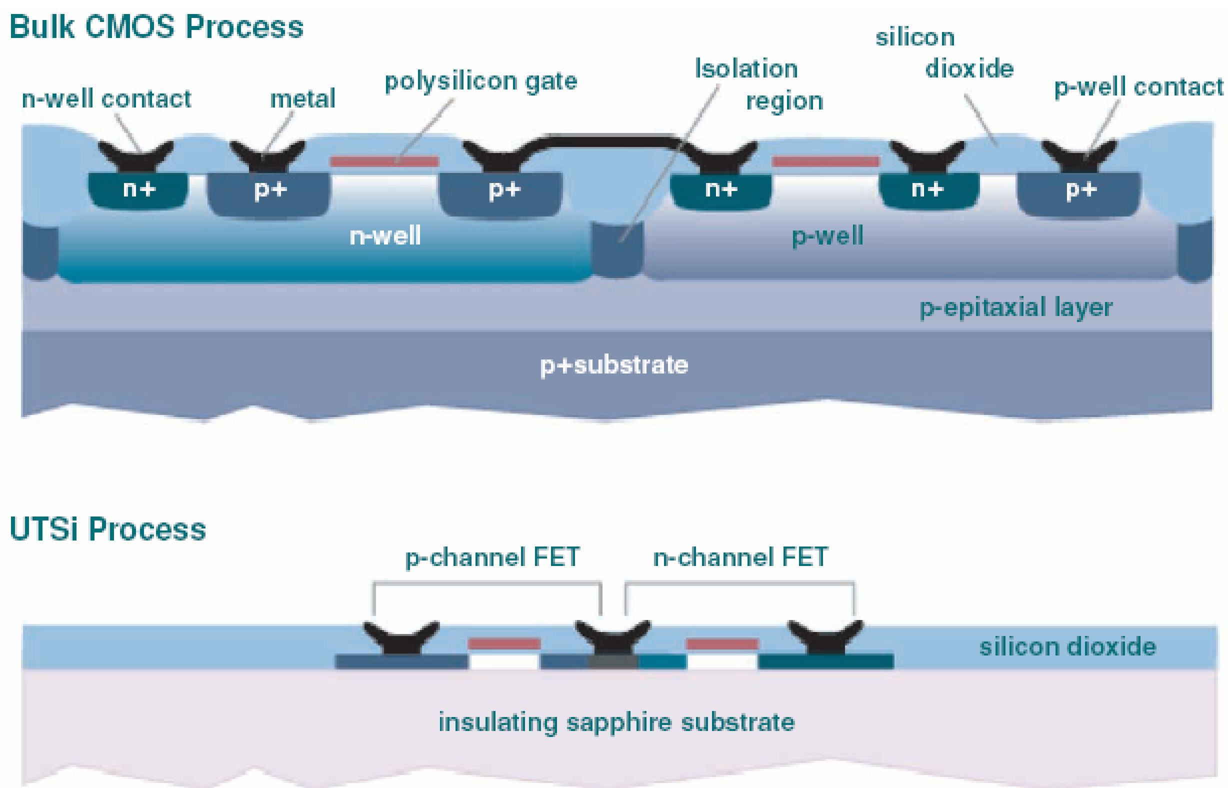
Dynamic Analog Memory Retention



- 10^9 cycles mean time between failure
- 8 bit effective resolution
- 20 µV increments/decrements
- 200 µm X 32 µm in 2 µm CMOS

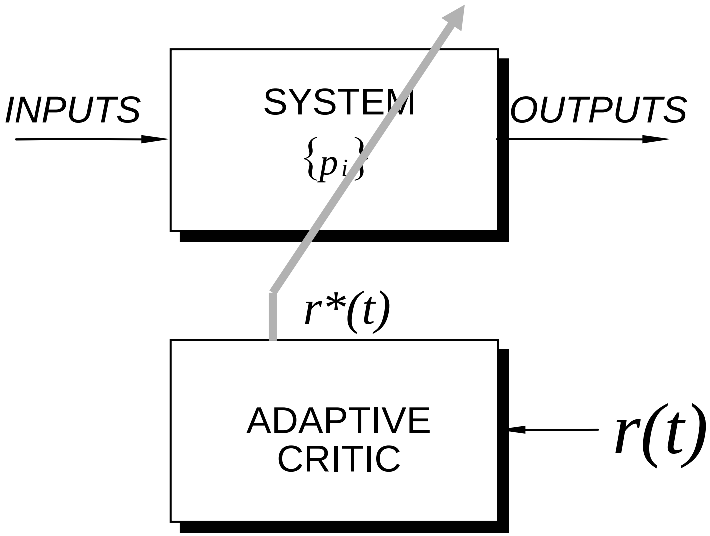
Silicon on Sapphire

Peregrine UTSi process



- Higher integration density
- Drastically reduced bulk leakage
  - *Improved analog memory retention*
- Transparent substrate
  - *Adaptive optics applications*

The Credit Assignment Problem
or How to Learn from Delayed Rewards

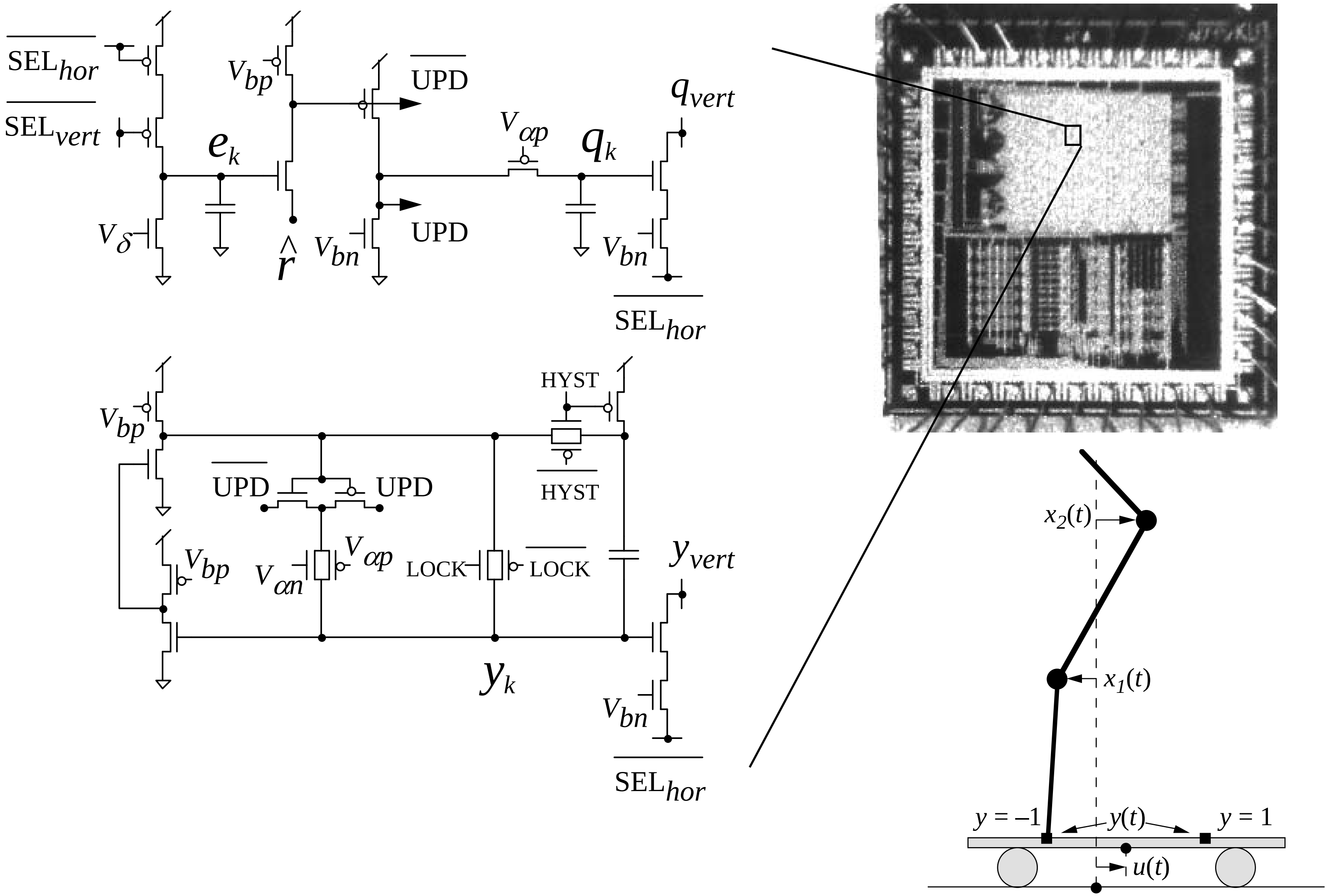


External, discontinuous reinforcement signal $r(t)$.

Adaptive Critics:

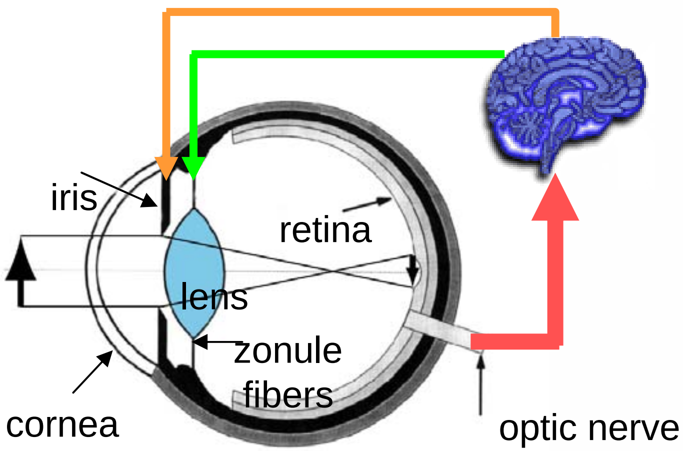
- Heuristic Dynamic Programming (Werbos, 1977)
- Reinforcement Learning (Sutton and Barto, 1983)
- TD(λ) (Sutton, 1988)
- Q-Learning (Watkins, 1989)

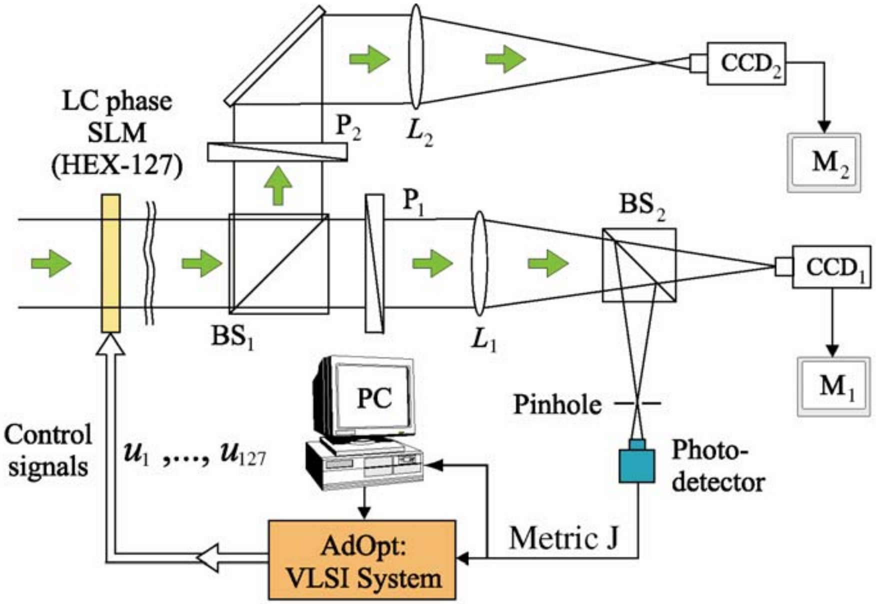
Reinforcement Learning Classifier for Binary Control

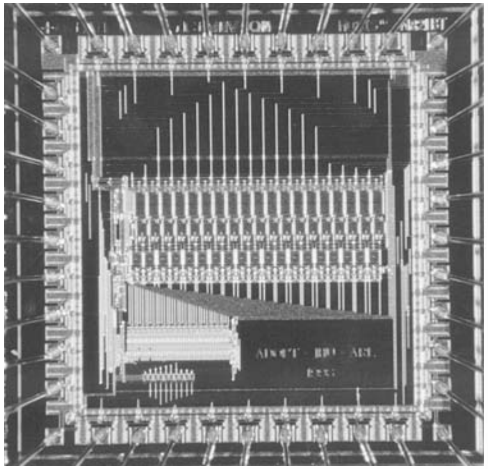


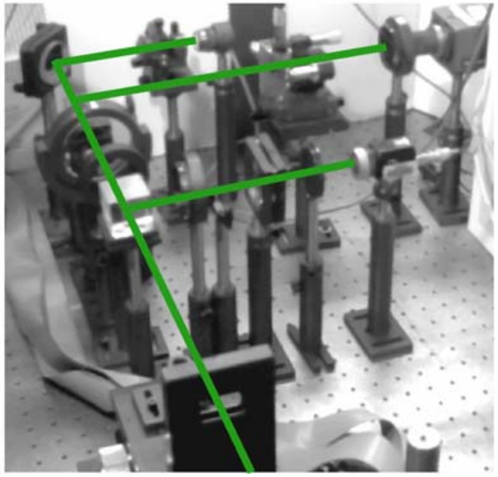
Adaptive Optical Wavefront Correction

with Marc Cohen, Tim Edwards and Mikhail Vorontsov



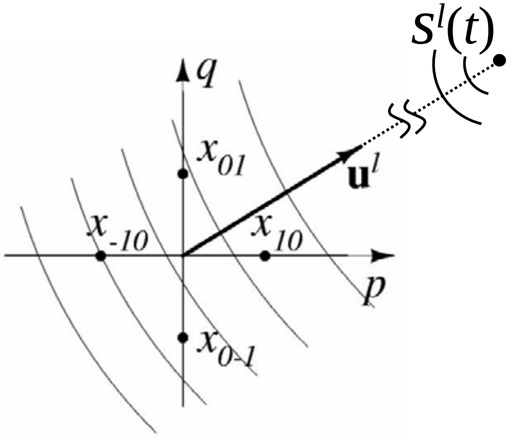


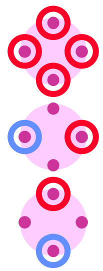




Gradient Flow Source Localization and Separation

with Milutin Stanacevic and George Zweig



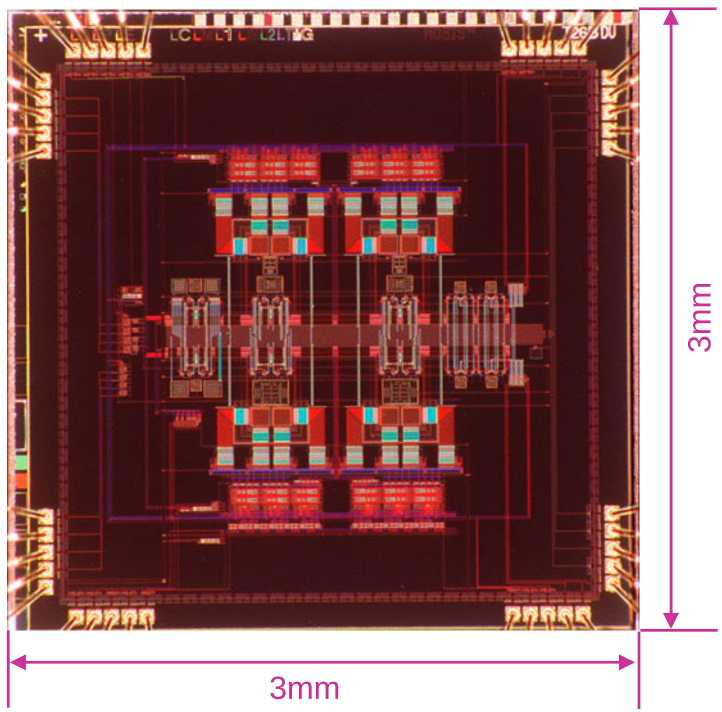


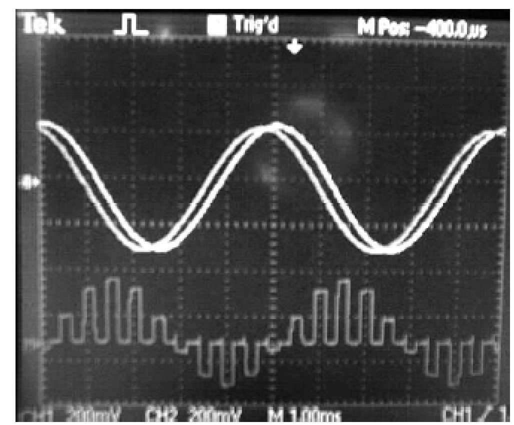
$$\frac{d}{dt}\frac{1}{4}\left(x_{-1,0}+x_{1,0}+x_{0,-1}+x_{0,1}\right) \approx \frac{\partial}{\partial t}x \approx \sum_\ell \dot{s}^\ell(t)$$

$$\frac{1}{2}\left(x_{1,0}-x_{-1,0}\right) \approx \frac{\partial}{\partial p}x \approx \sum_\ell \tau_1^\ell \dot{s}^\ell(t)$$

$$\frac{1}{2}\left(x_{0,1}-x_{0,-1}\right) \approx \frac{\partial}{\partial q}x \approx \sum_\ell \tau_2^\ell \dot{s}^\ell(t)$$

1cm





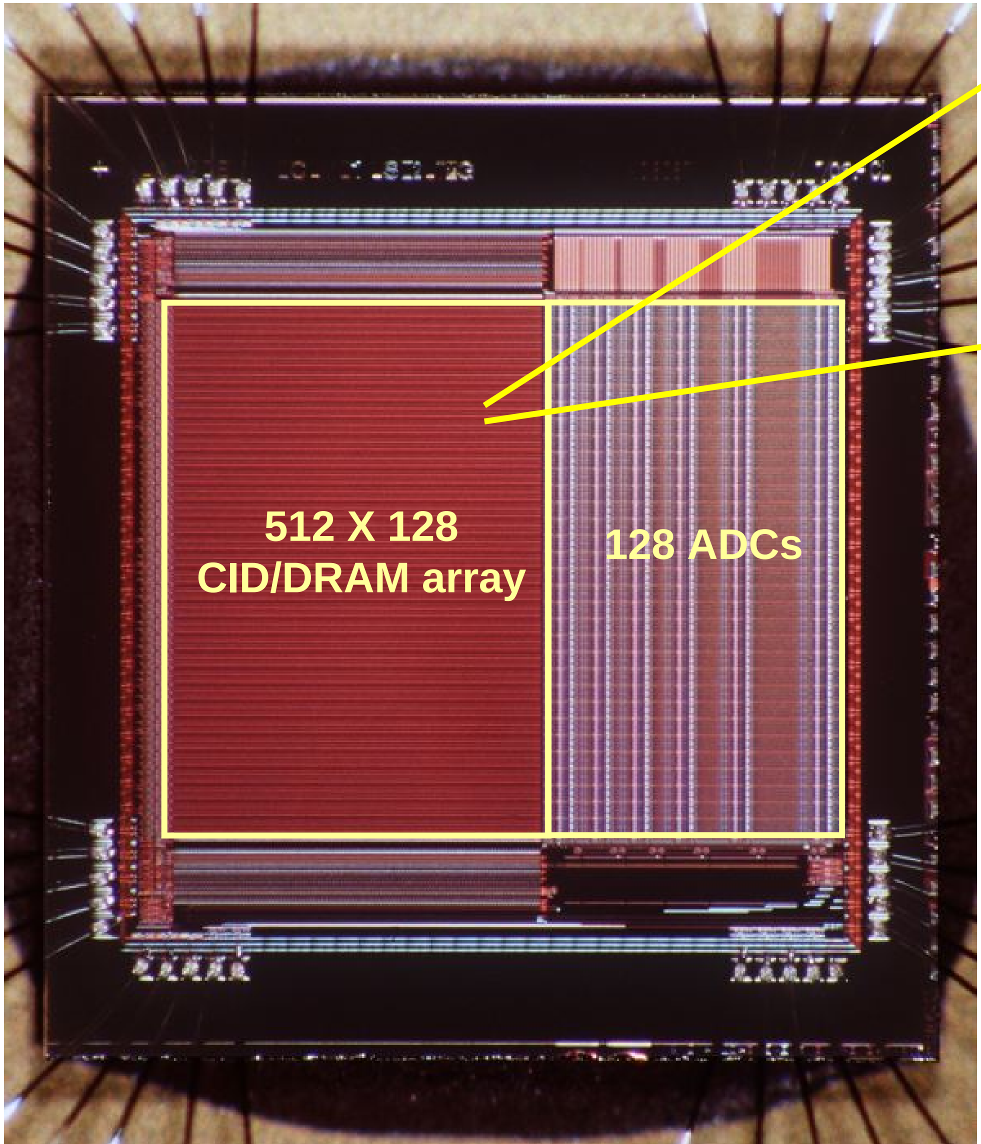
Digital LMS adaptive 3-D bearing estimation

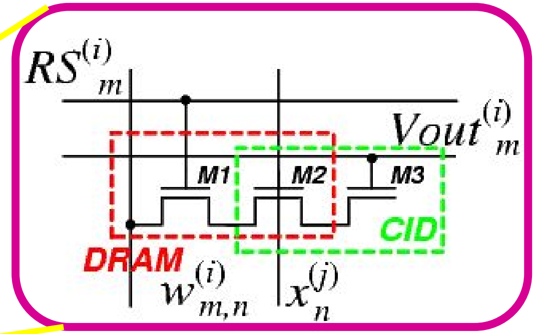
2µsec resolution at 2kHz clock

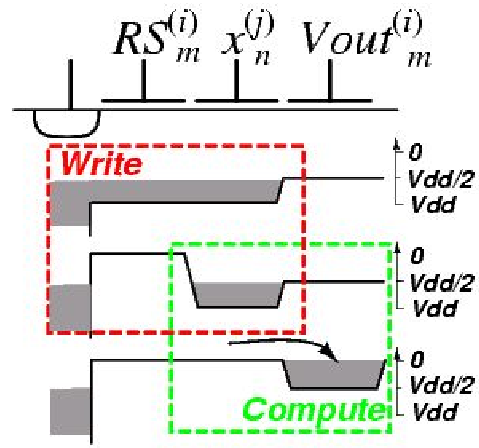
30µW power dissipation

The *Kerneltron*: Support Vector "Machine" in Silicon

Genov and Cauwenberghs, 2001







- *512 inputs, 128 support vectors*
- *3mm X 3mm in 0.5um CMOS*
- *"Computational memories" in hybrid DRAM/CCD technology*
- *Internally analog, externally digital*
- *Low bit-rate, serial I/O interface*
- *6GMACS throughput @ 6mW power*