Ania Mitros, Telluride Neuromorphic Workshop 2003

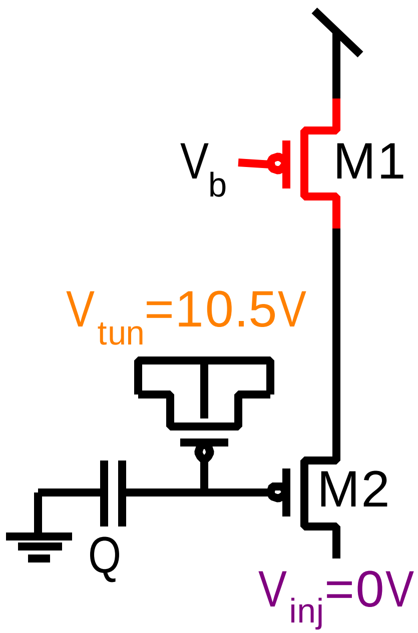
# Examples of storing a weight

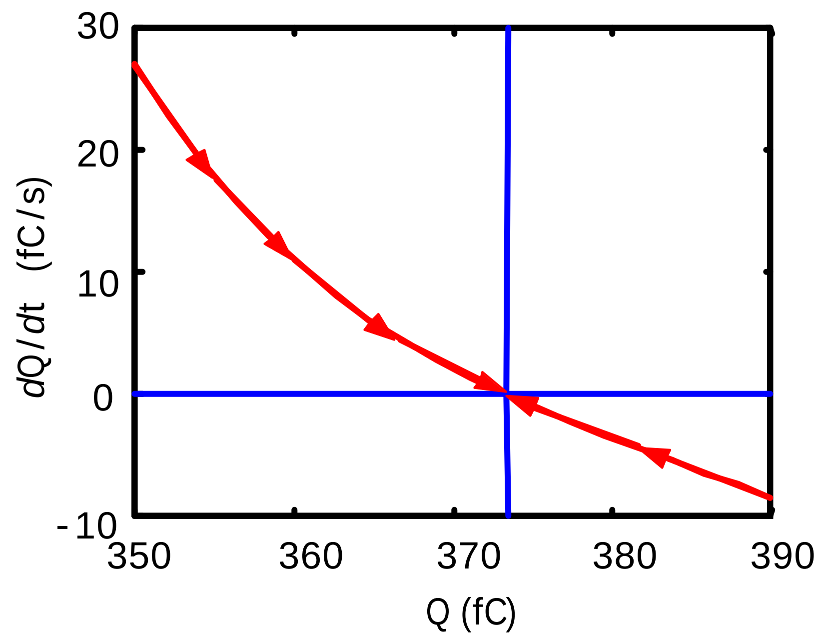
- As charge
- As current
  - storing a given current
  - spike-based incremental update with $\Delta W = func(W)$
- Storing a voltage
  - incremental update with $\Delta W$ independent of W
  - E-pots

Ania Mitros, Telluride Neuromorphic Workshop 2003

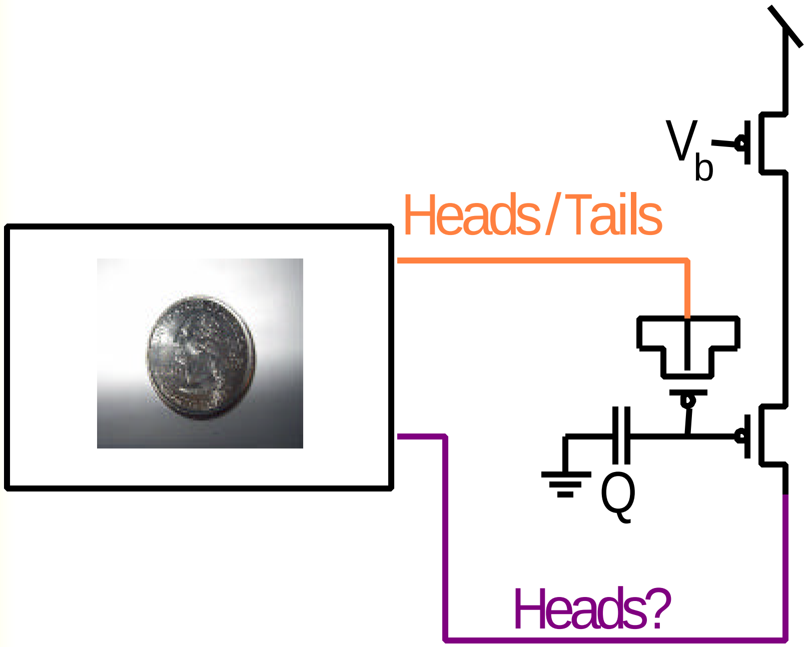
# Storing weight as charge

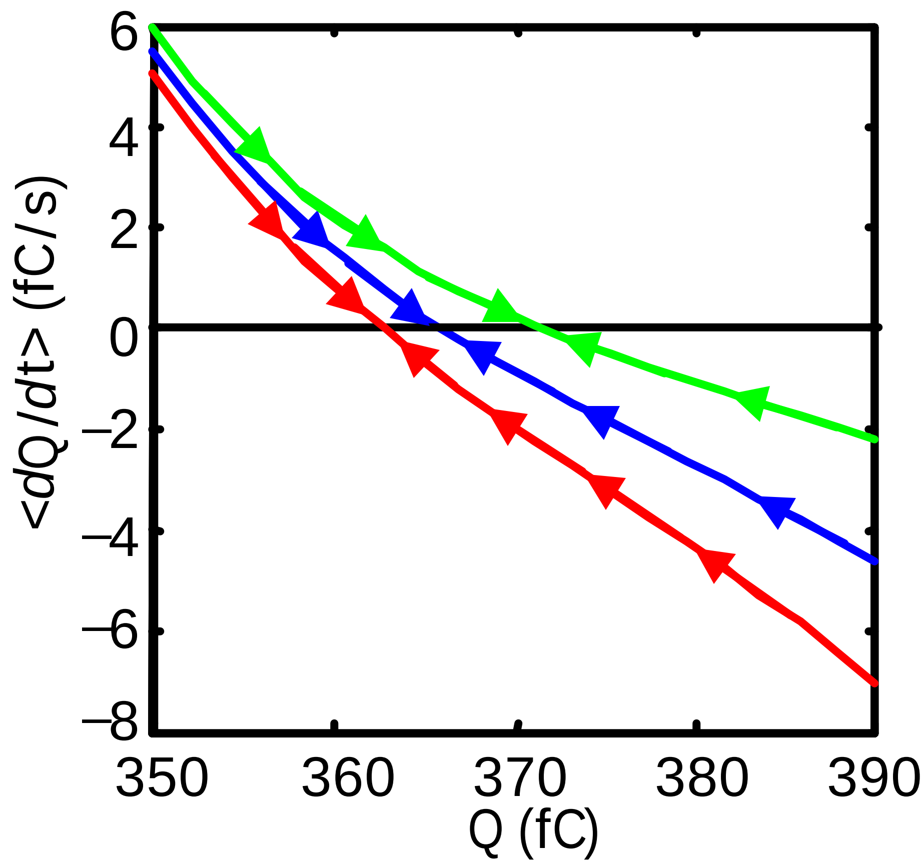
- Charge Q encodes state of system
- Weight set by competition between tunneling and injection (tunneling increases charge, injection decreases charge)



# Coin flipping




Q encodes the coin’s bias

from work by David Hsu

Ania Mitros, Telluride Neuromorphic Workshop 2003

# Spike-based equilibrium point

- Equilibrium weight $\left\langle \frac{dQ}{dt} \right\rangle = 0$
- Function of statistics of spike inputs $V_{tun}$ and $V_{inj}$

$$Q_{eq} = \left( \frac{P\left(\neg V_{inj}\right)}{P\left(V_{tun}\right)} \right)^{\frac{1}{b+c}}$$

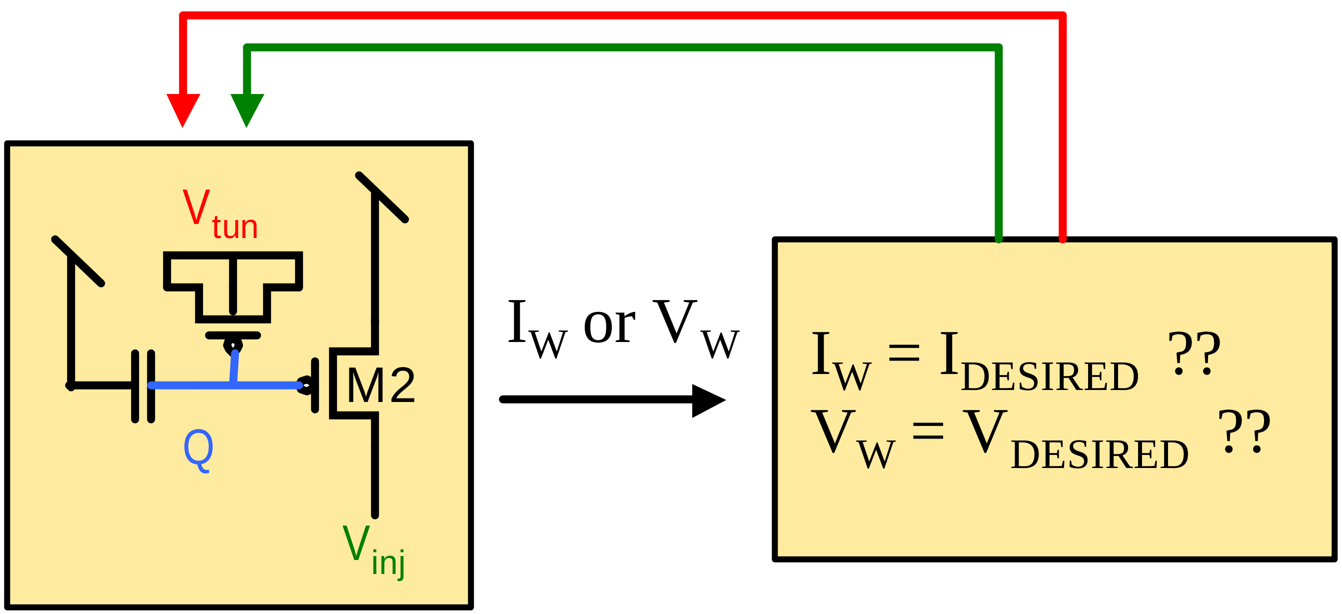
- Circuit learns power-law probability ratio
- $V_{inj}$ and $V_{tun}$ can be controlled by arbitrary logic functions
- Problems:
  - cannot read out Q directly
  - mismatch between tun & inj

# Using feedback

Since:

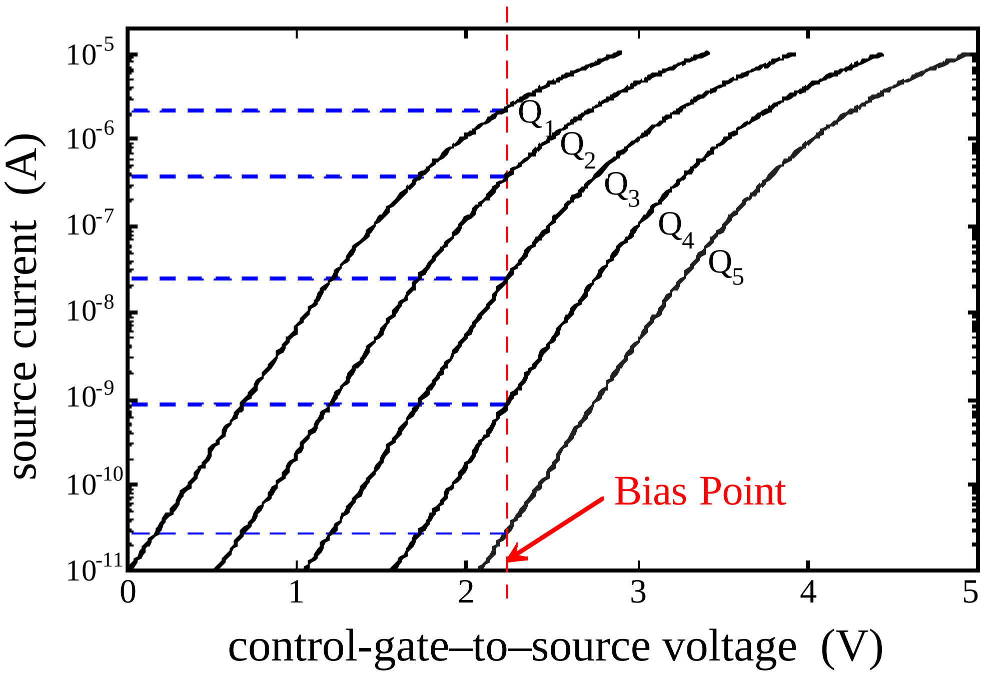
1. $I_{TUN}$ and $I_{INJ}$ are exponential
2. $I_{TUN}$ and $I_{INJ}$ are poorly matched

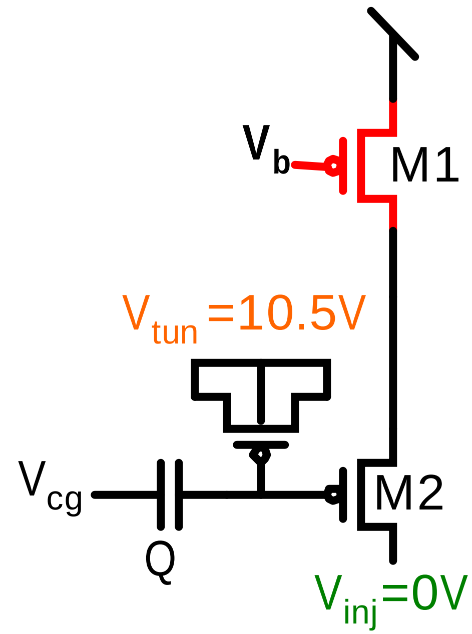
one option is to use feedback when storing a weight.



# Storing weight as current

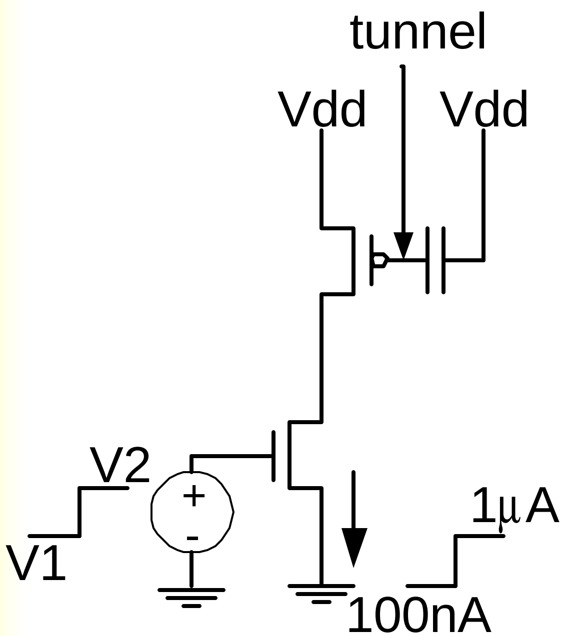
- Charge on floating gate shifts pFET transfer function
  - increase charge by electron tunneling (curve shifts right)
  - decrease charge by electron injection (curve shifts left)

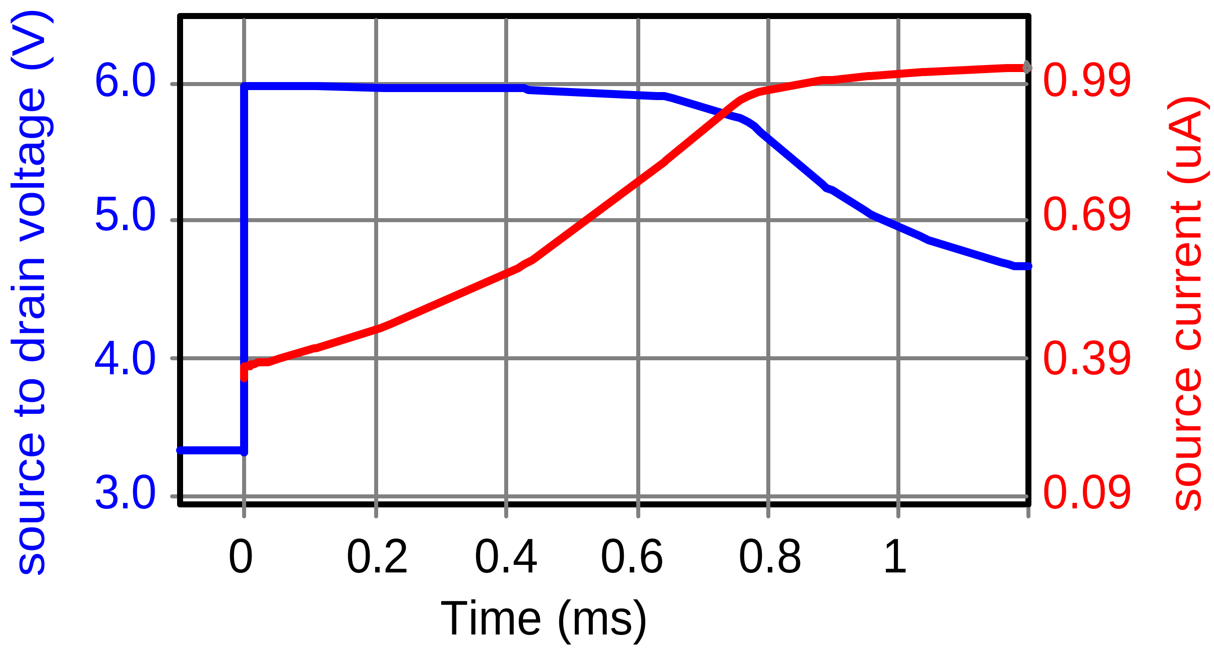


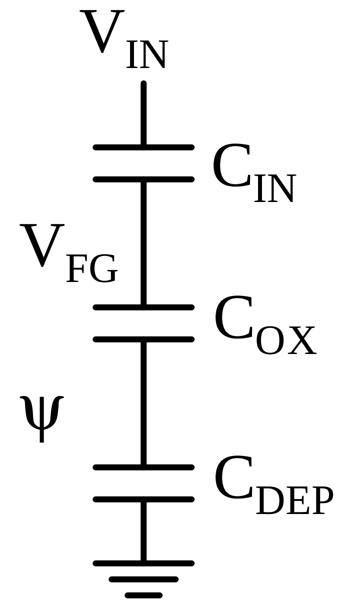
# Self-convergent programming

- Weight stored as current
- Current to be stored is input; then can be disconnected.
- Use tunneling to erase weight, injection to write



# $I_{out} = \text{Weight} * I_{baseline}$




General transistor current equation:

$$Ids = \frac{W}{L} * Io * e^{\frac{k * Vg}{Ut}}$$

## Floating gate voltage:

$$V_{FG} = \frac{Q_{FG}}{C_T} + \frac{C_{IN} V_{IN}}{C_T}$$

## Floating gate transistor equation:

$$Ids = \frac{W}{L} \bullet Io \bullet e^{\left[\frac{k(Q_{FG} + C_{IN} V_{IN})}{C_T \bullet Ut}\right]} = \frac{W}{L} \bullet Io \bullet W \bullet e^{\left[\frac{k' V_{IN}}{Ut}\right]}$$

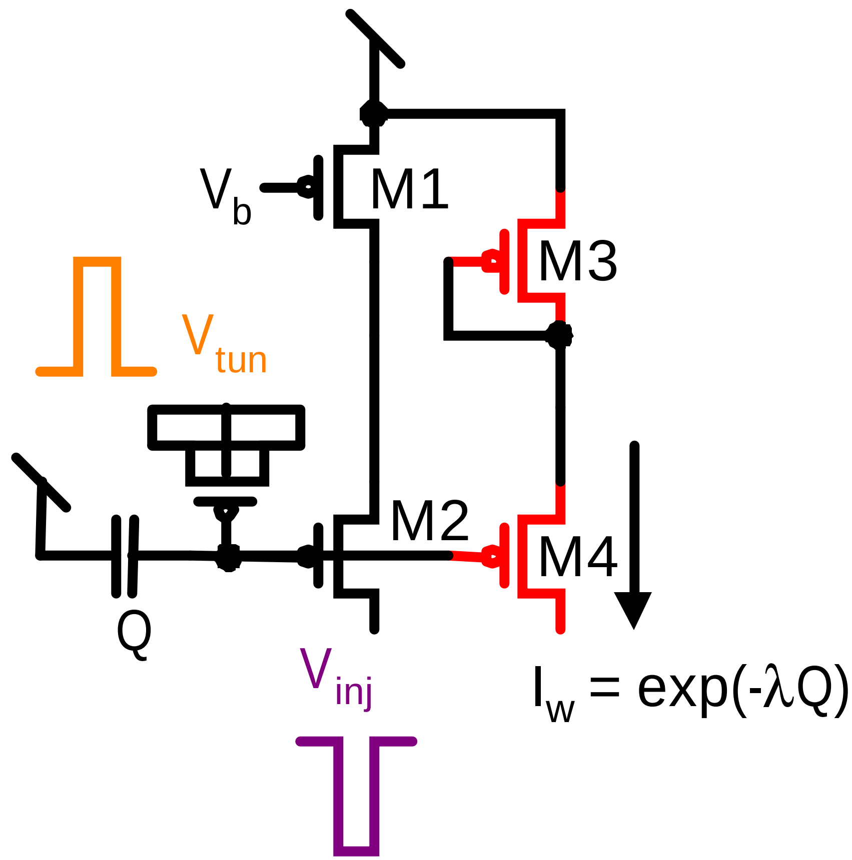
weight

# W as current: incremental updates

Store weight as current $I_w$ through M4

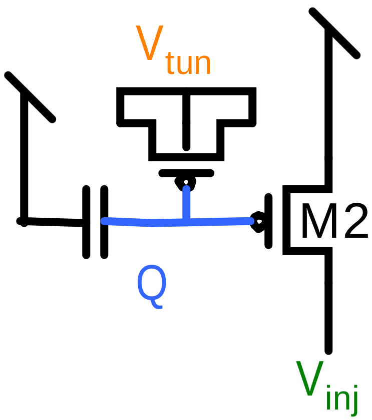
$I_w = \exp(-l\,Q)$

- Spike-based learning signals $V_{tun}$,$V_{inj}$ signal presence or absence of discrete events
- Learning rule: update magnitude depends on previous weight




$$\left\langle \frac{dI_w}{dt} \right\rangle \propto {I_w}^{1-c} P\left(\neg V_{inj}\right) - {I_w}^{1+b} P\left(V_{tun}\right)$$

# Incremental Update




What we've seen so far:

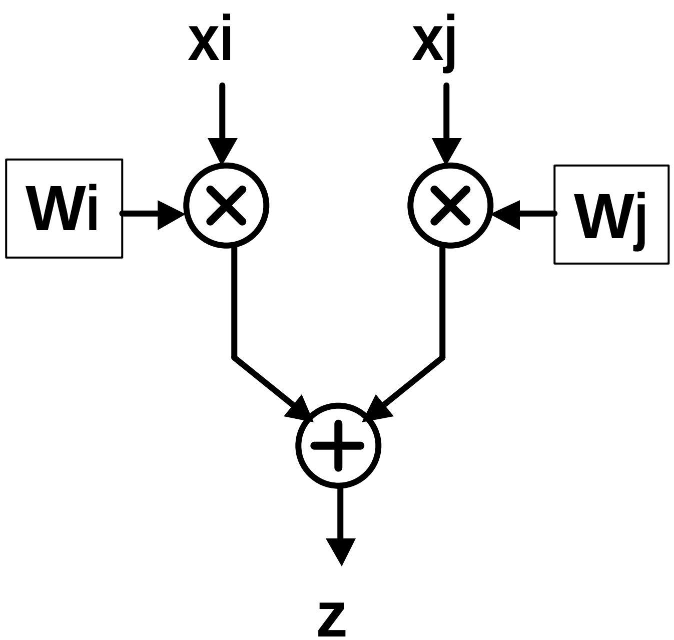
- Charge Q stored on FG, possibly encoding current through M2
- Tunneling and injection are exponential in Vfg; thus current weight W affects the size of dW if we simply pulse Vtun or Vinj
- But some learning rules require size of update to be based only on input and error: $\Delta wj = \alpha * xj * err$

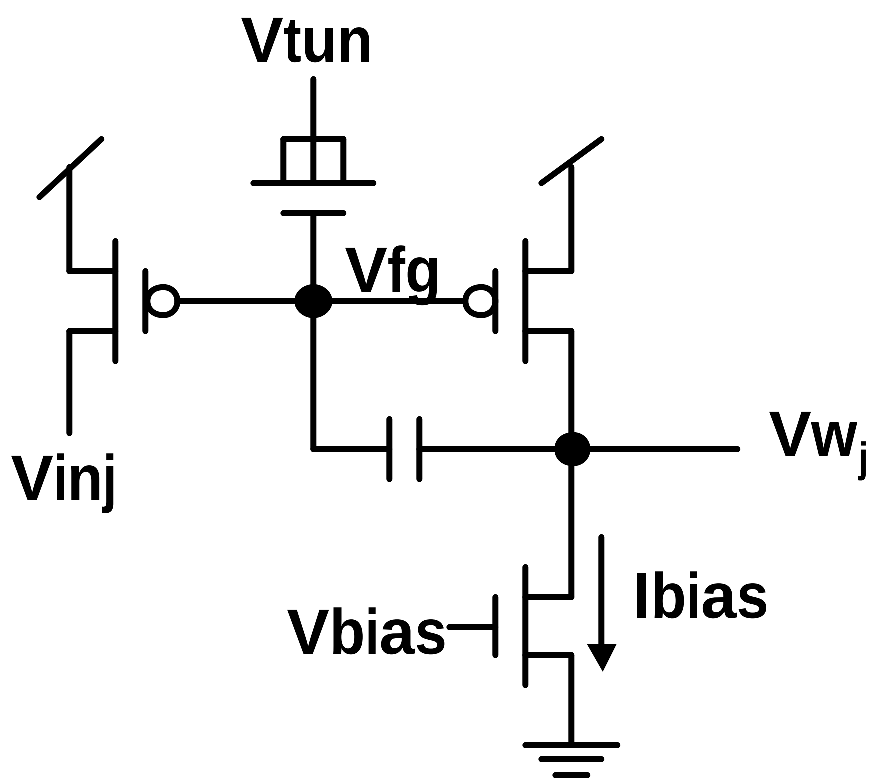
Ania Mitros, Telluride Neuromorphic Workshop 2003

# Weight-Independent Updates

• Feedback keeps Vfg constant, so a single pulse always has the same effect

• Weight stored as output voltage $Vw_j$

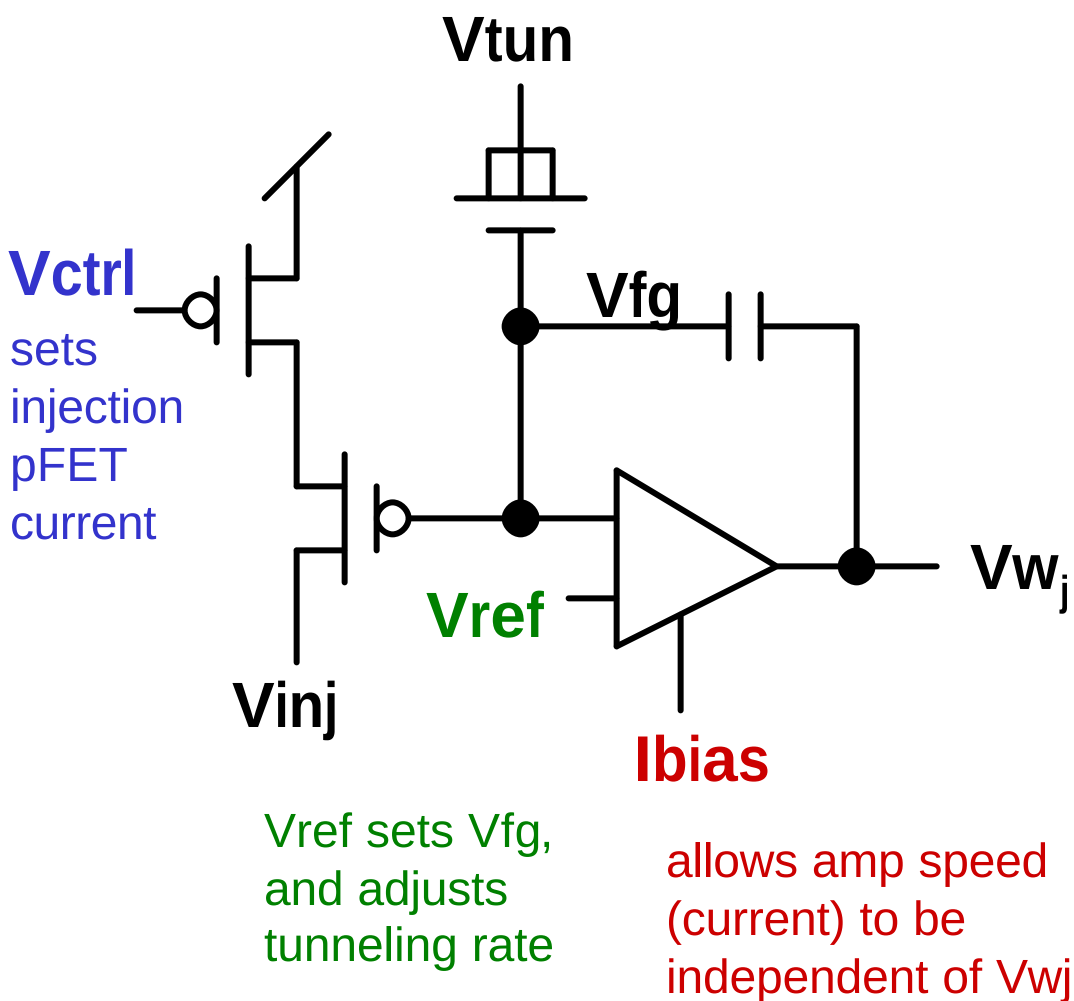



Dwj= a *xj*err

from Miguel Figueroa

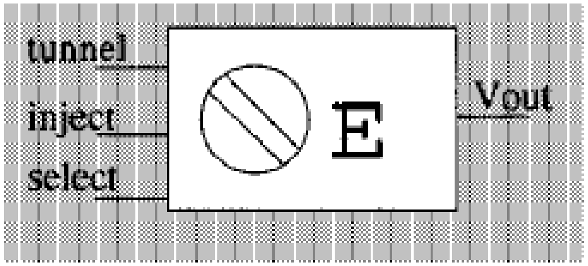
# Correcting matching of $I_{TUN}/I_{INJ}$

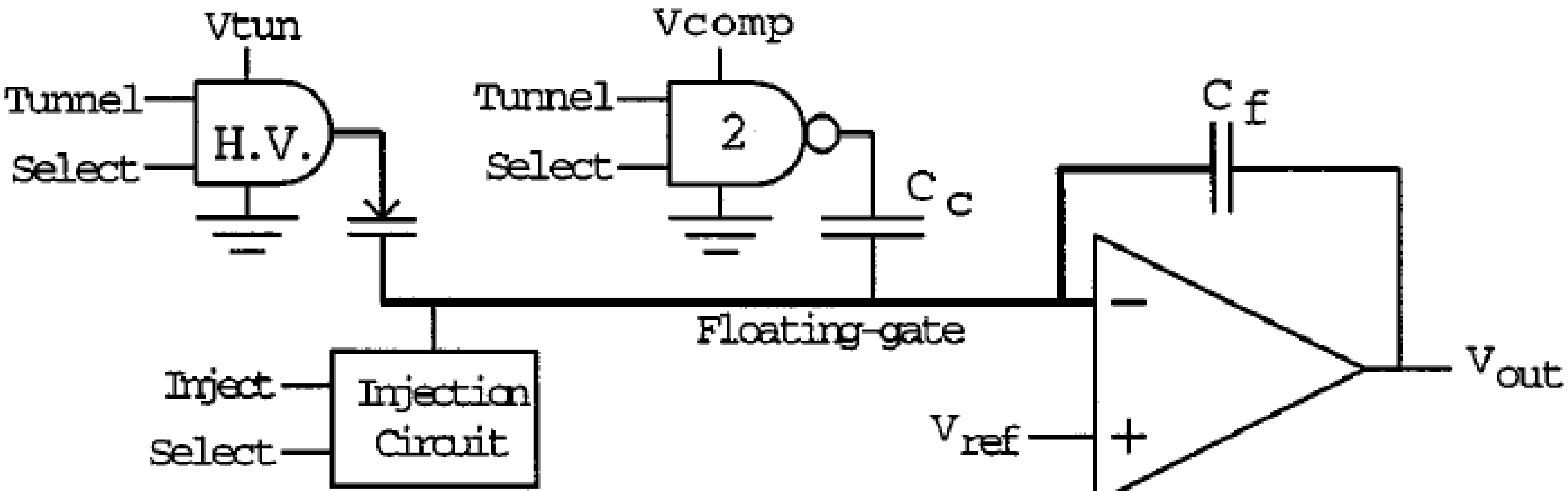



To make sure each $I_{TUN}$ pulse deposits an equal amount of charge as one $I_{INJ}$ pulse, we adjust inj and tun rate individually for each structure. A synapse transistor (not shown) can store value of Vctrl and Vref for each struct.

from Miguel Figueroa

# E-pots: long-term voltage storage




See Harrison, Bragg, Hasler, Minch, Deweerth, "A CMOS Programmable Analog Memory-Cell Array Using Floating-Gate Circuits", IEEE Trans. Circ. & Sys., Jan 2001

# E-pot plots

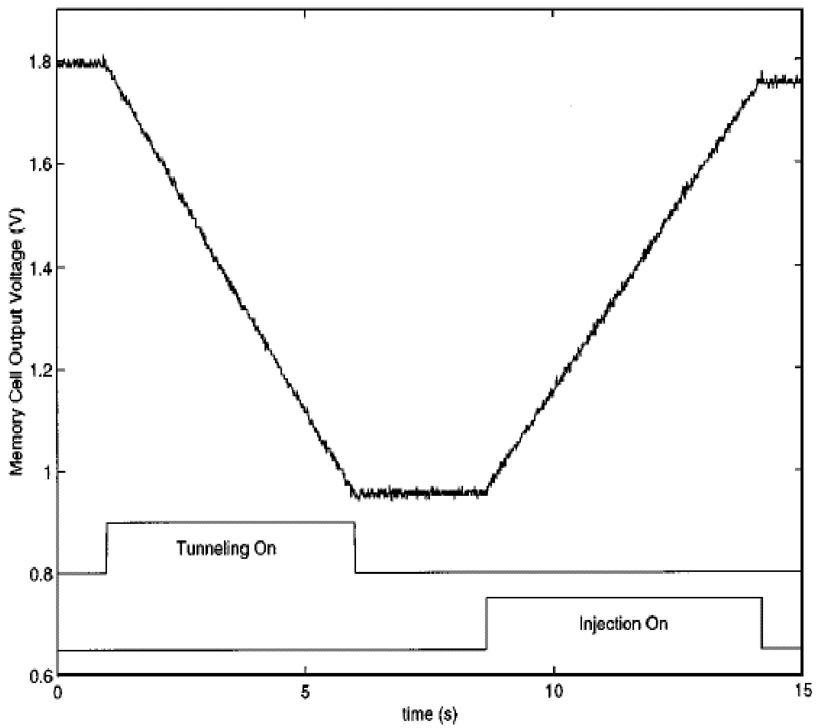



Fig. 3. Experimental measurement illustrating e-pot operation. Digital signals control tunneling and hot-electron injection, moving the e-pot output voltage up or down smoothly.

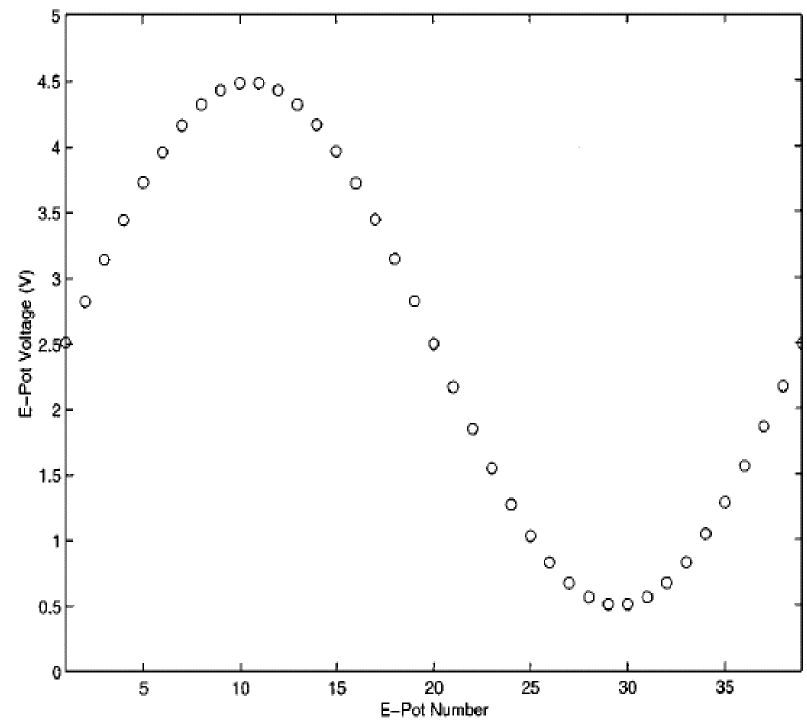



Fig. 4. Output voltages from all 39 e-pots from our 1.2-$\mu$m chip, after each element in the array had been programmed to a voltage proportional to the cosine of the e-pot position number. No crosstalk was observed between e-pot elements during programming.

See Harrison, Bragg, Hasler, Minch, Deweerth, "A CMOS Programmable Analog Memory-Cell Array Using Floating-Gate Circuits", IEEE Trans. Circ. & Sys., Jan 2001

# Summary thus far

- Store weight as charge (equilibrium reached when $I_{TUN}=I_{INJ}$)
- Store a copy of a current (with feedback)
- Increment weight using inj and tun spikes, and correcting mismatch between $I_{TUN}$ & $I_{INJ}$
- Storing a copy of voltage
- Weight-independent updates (keep $V_{FG}$ constant)

# Final suggestion

Use global tunneling as erase, and locally controlled injection to program

- Locally controlled tunneling requires either:
  - switching high voltages
  - local charge pumps (layout eats space)
- Global tunneling requires only small tunneling junction at each FG