Project

Digital Circuits
- Flip flops
- current

Regulator

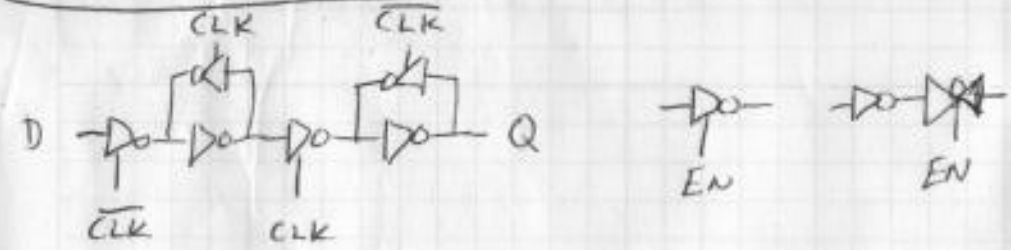
PGA gain

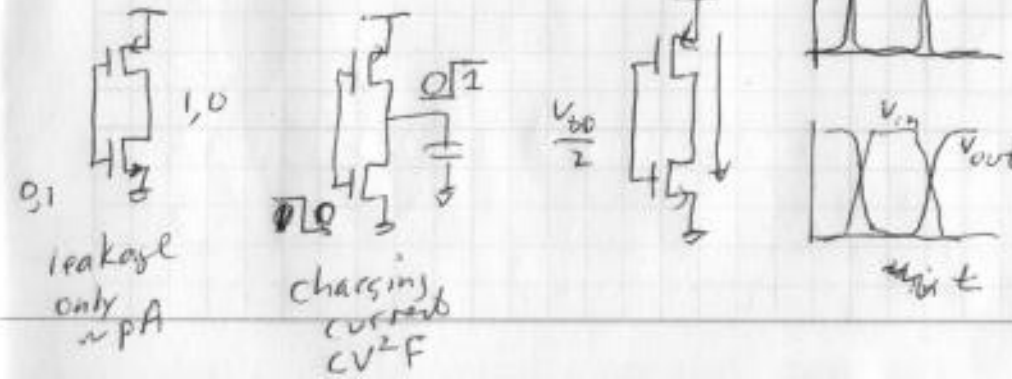
C program interface

Digital circuits, micro processor current

Flip-flops sample the input D on a rising edge of CLK

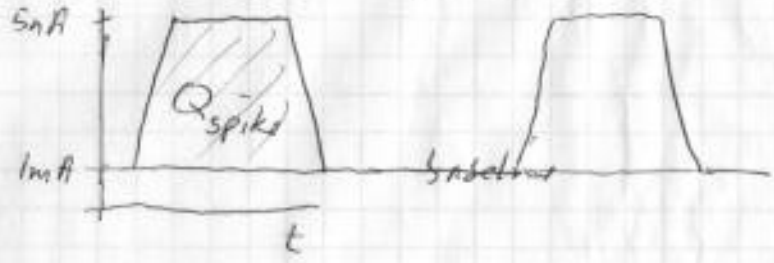
Combinational logic consumes power as it settles





Result: baseline leakage

spikes on rising edges



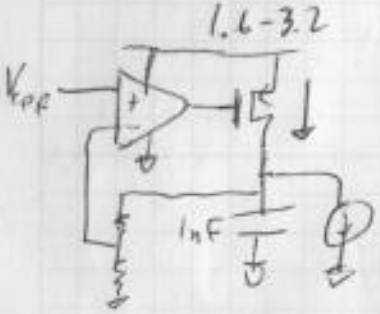
Q_{spike} CV + crowbar

your processor $(4mA)(40\mu s) = 160pC$

$= 0.16nC$

$Q = CV$ $\quad \Delta V_{spike} = \frac{Q}{C} = \frac{0.16nC}{1nF} = 0.16V$

$\approx 10\%$ supply



$\bar{I} = 1mA + (0.16nC)(10mHz)$

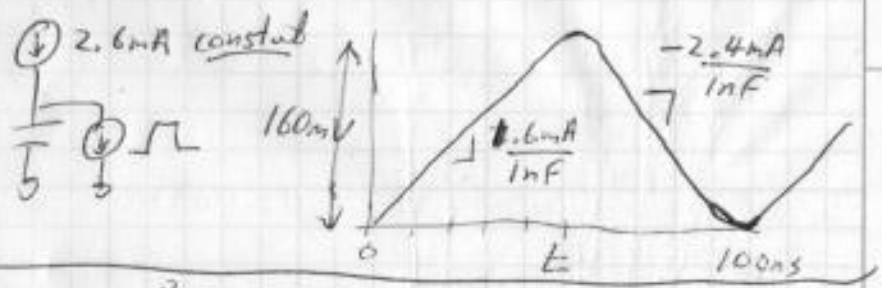
$= 2.6mA$

need average current 2.6mA

$\frac{2.6mA}{10mHz} = 260\frac{\mu A}{mHz}$ about right for 0.18μm

$(100K\,gates)(10\frac{fC}{gate}) = 1nF$

most don't switch on any given $\uparrow$ CLK

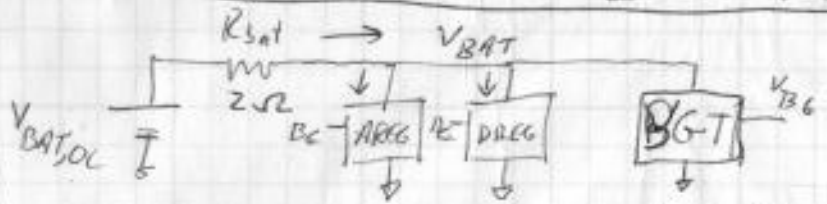


2 stage poles

2 extremes:

i) op-amp very slow.
Pmos gate tracks average current
(as just discussed)

2) op-amp very fast
Pmos gate & current track demand.



If DREG tracks digital demand perfectly
(case 2) then V_{BAT} varies by

$(2\,\Omega)(4mA) = 8mV$

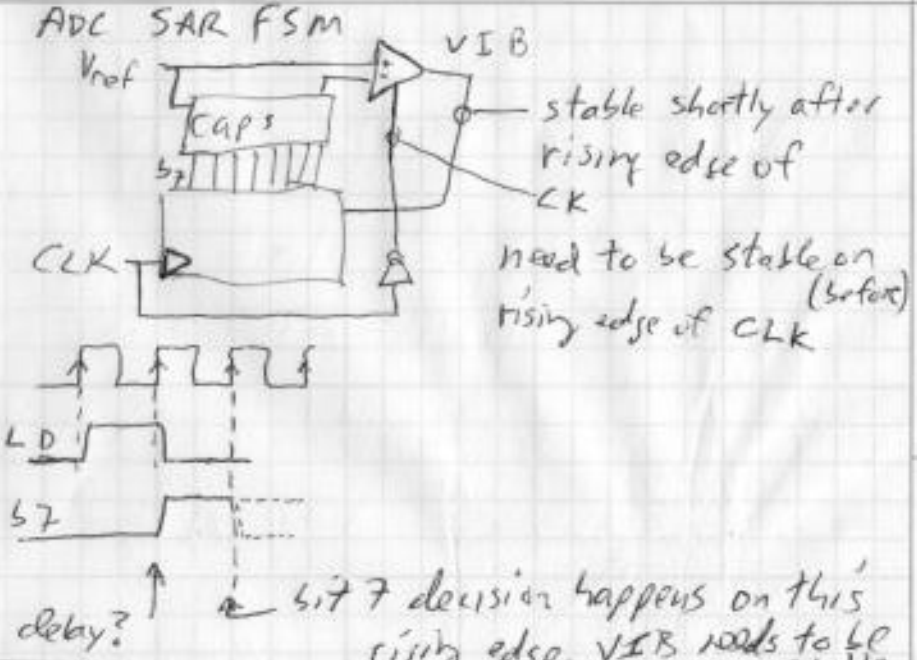
trouble? Depends on "Power Supply Rejection"

At low frequencies, feedback loop will counteract effect of supply on output. V_{DDA}

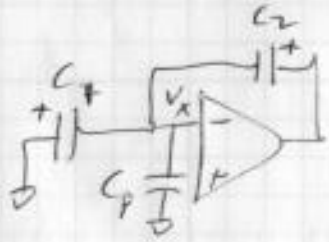
Above unity gain, will show up $\approx$ directly.

op-amp slow (case 1)

ADC SAR FSM



PGA gain

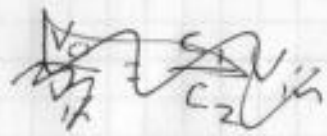


During ϕ_1 $\quad Q_- = -V_{in} C_1$

During ϕ_2 $\quad Q_- = -(V_o - V_x) C_2 + V_x (C_1 + C_p)$

$$= -C_2 V_o + V_x (C_1 + C_2 + C_p)$$

$V_x = \frac{-V_o}{A}$ $\quad = -C_2 V_o - \frac{C_1 + C_2 + C_p}{A} V_o$



$$V_{in} C_1 = \left(C_2 + \frac{C_1 + C_2 + C_p}{A}\right) V_o$$

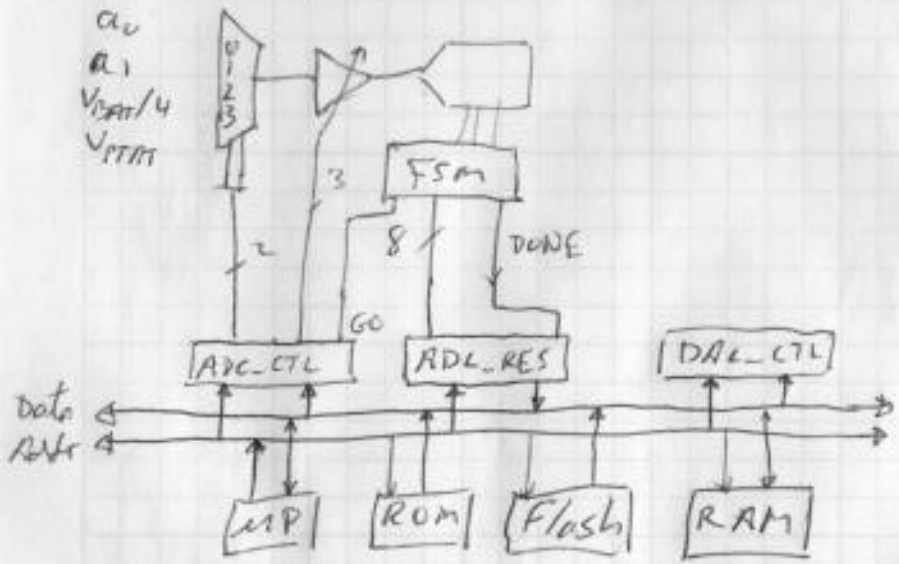
$$= C_2 \left(1 + \frac{C_1 + C_2 + C_p}{A}\right) V_o$$

$$= C_2 \left(1 + \frac{1}{Af}\right) V_o$$

$$\frac{V_o}{V_{in}} = \frac{C_1}{C_2} \frac{1}{1 + \frac{1}{Af}} \qquad f = \frac{C_2}{C_1 + C_2 + C_p}$$

$$\approx \frac{C_1}{C_2} \left(1 - \frac{1}{Af}\right)$$

to set error $< \varepsilon$ need $Af > \frac{1}{\varepsilon}$




```
#define ADC_CTL   0x1000  //32 bit words
#define ADC_RES   0x1004
#define DAC_CTL   0x1008
#define TEMP      3

int gain = 2;
int mux = TEMP;
int go = 1;

*ADC_CTL = (mux<<4)|(gain-1)<<1|go;

while (*ADC_RES & 1 == 0); //wait for done

int temp = (*ADC_RES)>>1;
```