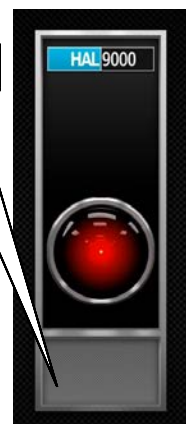
Lecture 15

Potpourri: Bluetooth, Clocking, Power, FFTs, etc…

Moving Forward

- Project Proposal Due by today 5pm
- 11/07 2:30-5pm and 11/09 2:30-4pm are Presentation Days:
  - Attendance Mandatory

Outline (October 31, 2017)



Topics:

- Bluetooth:
  - One more common communication protocol that you may interface with
- Power Consumption/Energy and Clocking in our systems
- Fast Fourier Transforms (with Examples!)

Bluetooth vs. Bluetooth Low Energy (BLE)

- Bluetooth was created in ~1994
- Originally supposed to be a drop-in replacement to RS232,only wireless, and you can still see vestiges of that heritage in documentation and some of its protocols.
- Works on ISM Band (2.4 GHz…shares with Wifi)

Bluetooth is like USB, many flavors, speeds, etc… Is a Multi-layered stack

- Every device has a unique* 48-bit identifier (like a MAC Address)
- Handshakes and pairing layer
- Public key encryption (192 bit DHKE), followed by 128bit on latter versions
- Actual bits are sent using PSM…either QDPSK or even 8DPSK through a Gaussian filter (don't hop from phase to phase instantaneously), and frequency hopping!:
  - Meaning bits are encoded using phase of carrier wave (improves throughput but requires more complex send/receive circuitry)
    - (QDPSK: four phases allows two bits at once)
    - (8DPSK: eight phases allows three bits at once)

Actually unique to the particular device, not **type of device like in i2C. Ideally no two devices will share this. They thought ahead 2^48 gives us 280 trillion possibilities, so up to 46,000 bluetooth headsets per person*

Drop-in modules exist

HC-0X Series

- Because of its initial goal of being a RS232 serial replacement, there are modules which literally take in UART (at 115.2 kbps let's say) and will convert to bluetooth and send and receive/convert back to UART
- Add a second one that is mated to it on the other end, and you can get a wireless RS232 link.
- Additional feature to maybe add to projects if helpful

BLE: Bluetooth Low Energy

Bluetooth SMART

- Complete Rethink of what Bluetooth is meant to be used for
- Can't send as much data reliably, but power usage is significantly reduced

Technical specification	Classic Bluetooth technology	Bluetooth Smart technology
Distance/range (theoretical max.)	100 m (330 ft)	>100 m (>330 ft)
Over the air data rate	1–3 Mbit/s	125 kbit/s – 1 Mbit/s – 2 Mbit/s
Application throughput	0.7–2.1 Mbit/s	0.27 Mbit/s
Active slaves	7	Not defined; implementation dependent
Security	56/128-bit and application layer user defined	128-bit AES with Counter Mode CBC-MAC and application layer user defined
Robustness	Adaptive fast frequency hopping, FEC, fast ACK	Adaptive frequency hopping, Lazy Acknowledgement, 24-bit CRC, 32-bit Message Integrity Check
Latency (from a non-connected state)	Typically 100 ms	6 ms
Minimum total time to send data (det. battery life)	100 ms	3 ms [36]
Voice capable	Yes	No
Network topology	Scatternet	Scatternet
Power consumption	1 W as the reference	0.01–0.50 W (depending on use case)
Peak current consumption	<30 mA	<15 mA
Service discovery	Yes	Yes
Profile concept	Yes	Yes
Primary use cases	Mobile phones, gaming, headsets, stereo audio streaming, smart homes, wearables, automotive, PCs, security, proximity, healthcare, sports & fitness, etc.	Mobile phones, gaming, smart homes, wearables, automotive, PCs, security, proximity, healthcare, sports & fitness, Industrial, etc.

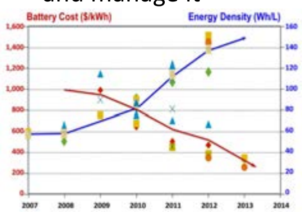
Power

While we're not really focussing on this in final projects, maybe think about this as another way to characterize your device's performance

Problem: Energy Consumption

The Energy Problem

- It is getting better, but there's no Moore's Law for Batteries
- We need to understand where power goes and manage it



https://forum.cosmoquest.org/showthread.php?166243-Energy-Density

7.5 cm^3 AA battery

Alkaline: ~10,000J

What can One Joule of energy do?

Mow your lawn for 1 ms

Send a 1 Megabyte file over 802.11b

Operate a mediocre processor for ~ 7s

Sleep ESP32 for 1.4 days

Run ESP32 for 2 seconds

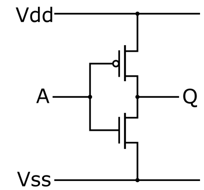
Digital Power Consumption

Dynamic power consumption

$$P = \alpha_{0\to1} C V^2 f + V I_{leak}$$

Static power consumption

- P : total power consumed
- $\alpha_{0\to1}$: fraction of gates switching
- C : Capacitance of gates
- V : Operating voltage (V_{dd})
- f : frequency of operation
- I_{leak}: Leakage Current:
  - Sub-threshold leakage
  - Gate-Leakage



What do we have control over?

- Dynamic Power usage is more closely tied to how we use the system:
  - Design, data structures, etc…
  - Clock
  - Temperature
  - Etc…
- Static Power usage is more closely tied to actual system fabrication and capabilities, but our usage of it can also factor in

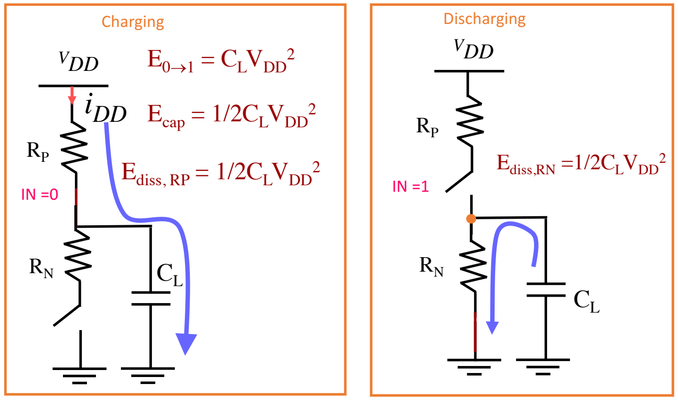
Given Fixed Hardware: Power Reduction Strategies

$$P = \alpha_{0->1} C_L V_{DD}^2 f$$

- Reduce Transition Activity or Switching Events
- Reduce Capacitance (e.g., keep wires short)
- Reduce Power Supply Voltage
- Frequency is sometimes fixed by the application, though this can be adjusted to control power

Optimize at all levels of design hierarchy

Dynamic Energy Dissipation



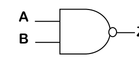
The switches are FETs

$$P = C_L V_{DD}^2 f_{clk}$$

The Transition Activity Factor

$\alpha_{0\to1}$

Current Input	Next Input	Output Transition
00	00	1 -> 1
00	01	1 -> 1
00	10	1 -> 1
00	11	1 -> 0
01	00	1 -> 1
01	01	1 -> 1
01	10	1 -> 1
01	11	1 -> 0
10	00	1 -> 1
10	01	1 -> 1
10	10	1 -> 1
10	11	1 -> 0
11	00	0 -> 1
11	01	0 -> 1
11	10	0 -> 1
11	11	0 -> 0



Assume inputs (A,B) arrive at f and are uniformly distributed (not guaranteed at all)
What is the average power dissipation?

$$\alpha_{0\to1} = 3/16$$

$$P = \alpha_{0\to1} C_L V_{DD}^2 f$$

System Level Power Reduction Strategies

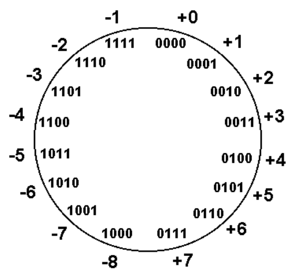
- System level
  - Airplane mode: switch off cell phone/text activity.
  - Display brightness control
  - Variable transmit power level
  - Minimize CPU cycles (encryption costs!)
- Chip level
  - Workload based clock frequency/clock gating
  - Power gating
  - Dynamic voltage scaling (DVS)
  - Multi V_{dd}

Power Consumption Can Be Data Dependent

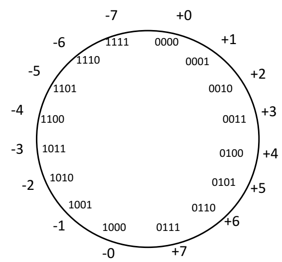
- We don't think about this at the C and up level, but at the bit level it can really matter!
- Is your data encoded in a way such that lots of bits flip lots of the time? (lots of charge/discharge cycles!)
  - Are common transitions using the fewest bit changes?
  - Glitches are no longer an annoyance, but leeches sucking our vital life fluids (power) from our bodies (electrical devices)

Number Representation:
Two's Complement vs. Sign Magnitude

Two's complement



Sign-Magnitude

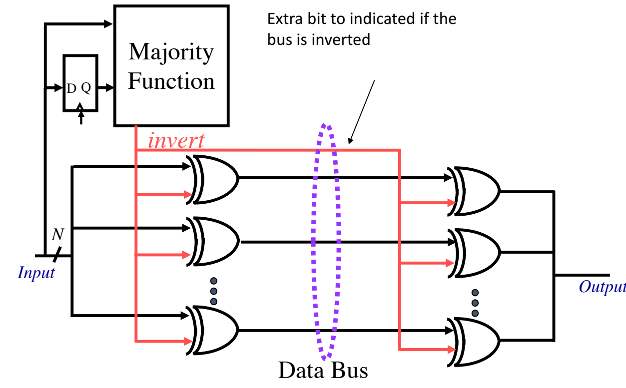


Consider a 16 bit bus where inputs toggles between +1 and –1 (i.e., a small noise input)

Which representation is more energy efficient?

Bus Coding to Reduce Activity

- Minimize bit transitions on high capacitance busses



[Stan94]

Hamming Distance

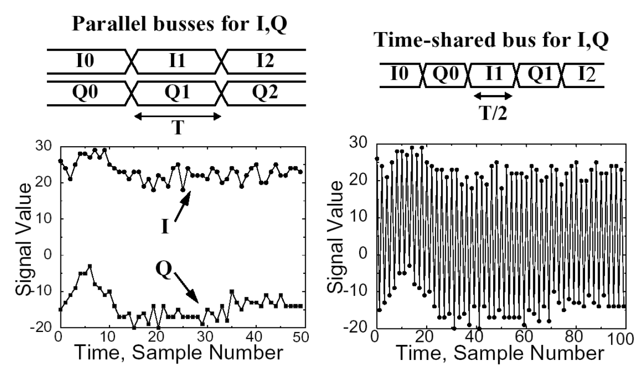
- Reduce Hamming Distance between sequences…don't count up with states using regular binary…use a Gray code perhaps
- Counting to 8 in regular 3bit binary involves 14 total bit changes
- Counting to 8 in 3bit Gray involves 8 total bit changes(big savings)

Count	Transitions
000	3
001	1
010	2
011	1
100	3
101	1
110	2
111	1
	14

Gray code by bit width

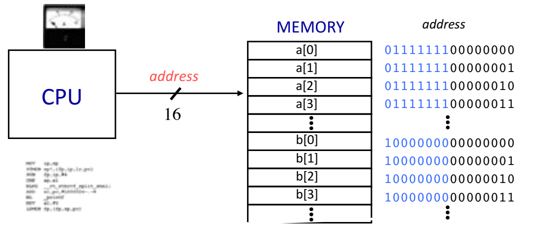
3-bit	4-bit
000	0000
001	0001
011	0011
010	0010
110	0110
111	0111
101	0101
100	0100
2-bit	1100
00	1101
01	1111
11	1110
10	1010
1-bit	1011
0	1001
1	1000

Time Sharing is a Bad Idea (From a power perspective)



Time Sharing Increases Switching Activity

Not just a 6-1 Issue: “Cool” Software




```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {
    a[i] = sin(pi * i /256);
    b[i] = cos(pi * i /256);
}
```

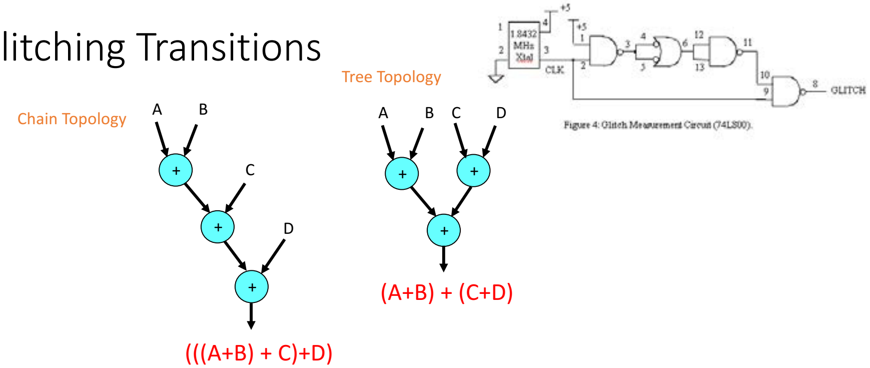
512(8)+2+4+8+16+32+64+128+256
= 4607 bit transitions

```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {a[i] = sin(pi * i /256);}
for (i = 0; i < 255; i++) {b[i] = cos(pi * i /256);}
```

2(8)+2(2+4+8+16+32+64+128+256)
= 1030 transitions

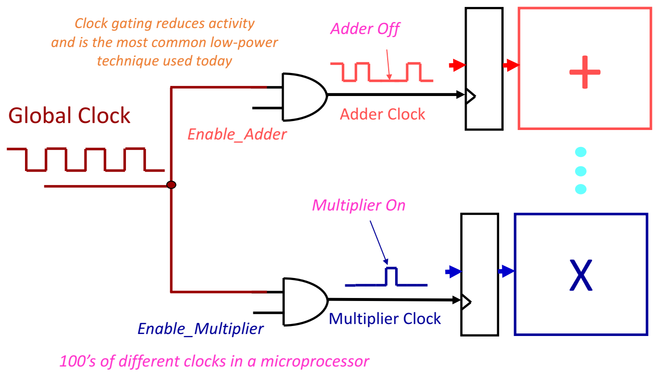
Glitching Transitions



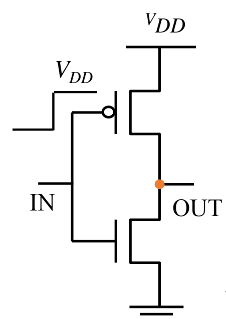
- Balancing paths reduces glitching transitions
- Structures such as multipliers have lot of glitching transitions
- Keeping logic depths short (e.g., pipelining) reduces glitching

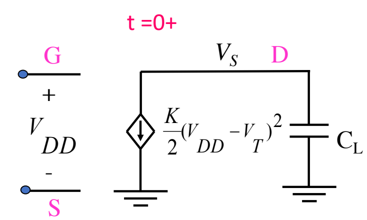
Clock Gating is a Good Idea!

(For energy conservation only…)



Reduce Supply Voltage : But is it Free?

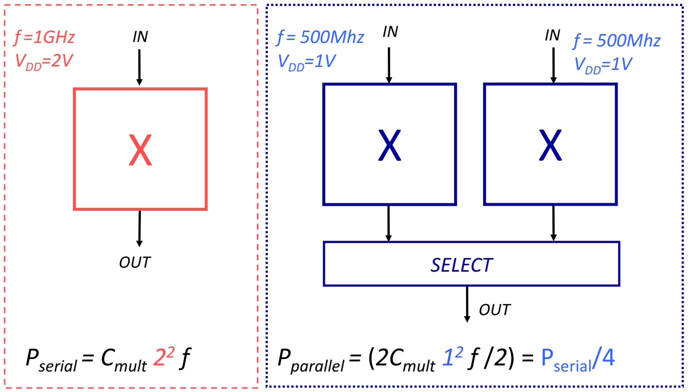




$$Delay = \frac{C_L \cdot \Delta V}{i_D} = \frac{C_L \cdot \frac{V_{DD}}{2}}{\frac{k}{2}(V_{DD}-V_T)^2} \propto \frac{V_{DD}}{(V_{DD}-V_T)^2} \approx \frac{1}{V_{DD}}$$

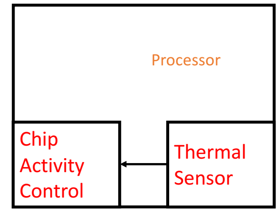
V_{DD} from 2V to 1V, energy ↓ by x4, delay ↑ x2

Transistors Are Free…
(What do you do with a Billion Transistors?)



Trade Area for Low Power

Does your GHz Processor run at a GHz?

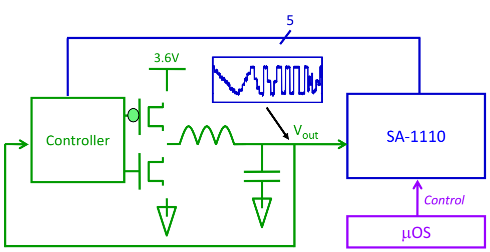


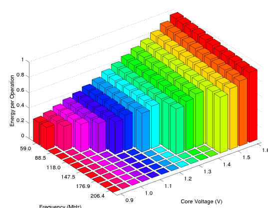
- Note that there is a difference between average and peak power
- On-chip thermal sensor (diode based), measures the silicon temperature
- If the silicon junction gets too hot (say 125 ° C), then the activity is reduced (e.g., reduce clock rate or use clock gating

Use of Thermal Feedback

Dynamic Voltage Scaling on a Processor

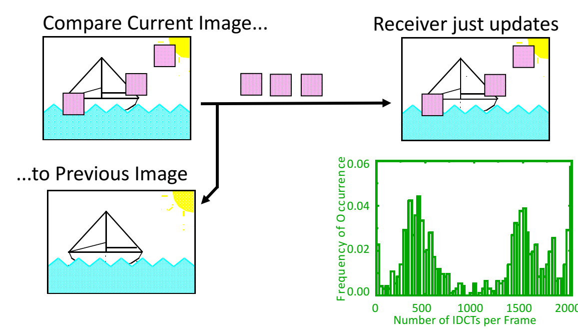
Digitally adjustable DC-DC converter
powers SA-1110 StrongArm core





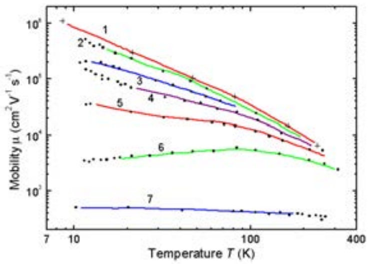
µOS selects appropriate clock frequency based on
workload and latency constraints

Algorithmic Workload



Temperature

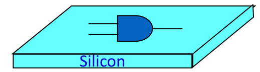
- While some input power gets used for information/computation, etc… a lot is ultimately lost as heat
- As temperature rises, carrier mobility will drop off quickly
- As mobility drops off, current delivered drops off, systems can't charge/discharge as quickly, we run into trouble



http://www.ioffe.ru/SVA/NSM/Semicond/Ge/electric.html

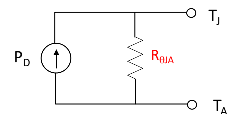
Junction (Silicon) Temperature

Simple Scenario



$T_j - T_a = R_{\theta JA} P_D$

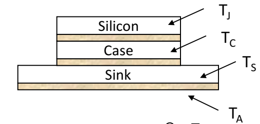
$R_{\theta JA}$ is the thermal resistance between silicon and Ambient

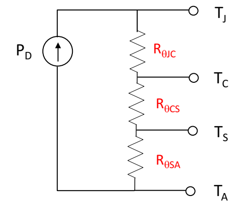


$T_j = T_a + R_{\theta JA} P_D$

Make this as low as possible

Realistic Scenario





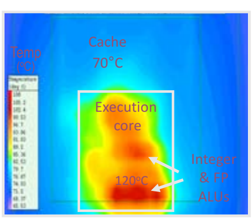
These are thermal model circuits
Not electrical circuits

$R_{\theta CA} = R_{\theta CS} + R_{\theta SA}$

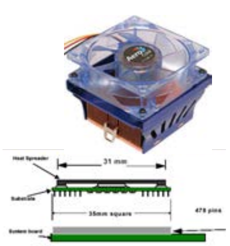
is minimized by facilitating heat transfer (bolt case to extended metal surface – heat sink)

Intel Pentium 4 Thermal Guidelines

- Pentium 4 @ 3.06 GHz dissipates 81.8W! (i7 Haswell 3.2GHz 65W)
- Maximum T_C = 69 °C
- R_{CA} < 0.23 °C/W for 50 C ambient
- Typical chips dissipate 0.5-1W (cheap packages without forced air cooling)

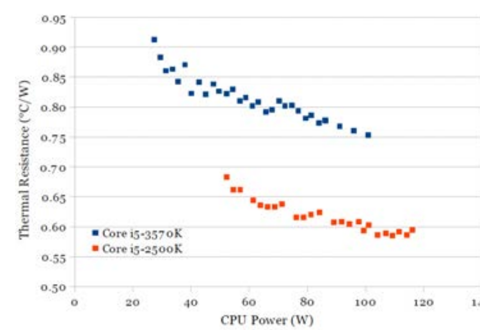


Courtesy of Intel
(Ram Krishnamurthy)



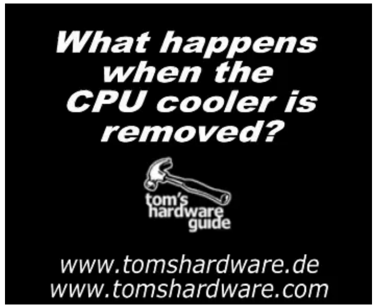
Processor and Core Frequency	Thermal Design Power[1,2] (W)
Processors with VID=1.500V	
2 GHz	52.4
2.20 GHz	55.1
2.26 GHz	56.0
2.40 GHz	57.8
2.50 GHz	59.3
2.53 GHz	59.3
Processors with VID=1.525V	
2 GHz	54.3
2.20 GHz	57.1
2.26 GHz	58.0
2.40 GHz	59.8
2.50 GHz	61.0
2.53 GHz	61.5
2.60 GHz	62.6
2.66 GHz	66.1
2.80 GHz	68.4
Processors with multiple VIDs	
2 GHz	54.3
2.20 GHz	57.1
2.26 GHz	58.0
2.40 GHz	59.8
2.50 GHz	61.0
2.53 GHz	61.5
2.60 GHz	62.6
2.66 GHz	66.1
2.80 GHz	68.4
3.06 GHz	81.8

Thermal Resistance of Recent Cores (i5 Sandy vs. Ivy)

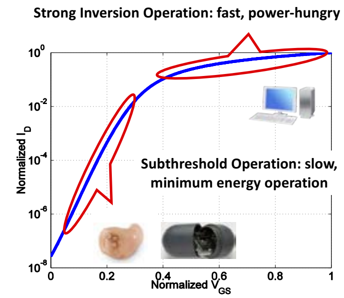


Thermal Resistance of Sandy Bridge (orange) and Ivy Bridge (blue)

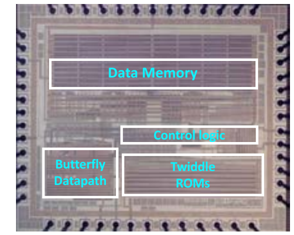
Thermal Shutdown



Low-Power Digital: Sub-Threshold Operation



V_{DD} = 0.18V



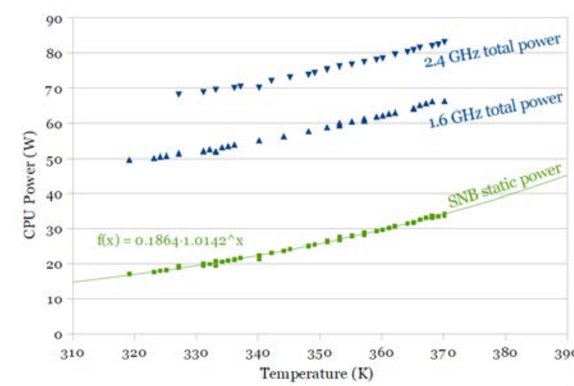
Exploit Sub-threshold Operation ($V_{DD} < V_T$) for Sensor Circuits

Other Considerations

Static Power

Power Consumed

Sandy Bridge Power



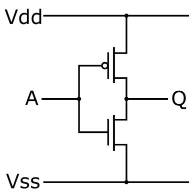
Digital Power Consumption

$$P = \alpha_{0 \to 1} C V^2 f + V I_{leak}$$

Dynamic power consumption

Static power consumption

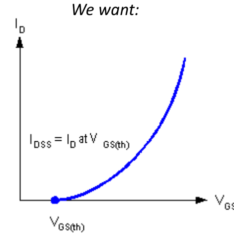
- P : total power consumed
- $\alpha_{0 \to 1}$: fraction of gates switching
- C : Capacitance of gates
- V : Operating voltage (V_{dd})
- f : frequency of operation
- I_{leak}: Leakage Current:
  - Sub-threshold leakage
  - Gate-Leakage



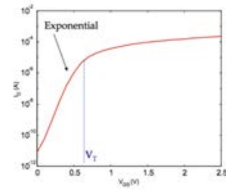
Types of Static Leakage:

Sub-Threshold Leakage:

We want:



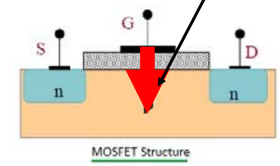
Zoom in around the threshold and we've got:



The fact that no transistor turns off below its threshold voltage

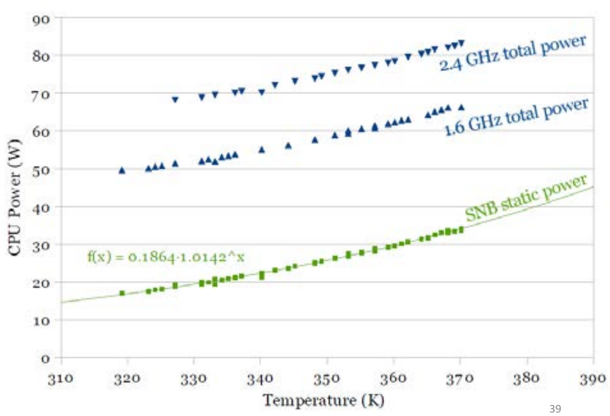
Gate Leakage:

Where an electron or hole just tunnels through the FET gate



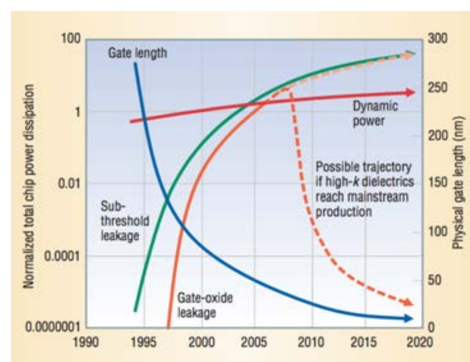
http://wps.prenhall.com/chet_paynter_introduct_6/6/1664/426188.cw/index.html
http://www-inst.eecs.berkeley.edu/~cs150/fa11/agenda/lec/lec22-power.pdf

How much?



Leakage has Gotten so bad

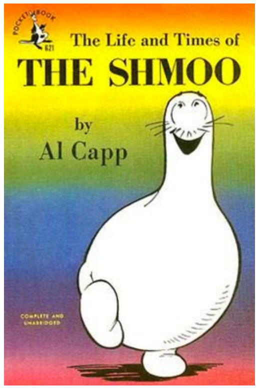
- How bad is it?
- In some contexts, static loss starts to dominate dynamic loss
- This is a really big deal since the primary loss mechanism is beyond the control of implementation design, etc…



From 2011

Leakage Current: Moore's Law Meets Static Power
http://www.ruf.rice.edu/~mobile/elec518/readings/DevicesAndCircuits/kim03leakage.pdf

Aside: Shmoo Plot

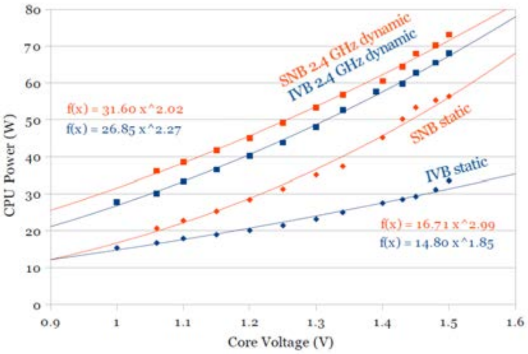


- Sometimes hear plots of various performance specs on semiconductors called "Shmoo" plots
- Called that because they plots look like Shmoos, weird bowling-pin like creatures from Lil Abner, even though they never do
- Anyways sometimes these comparison plots are called Shmoos

*Wikipedia finally explained this to me…pre-semiconductor, Shmoo plots looked like Shmoos with magnetic things

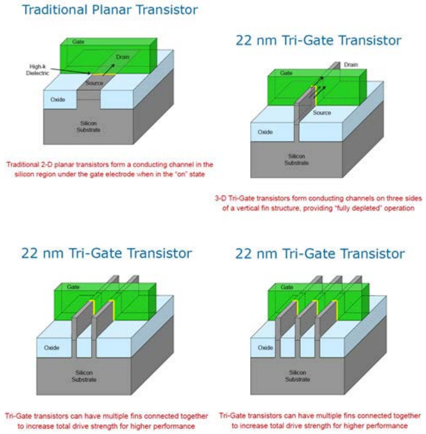
Sandy Bridge vs. Ivy Bridge (32nm vs. 22 nm core i5)

- Sandy Bridge was older model transistor
- Ivy Bridge was 3D transistor



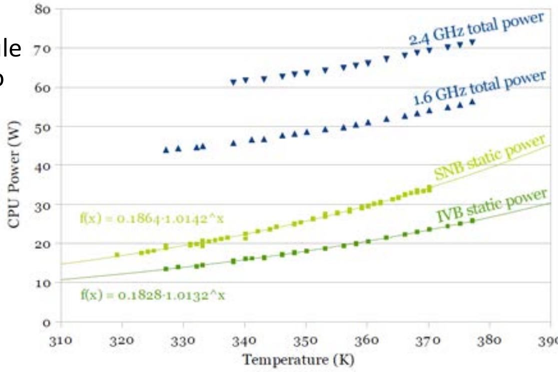
Trigate

- One of the first departures from planar semiconductor fabrication since we started doing it as humans in the early 1950s.
- Was in the pipeline since right around 2000, and finally started coming out in 2014
- Cuts static loss (sub-threshold loss in particular) by 50%



Sandy Bridge vs. Ivy Bridge (32nm vs. 22 nm core i5)

- Intel fell way behind schedule getting their 22nm tech into production, but its trigate devices in IVB, have drastically cut down static power loss



http://blog.stuffedcow.net/2012/10/intel32nm-22nm-core-i5-comparison/

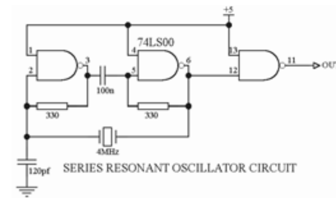
The Threat of Static Power Loss

- If your primary loss mechanism becomes static phenomena, then there will become a point where a cranking the clock could be beneficial!
- Run as fast as possible with the best hardware as possible (32 bit MCU if appropriate vs. 8 bit or something)
- Then sleep! (the static monster won't get you if you're in sleep)
- Not necessarily the right solution, particularly as new transistor models come in and keep static loss at bay, but you never know.

Where do we get frequencies?

16MHz Crystal

- Most frequencies come from Crystal Oscillators made of quartz
- Equivalent to very High-Q LRC tank circuits
- https://en.wikipedia.org/wiki/Crystal_oscillator_frequencies
- Incorporate into circuit like that below and boom, you've got a square wave of some specified frequency dependent largely on the crystal



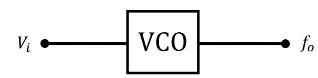
http://www.z80.info/uexosc.htm
https://en.wikipedia.org/wiki/Crystal_oscillator

High Frequencies

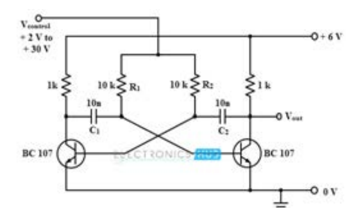
- Very hard to get a crystal oscillator to operate above ~200 MHz (7th harmonic of resonance of crystal itself, which usually is limited to about 30 MHz due to fabrication limitations)
- Where does the 2.33 GHz clock of my iPhone come from then?
- Frequency Multipliers!

Voltage Controlled Oscillator

- It is very easy to make voltage-controlled oscillators that run up to 1GHz or more.
- Why don't we just:



  - Pick the voltage V_i that is needed to get the frequency we want f_o? That's gotta be specified right?
  - Same reason we don't see op amps in open loop out in the wild…they are too unstable…gotta place them in negative feedback

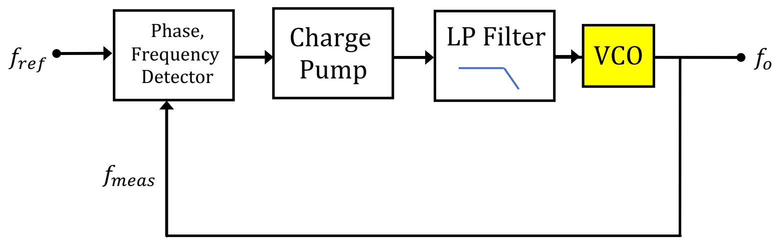


A simple VCO (not found in FPGA)

http://www.electronicshub.org/voltage-controlled-oscillators-vco/

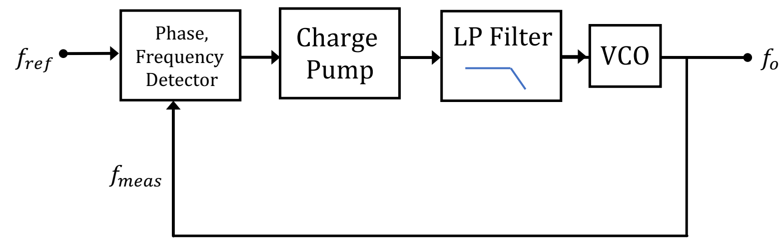
Phase Locked Loop

- Place the unstable, but capable VCO in a feedback loop.
- This type of circuit is a phase-locked loop variant

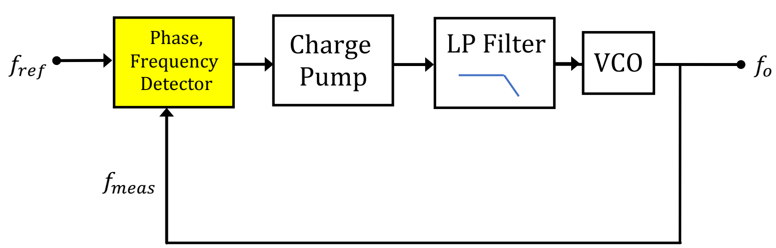


Phase Locked Loop

- Circuit that can track an input phase of a system and reproduce it at the output

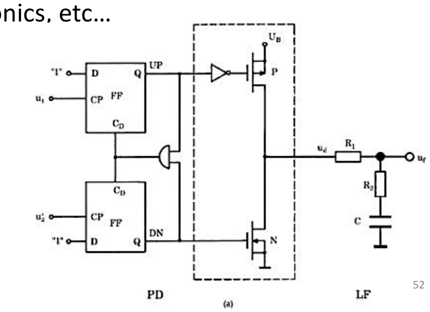


Phase, Frequency Detector



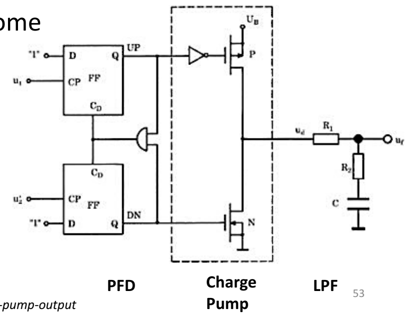
Phase Detector

- Can be a simple XOR gate
- If near the desired frequency already this can work…if it is too far out, it won't and can be very unreliable since phase and frequency are not the same thing, it will lock onto harmonics, etc…
- Instead use a PFD:
  - Phase/Frequency Detector:



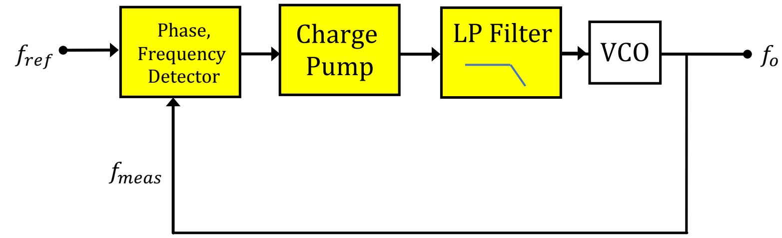
Phase-Frequency Detection

- Detects both change and which clock signal is consistently leading the other one
- Using MOSFETs you charge/discharge a capacitor accordingly which also with some resistors low-pass filter's the signal
- The output voltage is then roughly proportional to the frequency error!



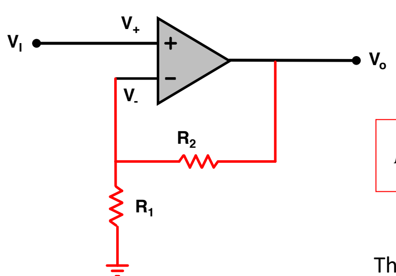
http://www.globalspec.com/reference/72819/203279/2-7-phase-detectors-with-charge-pump-output

PFD, Charge Pump, LP Filter



- So this circuit can make $f_0 = f_{ref}$ That doesn't help us!
- How can we make a higher frequency?

Use Resistors in Voltage Divider in Feedback Path!



- A voltage divider in feedback path gives us voltage gain!

$$K = \frac{1}{1 - p + G} \quad p \approx 0.9999 \text{ means} \quad K = \frac{1}{G}$$

$$G = \frac{R_1}{R_1 + R_2}$$

The gain A_v of this circuit is therefore:

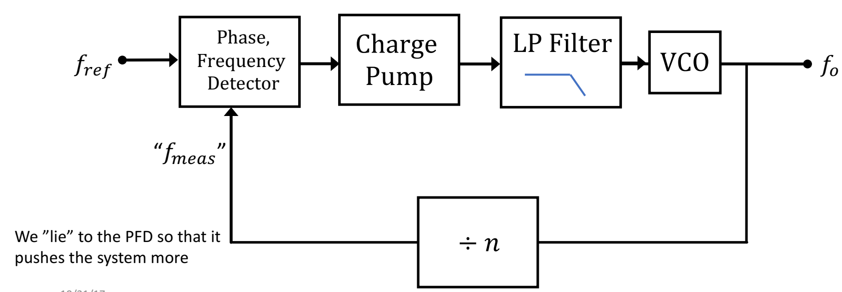
$$A_v = \frac{R_1 + R_2}{R_1}$$

The gain of a "non-inverting amplifier"

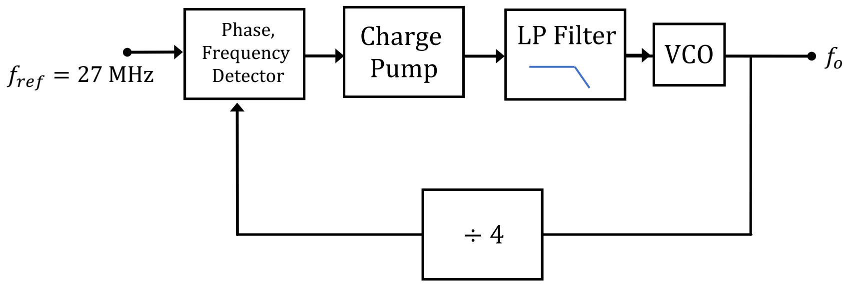
$$V_- = V_o \frac{R_1}{R_1 + R_2}$$

Use a Clock Divider in Feedback Path!

- A clock divider in feedback path gives us clock gain!



Use a Clock Divider in Feedback Path!



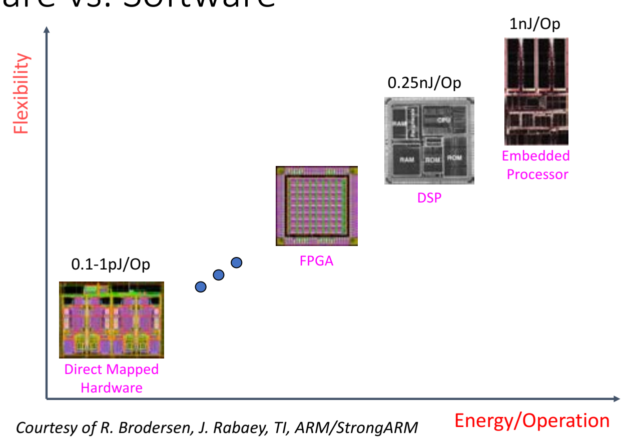
Clock Generation Uses Power!

- In general with everything, if you don't need it, don't use it.
- Human eye can't tell difference between these two dimmers

Running Blue LED at 50% duty cycle at 100Hz
Based off of 12 MHz clock
Consuming 0.35 W

Running Blue LED at 50% duty cycle at 4000Hz
Based off of 480 MHz clock
Consuming 0.5W

Hardware vs. Software



Courtesy of R. Brodersen, J. Rabaey, TI, ARM/StrongARM

Alternative Energy Sources

- Energy Harvesting – movement
- Energy Harvesting – thermoelectric generator
- Ambient RF
- Grapes
- Gastric fluids

Trends: Energy Scavenging

MEMS Generator

Jose Mur Miranda/
Jeff Lang

Vibration-to-Electric Conversion

~ 10mW

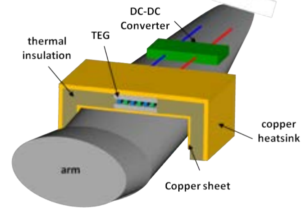
Power Harvesting Shoes

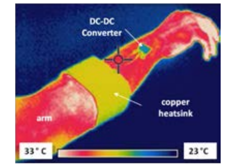
Joe Paradiso
(Media Lab)

After 3-6 steps, it provides 3 mA for 0.5 sec

~10mW

Low-Profile Wearable Body-Powered Thermoelectric Generator

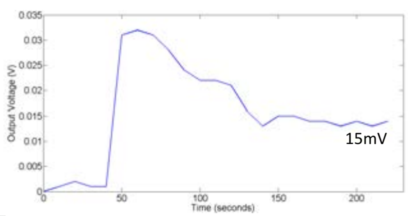




- Low profile, lightweight, conformal.
- Utilization of small temperature difference
- Utilization of natural convection for cooling

Credit: Krishna Settaluri MIT '2010

Experimental Results

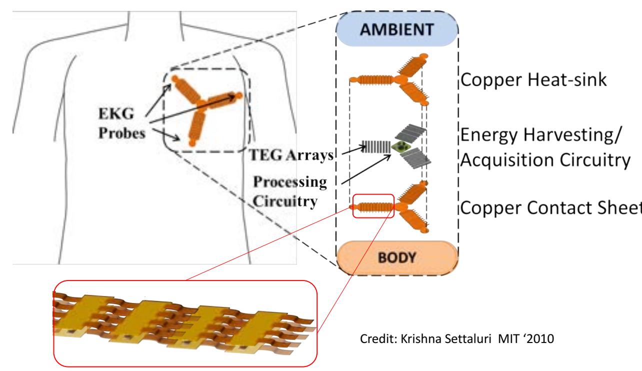


16 TEG Islands (2 TEG modules)

Optimal Electrical Load Resistance	33Ω (20Ω theoretical)
Optimized Power	11μW

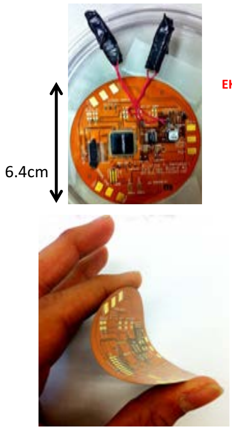
Credit: Krishna Settaluri MIT '2010

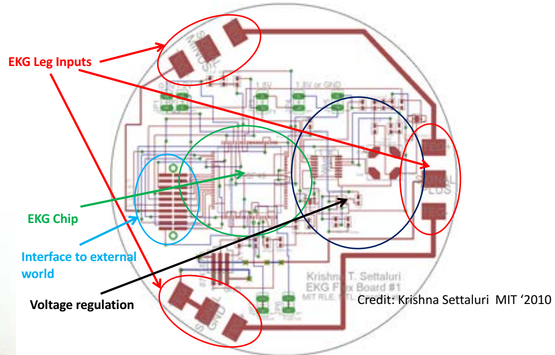
Body-Powered, Flex EKG System



Credit: Krishna Settaluri MIT '2010

Board Layout





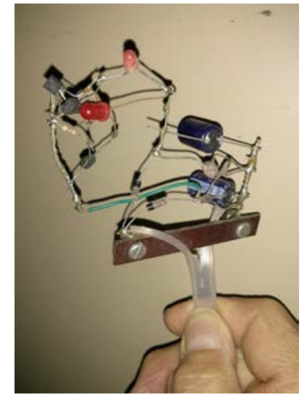
- Thin flex substrate – 0.0074 in ~ 0.02cm

Ambient RF

Prudential Center
FM Stations:
WZLX 100.7, WBMX 104.1, WMJX 106.7, and WXKS-FM 107.9, WBOS 92.9, WBQT 96.9, and WROR-FM 105.7.

Power output:
22,000 watts

Recovered:
~ 0.2 milliwatt



Grape Power

Grape Juice Voltage

Copper penny
Zinc screw

Newman's Own
Grape Juice

Inifinite Power! If we ignore

Gastric Fluid Powered

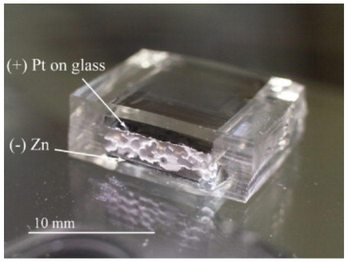


Fig. 3. Photo of GMB prototype.

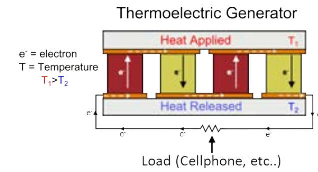
Hikaru Jimbo, Norihisa Miki
Gastric-fluid-utilizing micro battery for micro medical devices
Sensors and Actuators B: Chemical, Volume 134, Issue 1, 2008, 219–224
http://dx.doi.org/10.1016/j.snb.2008.04.049

Research conducted at MIT: Phil Nadeau

Using 4mm x 4mm electrodes (Zn/Cu), 10uW of average power from in brief (< 30 min) in vivo measurements

Energy harvesting

- Thermo-electric generator
  - Thermoelectric material converts temperature difference into voltage



40 K temp difference
1.8 V @ 368 mA

https://www.adafruit.com/products/700
http://electronicdesign.com/content/content/73937/73937-fig2.gif

TEXAS INSTRUMENTS bq25504
SLUSAH0C – OCTOBER 2011 – REVISED JUNE 2015

bq25504 Ultra Low-Power Boost Converter With Battery Management for Energy Harvester Applications

1 Features

- Ultra Low-Power With High-Efficiency DC-DC Boost Converter/Charger
  - Continuous Energy Harvesting From Low-Input Sources: $V_{IN} \geq 80$ mV (Typical)
  - Ultra-Low Quiescent Current: $I_Q < 330$ nA (Typical)
  - Cold-Start Voltage: $V_{IN} \geq 330$ mV (Typical)
- Programmable Dynamic Maximum Power Point Tracking (MPPT)
  - Integrated Dynamic Maximum Power Point Tracking for Optimal Energy Extraction From a Variety of Energy Generation Sources
  - Input Voltage Regulation Prevents Collapsing Input Source
- Energy Storage
  - Energy Can be Stored to Rechargeable Li-ion Batteries, Thin-film Batteries, Super-Capacitors, or Conventional Capacitors
- Battery Charging and Protection
  - User Programmable Undervoltage and Overvoltage Levels
  - On-Chip Temperature Sensor With Programmable Overtemperature Shutoff
- Battery Status Output
  - Battery Good Output Pin

3 Description

The bq25504 device is the first of a new family of intelligent integrated energy harvesting nano-power management solutions that are well suited for meeting the special needs of ultra low power applications. The device is specifically designed to efficiently acquire and manage the microwatts (µW) to miliwatts (mW) of power generated from a variety of DC sources like photovoltaic (solar) or thermal electric generators. The bq25504 is the first device of its kind to implement a highly efficient boost converter/charger targeted toward products and systems, such as wireless sensor networks (WSNs) which have stringent power and operational demands. The design of the bq25504 starts with a DC-DC boost converter/charger that requires only microwatts of power to begin operating.

Once started, the boost converter/charger can effectively extract power from low-voltage output harvesters such as thermoelectric generators (TEGs) or single- or dual-cell solar panels. The boost converter can be started with V_{IN} as low as 330 mV, and once started, can continue to harvest energy down to V_{IN} = 80 mV.

Device Information(1)

PART NUMBER	PACKAGE	BODY SIZE (NOM)
bq25504	VQFN (16)	3.00 mm x 3.00 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

FFTs

Audio Feature Extraction

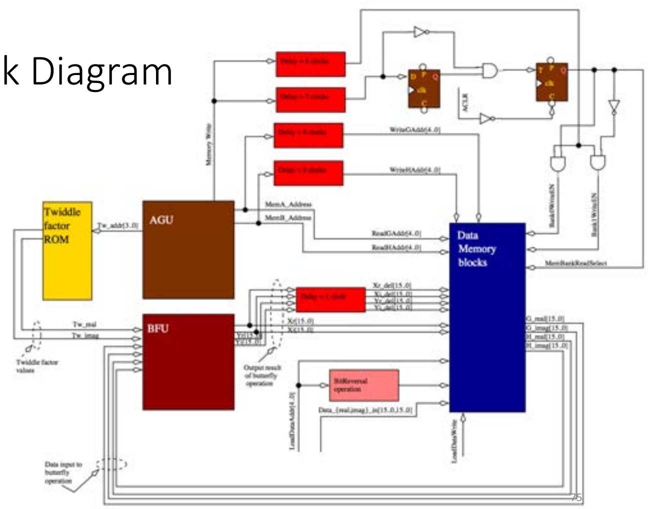
- Most features are best recognized in the frequency domain
- Use Discrete Fourier Transform
  - Algorithm used: Fast Fourier Transform (FFT)
  - Input: N data values acquired at sample frequency ω_S
    - Nyquist rate is $\omega_S/2$
  - Output: N complex values representing DFT coefficients in the frequency range $-\omega_S/2$ to $+\omega_S/2$.
    - Each value covers a frequency range of ω_S/N
    - Indices (0,(N/2)-1) are for frequencies $i*(\omega_S/N)$
    - Indices (N/2,N-1) are for frequencies $-\omega_S/2 + (i - N/2)*(\omega_S/N)$
  - If N is even, output is symmetric, so we can calculate magnitude using only positive frequencies. Magnitude ≈ $\sqrt{r^2+i^2}$ * constant factors.
- Example
  - Audio data from AC97 sampled at 8kHz
  - 2048 data points => 2048-point FFT
  - 2048 complex results, each result covers 8k/2048 = 4Hz range

Fast Fourier Transforms

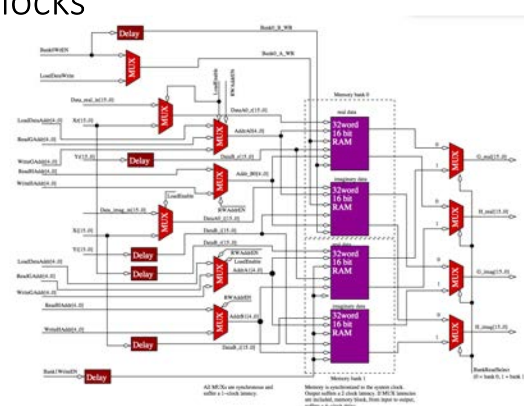
- FFTs are really central to a lot of DSP
- Software FFTs are *relatively* easy to do
- Implementing one in an FPGA from the ground up is a bit less intuitive, but it can be done, and since it can do much of its operations in parallel it has a niche in a lot of real-time applications
- Great review article below (sort of step-by-step build) which should be accessible if you've seen/done/thought about implementing FFTs before in something like C for example…will post on Course site

Slade, George. (2013). The Fast Fourier Transform in Hardware: A Tutorial Based on an FPGA Implementation. .

Full FFT block Diagram



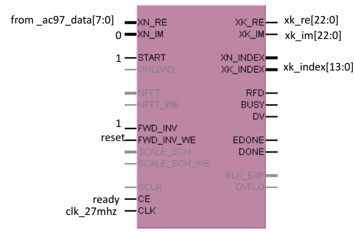
Data Memory Blocks



FFT example - Labkit

Might not be accessible anymore in current ISE!!

- IP wizard will build a N-point FFT module
  - WARNING: they're big!
- In theory, there are two operating modes (select at build time)
  - "pipelined" where you get a complex value out for every sample you send the module – runs continuously
  - "burst" where you load up N samples, wait a while and get your answer while loading the set of samples.
- To use FFT, use sample Verilog
- Demo: audio spectrum analyzer
  - Uses "pipelined" mode
- 44 page datasheet



FFT – Nexys4 DDR

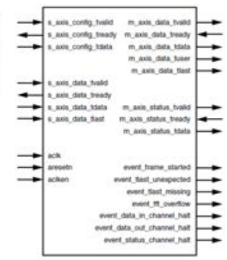
- IP core uses AXI4 protocol
- 97 page datasheet

Table 3-6: **Data Input Channel TDATA Fields**

Field Name	Width	Padded	Description
XN_RE	b_{xn}	Yes	Real component (b_{xn} = 8 - 34) in twos complement or single precision floating-point format.
XN_IM	b_{xn}	Yes	Imaginary component (b_{xn} = 8 - 34) in twos complement or single precision floating-point format.

Table 3-9: **Data Output Channel TDATA Fields**

Field Name	Width	Padded	Description
XK_RE	b_{xk}	Yes - sign extended	Output data: Real component in twos complement or floating-point format. (For scaled arithmetic and block floating-point arithmetic, b_{xk} = b_{xn}. For unscaled arithmetic, b_{xk} = b_{xn} + $\log_2$ (maximum point size) +1. For single precision floating-point b_{xk} = 32).
XK_IM	b_{xk}	Yes - sign extended	Output data: Imaginary component in twos complement or single precision floating-point format. (For scaled arithmetic and block floating-point arithmetic, b_{xk} = b_{xn}. For unscaled arithmetic, b_{xk} = b_{xn}+ $\log_2$ (maximum point size) +1. For single precision floating-point b_{xk} = 32)



AXI Protocol

- Separate data and address connections for reads and writes: simultaneous, bidirectional data transfer.
- Useful for memory mapped applications

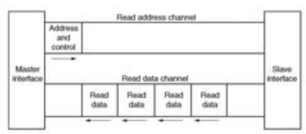


Figure 1-1: **Channel Architecture of Reads**

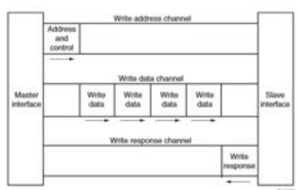


Figure 1-2: **Channel Architecture of Writes**

FFT of AC97 data

To process AC97 samples:

- use Pipelined mode (input one sample in each cycle, get one sample out each cycle).
  - FFT expects one sample each cycle, so hook READY to CE so that FFT only cycles once per AC97 frame
- use Unscaled mode, do scaling yourself
  - Number of output bits = (input width) + NFFT + 1

  - NFFT is $\log_2$(size of FFT)
- let number of FFT points = P, assume 48kHz sample rate
  - there are P frequency bins
  - positive freqs in bins 0 to (P/2 – 1)
  - negative freqs in bins (P/2) to (P-1)
  - each bin covers (48k/P)Hz
  - Use XK_INDEX to tell which bin's data you're getting out
  - Typically you want magnitude = sqrt(xk_re^2 + xk_im^2)

Tools

- Labkit hardware with sample Verilog
  - NTSC Camera – display BW images
  - ZBT Memory – high speed memory two 512Kx36 banks
  - Alphanumeric data with hex display
  - Compact Flash – 128Mbits non-volatile memory
- Nexys4 hardware with sample Verilog
  - VGA Camera
  - SD card read/write
- Application support
  - Sound -Matlab script: convert wav files to AC97 8bit COE file Images - Matlab script: convert BMP COE field
  - USB PC-Labkit data transfer
- git – Shared project team repository with version control
- hg – Shared project team repository with version control for people who want to be different

Iterative SQRT module

```
// takes integer square root iteratively
module sqrt #(parameter NBITS = 8,  // max 32
                        MBITS = (NBITS+1)/2)
            (input wire clk,start,
             input wire [NBITS-1:0] data,
             output reg [MBITS-1:0] answer,
             output wire done);
  reg busy;
  reg [4:0] bit;
  // compute answer bit-by-bit, starting at MSB
  wire [MBITS-1:0] trial = answer | (1 << bit);

  always @(posedge clk) begin
    if (busy) begin
      if (bit == 0) busy <= 0;
      else bit <= bit - 1;
      if (trial*trial <= data) answer <= trial;
    end
    else if (start) begin
      busy <= 1;
      answer <= 0;
      bit <= MBITS - 1;
    end
  end

  assign done = ~busy;
endmodule
```

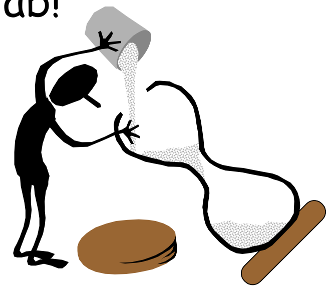

Special Sessions

- Thu: tutorials (in lab; optional)
  - labkit NTSC camera 2:30p Gim
  - labkit flash memory 2:45p Gim
  - using images and COE files 3pm Gim
  - chroma keying 8p Diana
  - PC interfacing 9p Diana
  - Nexys4 camera (8:30p) Weston
  - Device Interfacing (3:15p) Joe
- Other sources of information:
  - general computer vision and image processing ideas James
  - XADC, Vivado block designs, or ILA - Mitchell, Joe
  - motors, and in particular servos – Elizabeth
  - FIR filters / generated coefficients and “tested” the filters - Madeleine

How to Make Your Projects Work

- What are the power requirements?
  - Labkit: 3.3V, 5V, +12, -12
  - Nexys4: 3.3V
- Characterize external components before designing your system
  - Understand input/output voltage specs
  - Understand behavior of unused input/control lines
  - Understand tri-state control lines
- Synchronize external signals to system clock.
- Exercise with care: grounds – in particular high current devices
- Look at waveforms on a scope for external signals >1Mhz
- Do NOT assume plug/play except for speakers, microphones, NTSC camera
- Verilog modules except for Labs 3-5 are provided “As-is”. No warranty expressed or implied.

See you in lab!



Final project represents 72 hours of credit, so you should average 2-3 hours/day of work on your project assuming you give yourself the occasional day off…