inst.eecs.berkeley.edu/~eecs151

EECS151 : Introduction to Digital Design and ICs

Lecture 26 – Flash, Parallelism

Bora Nikolić

Google's Tensor Inside of Pixel 6, Pixel 6 Pro: A Look into Performance and Efficiency

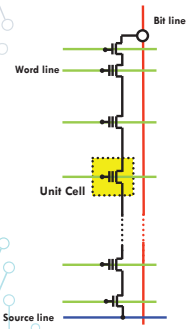
AnandTech, Nov. 2...One of the biggest changes, and most interesting to our readers, is the fact that the Pixel 6 and Pixel 6 Pro come powered on by Google's own "Tensor" SoC. And it's here where there's quite a bit of confusion as to what exactly the Tensor is. Google explains that the Tensor is Google's start in a journey towards the quest of enabling new kinds of workloads, which in the company's words, were simply not possible or achievable with "standard" merchant silicon solutions. Taking advantage of Google research's years of machine learning experience, it's a chip that's heavily focused towards ML as its primary differentiating feature, and what is said to allow the Pixel 6 phones to have many of the new unique feature exclusive to them...

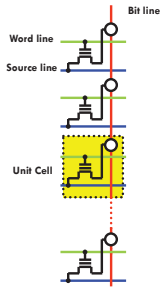
Review

- Multiple cache levels make memory appear both fast and big
- Direct mapped and set-associative cache
- Memory compilers generate SRAM blocks
- Several options for memory on FPGAs: Distributed, BlockRAM, UltraRAM
- Many more bits stored in DRAM and Flash
- Flash
  - Single-level vs multi-level
  - Read and Write Flash Cell
  - NAND vs NOR

Flash Organization

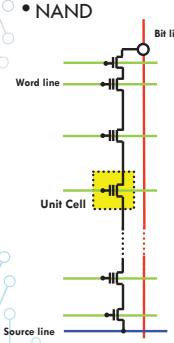
NAND vs NOR Flash





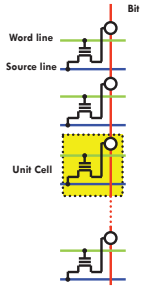
NAND vs NOR Flash

- NAND



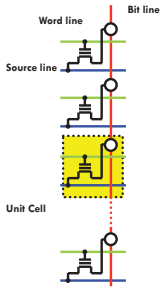
NAND:
- High Density
- Used for data storage
  - USB drives
  - Memory cards
  - SSD

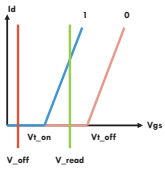
- NOR



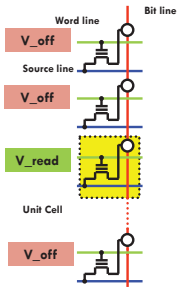
NOR:
- Lower Latency
- Used for code storage
  - Embedded systems

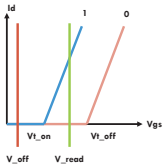
NOR Flash Read



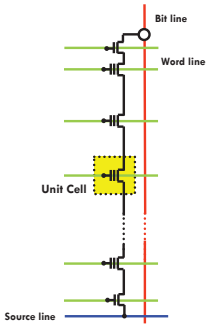


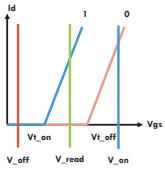
NOR Flash Read



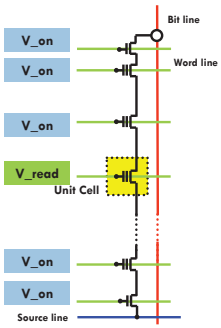


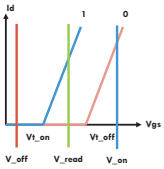
NAND Flash Read





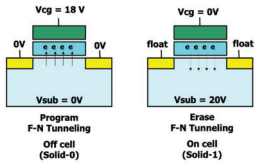
NAND Flash Read

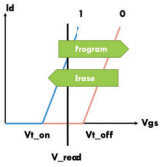




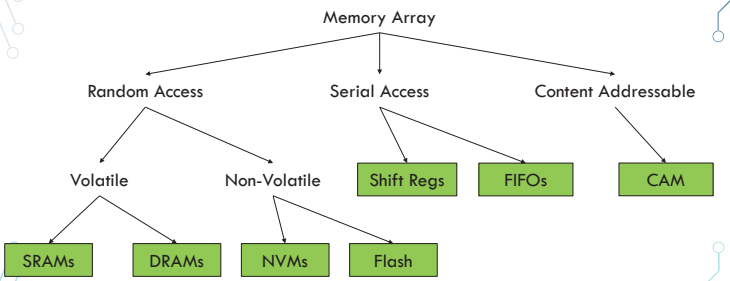
Flash Write

- Step 1: Erasing.
  - Erase all the FG transistors to set them to 1
  - Apply a negative voltage to the gate -> Electrons flow from the floating gate to the substrate.
- Step 2: Programming
  - Reprogram the appropriate FG transistors to set them to 0
  - Apply a high voltage to the gate -> Electrons are tunneled onto the floating gate.





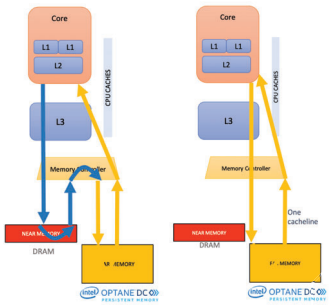
Memory Overview



Other Memories

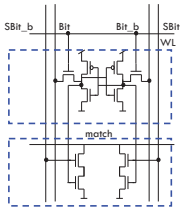
Non-Volatile Main Memory

- Intel Optane DC Persistent Memory
- Non-Volatile
- Storage based on resistance:
  - High resistivity : 0
  - Low Resistivity: 1
- High capacity:
  - 128, 256, 512 GB
- Modes:
  - Memory Mode
  - App-directed Mode



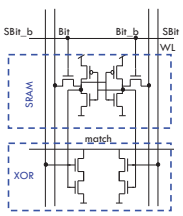
Content Addressable Memory

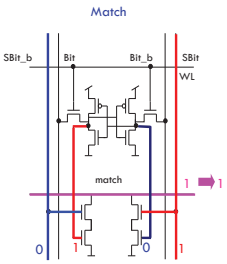
- Commonly used in translation lookaside buffers (TLBs).
- Matching asserts a matchline output for each world that contains a specified key

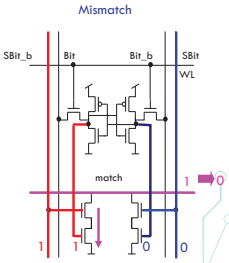


Content Addressable Memory

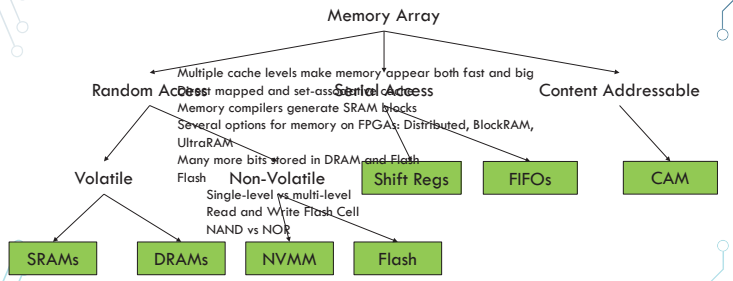
- Commonly used in translation lookaside buffers (TLBs).
- Matching asserts a matchline output for each world that contains a specified key







Memory Overview



Administrivia

- Special lecture on Monday, Dec 6. at 11:00am
  - FPGA prototyping
  - Not on final, but very useful
- Project, project, project !
  - Final checkoffs on Dec. 7
- Homework 11 due Dec 3.
- Last class on Wednesday!
- Final is on December 13, 11:30-2:30

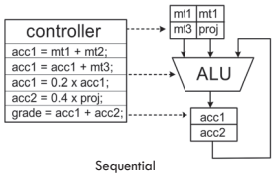
Parallelism

Parallelism

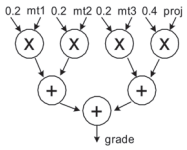
- Doing more than one thing at a time.
- Optimization in hardware design often involves using parallelism to trade off cost and performance.
- Can also be used to improve energy efficiency.

```
float mt1, mt2, mt3, project;

grade = 0.2 × (mt1 + mt2 + mt3)
      + 0.4 × project;
```



Sequential



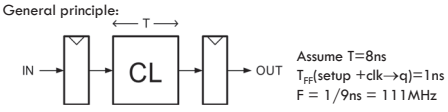
Parallel

Types of Parallelism in Hardware

- Parallelism in Hardware
  - Pipelining
  - SIMD Processing
  - Multithreading

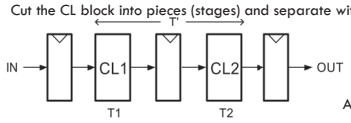
Pipelining

- General principle:



Assume T=8ns
T_{FF}(setup +clk→q)=1ns
F = 1/9ns = 111MHz

- Cut the CL block into pieces (stages) and separate with registers:



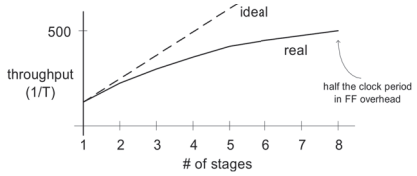
Assume T1 = T2 = 4ns

T' = 4ns + 1ns + 4ns +1ns = 10ns
F = 1/(4ns +1ns) = 200MHz

- CL block produces a new result every 5ns instead of every 9ns.

Limits of Pipelining

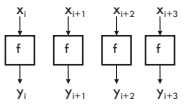
- Without FF overhead, throughput improvement ∝ # of stages.
- After many stages are added FF overhead begins to dominate:



- Other limiters to effective pipelining:
  - clock skew contributes to clock overhead
  - imballanced stages
  - FFs dominate *cost*
  - clock distribution power consumption

SIMD Parallelism

- Make multiple instances of the loop execution data-path and run them in parallel, sharing the some controller.
- Example: $y_i = f(x_i)$



Usually called SIMD parallelism. Single Instruction Multiple Data

- Example: vector processors.

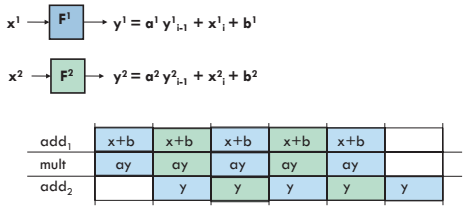
Multithreading

- Fill in "holes" in the pipeline with another (independent) computation

add_1	x+b		x+b		x+b	
mult	ay		ay		ay	
add_2		y		y		y

Multithreading

- Fill in "holes" in the pipeline with another (independent) computation.



- Example: multi-threading in microprocessors

Low Power Design

Review: Sources of Power Dissipation

- $P_{total} = P_{dynamic} + P_{static}$
- Dynamic power: $P_{dynamic} = P_{switching} + P_{shortcircuit}$
  - Switching load capacitances
  - Short-circuit current
- Static power: $P_{static} = I_{sub} V_{DD}$
  - Mostly subthreshold leakage

To Lower Power

- Dynamic power: $P = \alpha C V_{DD}^2 f$
- To lower power:
  - Lower VDD – quadratic!
  - Lower switching activity
  - Lower capacitance
  - Lower frequency (doesn't save energy)
- Static/leakage power: $P = I_{Leak} V_{DD}$
- To lower power:
  - Lower VDD – more than quadratic!
  - Lower I_{Leak}
    - Increasing threshold
    - Stacking devices
    - Turning off parts that are not being used

Lowering Power

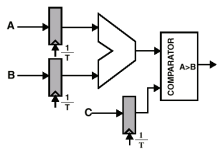
- Design-time techniques: Applied by the designer to lower power (by lowering V_{DD}, C or activity)
- Run-time techniques: Applied during runtime, to save power when lower performance is required (by dynamically lowering V_{DD}, increasing V_{Th} or turning off parts)

Lowering V_{DD}

- $E = \alpha C V_{DD}^2$ - energy drops quadratically
- $f \sim V_{DD}$ - frequency drops linearly
  - $I_{on} \sim V_{DD}$, $f \sim I_{on}$
  - Linear at high supplies, higher penalty when approaching threshold
- Key idea: Use parallelism to increase performance at lower supplies

Fixed-Throughput Design

- Reference design (Chandrakasan and Brodersen, IEEE JSSC'92)

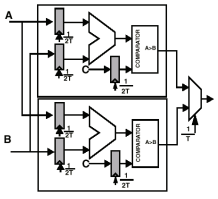


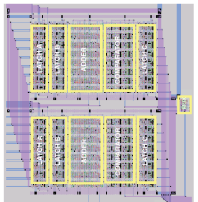
- Critical path delay $\Rightarrow T_{adder} + T_{comparator}$ (= 25ns)
  $\Rightarrow f_{ref} = 40$Mhz
- Total capacitance being switched = C_{ref}
- $V_{dd} = V_{ref} = 5V$
- Power for reference datapath = $P_{ref} = C_{ref} V_{ref}^2 f_{ref}$

from [Chandrakasan92] (*IEEE JSSC*)

Fixed-Throughput Design

- Parallel datapath

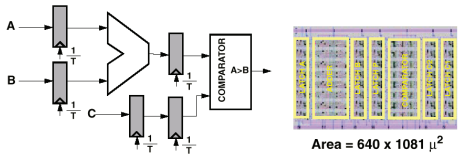




- The clock rate can be reduced by half with the same throughput $\Rightarrow f_{par} = f_{ref} / 2$
- $V_{par} = V_{ref} / 1.7$, $C_{par} = 2.15 C_{ref}$
- $P_{par} = (2.15 C_{ref}) (V_{ref}/1.7)^2 (f_{ref}/2) = 0.36\ P_{ref}$

Fixed-Throughput Design

- Pipelined datapath



Area = 640 x 1081 μ^2

- Critical path delay is less $\Rightarrow$ max [T_{adder} , $T_{comparator}$]
- Keeping clock rate constant: $f_{pipe} = f_{ref}$
  Voltage can be dropped $\Rightarrow V_{pipe} = V_{ref} / 1.7$
- Capacitance slightly higher: $C_{pipe} = 1.15C_{ref}$
- $P_{pipe} = (1.15C_{ref}) (V_{ref}/1.7)^2 f_{ref} \approx 0.39 P_{ref}$

Fixed-Throughput Design

- Summary of techniques

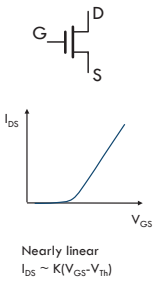
Architecture type	Voltage	Area	Power
Simple datapath (no pipelining or parallelism)	5V	1	1
Pipelined datapath	2.9V	1.3	0.39
Parallel datapath	2.9V	3.4	0.36
Pipeline-Parallel	2.0V	3.7	0.2

Supply Reduction in Modern Technologies

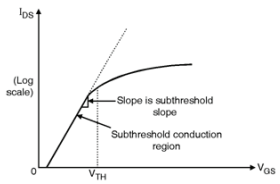
- Assume V_{DD} = 1V, V_{Th} = 0.3V
- Delay roughly doubles at V_{DD} = 0.65V (= V_{ref}/1.53)
- $P_{par} = 2C_{ref} (V_{ref}/1.43)^2 f_{ref}/2 = 0.42P_{ref}$
- Power still lower, but not as effective as before

Leakage Power

Review: Subthreshold Leakage



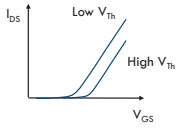
Nearly linear
$I_{DS} \sim K(V_{GS}-V_{Th})$



I_{DS} Vs V_{GS} characteristics in log scale

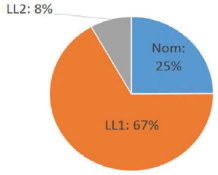
Multiple Thresholds

- Modern technologies have transistors with multiple thresholds
  - Low thresholds are faster (higher Ion), but leak more
  - Typically separated by 80-90mV – one order of magnitude of leakage



Leakage

- Modern example: Intel Xeon Skylake-SP (S. Tam, ISSCC'18)
  - 14nm 28-core datacenter processor
  - Three transistor types
  - Design starts with middle (LL1)
  - Speed critical paths up by adding Nom
  - Save leakage power by adding LL2 to non-critical paths (so they become more crit



Performance: Nom > LL1 > LL2
Leakage: Nom > LL1 > LL2

Summary

- Multiple cache levels make memory appear both fast and big
- Direct mapped and set-associative cache
- Memory compilers generate SRAM blocks
- Several options for memory on FPGAs: Distributed, BlockRAM, UltraRAM
- Many more bits stored in DRAM and Flash
- Flash
  - Single-level vs multi-level
  - Read and Write Flash Cell
  - NAND vs NOR