Project Goals

- Design a simple, but relatively complete, digital logic system in CMOS using a modern approach to logic cell design
- Estimate complexity, area, speed, power for a variety of possible technologies
  - From original to today
  - Including ND 2 micron
- Compare where possible to real implementations
- Serve as a possible basis for a second semester senior design project

Project Approach

- Select some “reference” implementation of some chip
  - Ideally very simple with known characteristics
  - Identify basic library of standard cells
- Estimate how many and where transistors are used in its design
- Project what versions in other technologies might look like: Area, Power, Max clock
  - In variety of technologies (esp. ND 2 micron & today)
- Perform a more detailed design
  - Either Verilog description or standard cell layout
- Analyze that design
- Prepare a report and make a class presentation

Reference Options

- See class website link page for pointers to documentation

Microprocessor	Data Width (bits)
6502	8
1802	8
8048/8051	8
8080	8
PDP-8	12
Simple 12	12
JAM-8	8
NOVA	16
MIPS	32
mini TPU	8

Presentation Outline

- Gather reference implementation characteristics
  - Technology, speed, power, area, transistor count, …
- Overview of instruction set (if processor)
- Overview of microarchitecture and how data flows
- What “off-chip” connections needed (include power, clocks, reset)
- Outline major blocks (and estimate area)
- Transistor estimate for each
  - Where appropriate show transistor diagrams of block
- Project ahead to smaller feature sizes
  - Estimate area, power, speed, …
- Do projection in 2 steps
  - Include stop at 2 microns (ND process)
  - Using Dennard scaling (until 2004 technology)
  - Using constant voltage scaling

Options

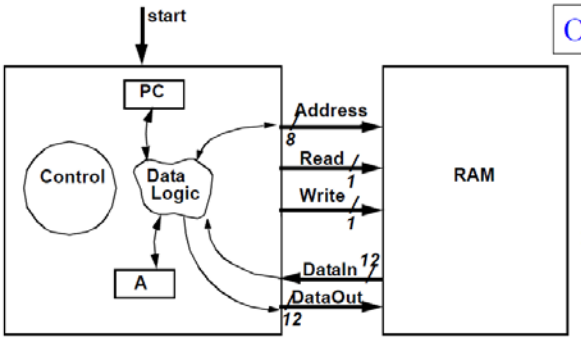
- Design options
  - Verilog of complete processor (dataflow & control)
  - Electric layout of a dataflow slice
  - Electric layout of control logic using standard cell
- One person: one of first two above
- Two persons: Verilog and dataflow slice
- Three persons: All three
- Permitted simplifications (review with instructor)
  - For a microprocessor: Simplified instruction set
  - “Low speed” data flow (i.e. a simple ripple adder)
  - Don’t worry about off-chip pads/drivers

Standardizing Libraries OK

- Class development of std cell & bit slice library encouraged
- Tuesday Oct. 23rd:
  - Bring your understanding of your chip
  - Identify common cells
  - Each student selects subset of cells for implementation
- Later in semester
  - Agree on
    - standard height (for standard cells)
    - standard width (for bit slices)
    - Standard ports
  - Revise Electric implementation & share

Example: Simple 12

Top Level Simple12



Opcode (4) | Address (8)

256 words of 12 bits each.

Addresses all 8 bits wide.

Opcode selects one of 16 instructions from 4 Classes:
- *Jump*: change PC
- *Load/Store*: access memory
- *Operate*: compute new value
- *Reserved:* (for your later enjoyment)

Programmer Visible Registers:

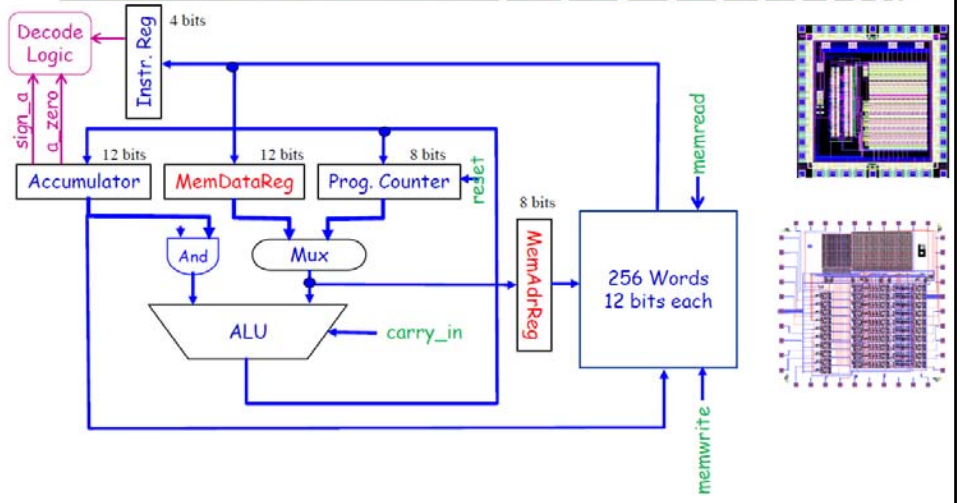
A: Accumulator: - Holds 12 data bits

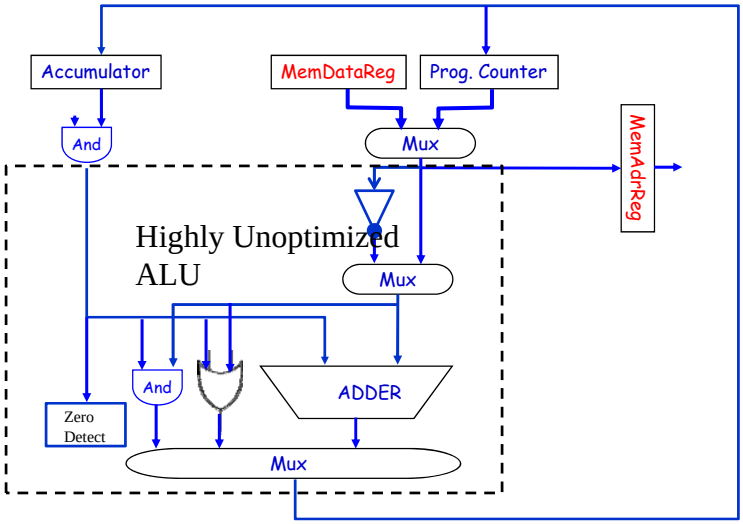
PC: Program Counter- Holds 8 bits of current instruction address

Simple12 ISA

Opcode	Mnemonic	RTL
0000	STOP	Stop execution until reset
0001	JMP X	PC<-X
0010	JN X	if A<0 then PC<-X else PC++
0011	JZ X	if A=0 then PC<-X else PC++
0100	LOAD X	A<-M(X) || PC++
0101	STORE X	M(X)<-A || PC++
0110	reserved	
0111	reserved	
1000	AND X	A<- A and M(X) || PC++
1001	OR X	A<- A or M(X) || PC++
1010	ADD X	A<- A + M(X) || PC++
1011	SUB X	A<- A - M(X) || PC++
1100	reserved	
1101	reserved	
1110	reserved	
1111	reserved	

A Simple Multi-Cycle Data Flow



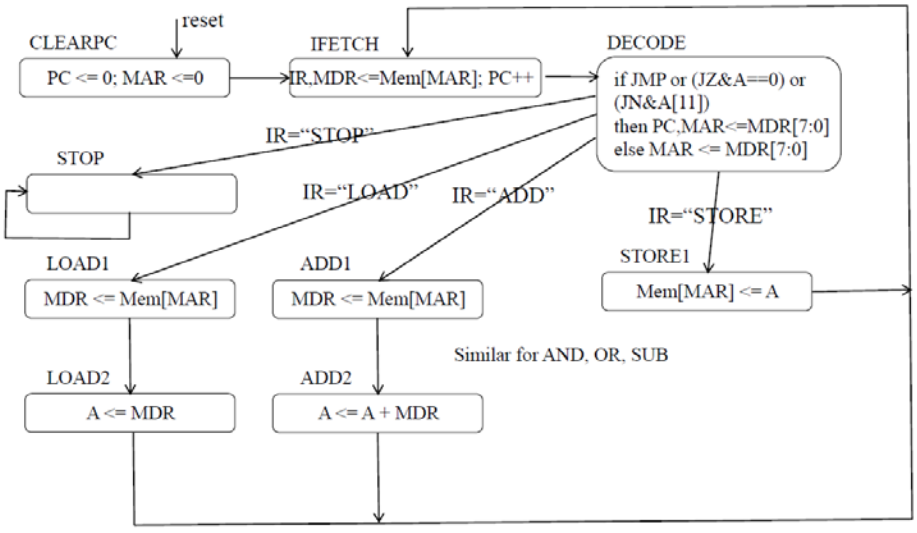


Dataflow that does AND/OR/SUB

Possible Optimizations:

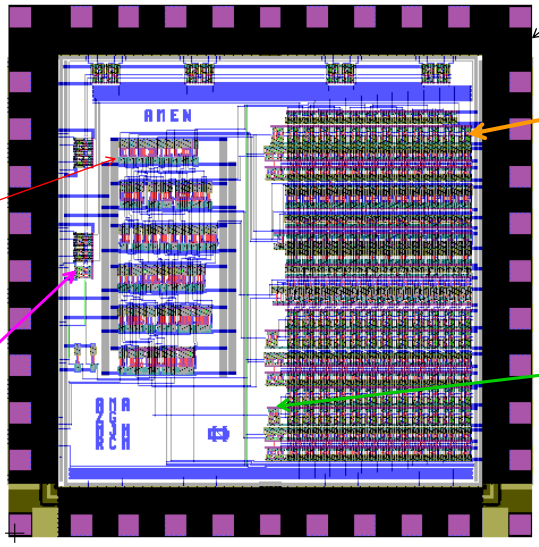
- Combine right side muxes
- AND and OR from Adder for logic functions
- Remove Left-side AND (we have a 4th input on the ALU Mux)

A Matching Execution Cycle



Simple 12

Pad Frame
-standard



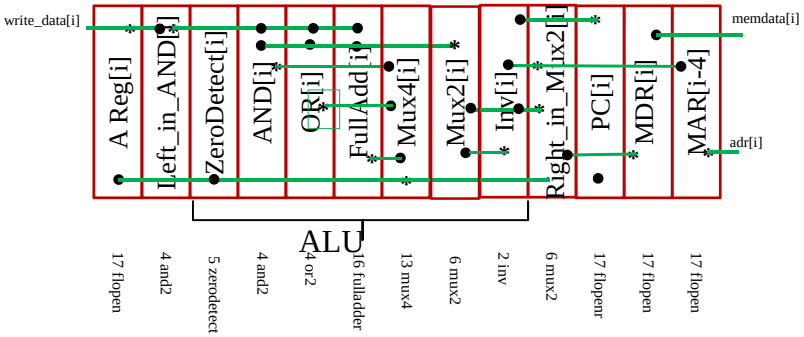
Standard cells
-state machine & timing
-common height
-common power
-ports to wiring bays below

Bit Slices
-data flow
-common width
-common power
-wiring up & down in slice
-wiring horizontal for cross slice

I/O Cells
-standard

Decoder logic
-drive row in dataflow
"random" logic

Simple12 Bit Slice



Tensor Processing Unit

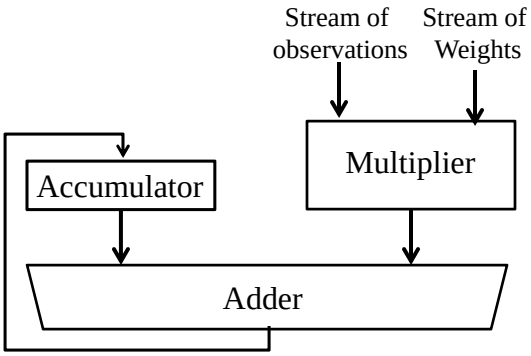
Floating Point

- Goal: represent large numeric range in small # of bits
- Typical scientific number: +/-**1.xxx**$_2$2^e
  - **Exponent e** has some limited range
  - **Mantissa** has some fixed precision (# of bits)
- Today's floating point formats

	Sign bits	Exponent bits	Mantissa bits	Numeric Range
32b	1	8	23	10^{-38} to 10^{+38}
64b	1	11	52	10^{-307} to 10^{+307}
128b	1	15	112	10^{-4914} to 10^{+4914}

- New apps (ML, AI) need *much less* precision
  - but small ints have *insufficient* range
- New ML,AI processors moving to 8 & 16b floating point

Key ML/AI Operation: Vector Inner Product

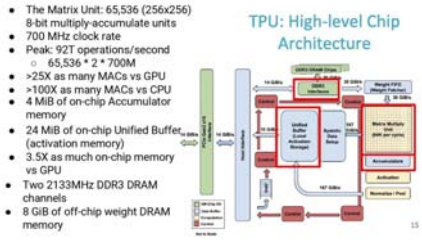


For same stream of observation data, find set of weights that maximizes the inner product.

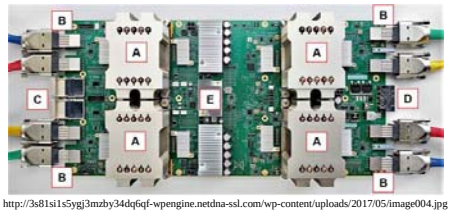
AI Accelerators

First Generation Google **Tensor Processing Unit** Chip:

- H/W Dense Matrix-Vector Product
- Peak 92,000 G flops/s (8 bit floats)



https://images.anandtech.com/doci/11749/hc29.22.730-tensorpu-young-google-page-015.jpg



http://3s81si1s5ygj3mzby34dq6qf-wpengine.netdna-ssl.com/wp-content/uploads/2017/05/image004.jpg

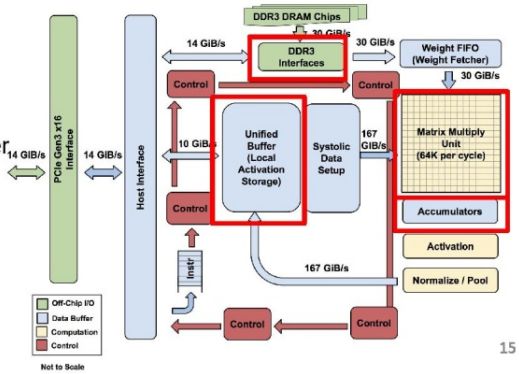
4 TPU2's per card

http://3s81si1s5ygj3mzby34dq6qf-wpengine.netdna-ssl.com/wp-content/uploads/2017/05/image003.jpg

TPU1

- The Matrix Unit: 65,536 (256x256) 8-bit multiply-accumulate units
- 700 MHz clock rate
- Peak: 92T operations/second
  - 65,536 * 2 * 700M
- >25X as many MACs vs GPU
- >100X as many MACs vs CPU
- 4 MiB of on-chip Accumulator memory
- 24 MiB of on-chip Unified Buffer (activation memory)
- 3.5X as much on-chip memory vs GPU
- Two 2133MHz DDR3 DRAM channels
- 8 GiB of off-chip weight DRAM memory

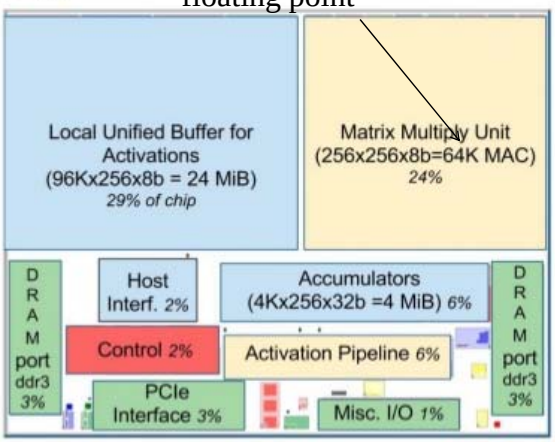
TPU: High-level Chip Architecture



Possible Project:
one of these but in 8-bit floating point

Floorplan of TPU Die

- The Unified Buffer is almost a third of the die
- Matrix Multiply Unit is a quarter
- Control is just 2%



https://image.slidesharecdn.com/in-datacenterperformanceanalysisofatensorprocessingunit-180512155053/95/in-datacenter-performance-analysis-of-a-tensor-processing-unit-15-638.jpg?cb=1526140547

F8: An 8-bit Float Format

S	E3	E2	E1	E0	M2	M1	M0

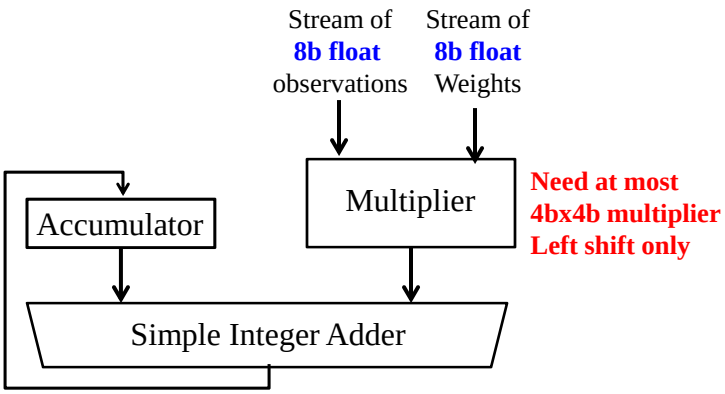
- **S** Sign. 0=>"Positive": 1=>"Negative"
- **E3:E0** Exponent: a 4-bit uint "e"
- **M2:M0**: Mantissa: 3 bit "fraction"
- Special Cases
  - **X0000000** = "Zero"
  - **01111000** = Positive infinity
  - **11111000** = Negative infinity
  - **X1111xyz** (xyz!=000) = NaN i.e. "Not a Number"
- "Denormalized" case: **S0000xyz**: $+/-0.xyz_2 * 2^b$
- Normal Case: **Sefghxyz:** $+/-1.xyz_2 * 2^{e+b-1}$
  - "b" is exponent "bias"
  - Note normalized has a leading "**1**" added

Assuming Bias = 3

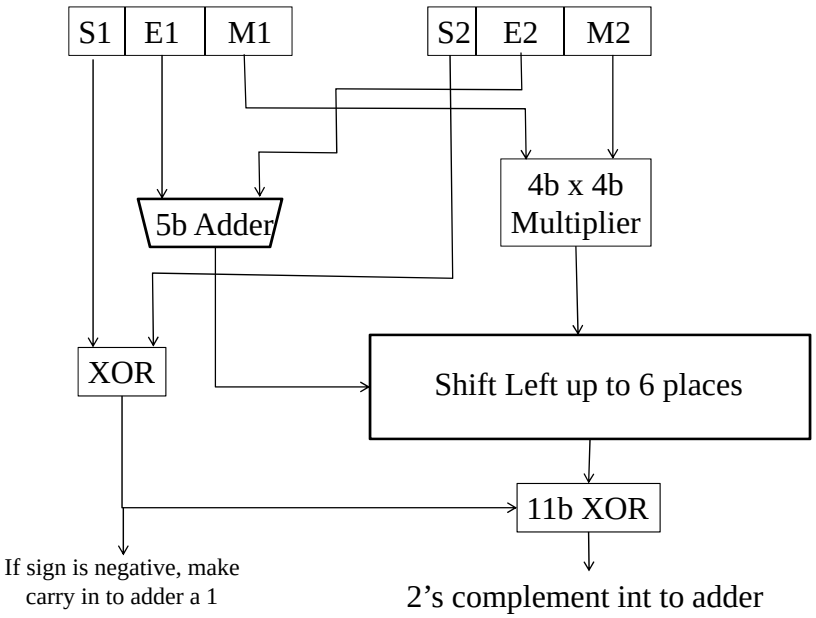
S	E3	E2	E1	E0	M2	M1	M0	Value (All values are Ints)
X	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	$+0.001_2 * 2^3 = 1$ (smallest + non-zero)
1	0	0	0	0	0	0	1	$-0.001_2 * 2^3 = -1$ (smallest - non-zero)
0	0	0	0	0	1	1	1	$+0.111_2 * 2^3 = 7$
S	0	0	0	1	0	0	0	$+/- 1.000_2 * 2^{1+3-1} = 8$ (least normalized)
S	0	0	0	1	0	0	1	$+/- 1.001_2 * 2^{1+3-1} = 9$
S	1	1	1	0	1	1	1	$+/- 1.111_2 * 2^{14+3-1} = 15 * 2^{13} = 122,280$ (largest normalized)
0	1	1	1	1	0	0	0	+∞
1	1	1	1	1	0	0	0	-∞
X	1	1	1	1	0	0	1	NaN

Other biases provide rational numbers at reduced range
E.g. if bias = -14, range is from 0.000053 to 1.875

Advantage of This Bias



Notional F8 MAC



Possible Group Project

- Extend ISA of simple core with F8 MAC instructions
- Develop sample code to do simple AI/ML operations
- Develop Verilog design of:
  - Core
  - F8 Accelerator
  - and combine
- Develop layout (in Electric) of
  - Core
  - F8 Accelerator
  - and combine
- Size and project