Introduction of Processor Design for AI Applications

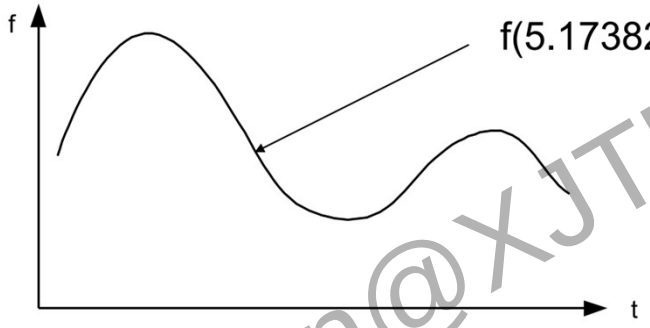
L02 – Data Streaming Applications & Various Architectures

Pengju Ren
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University

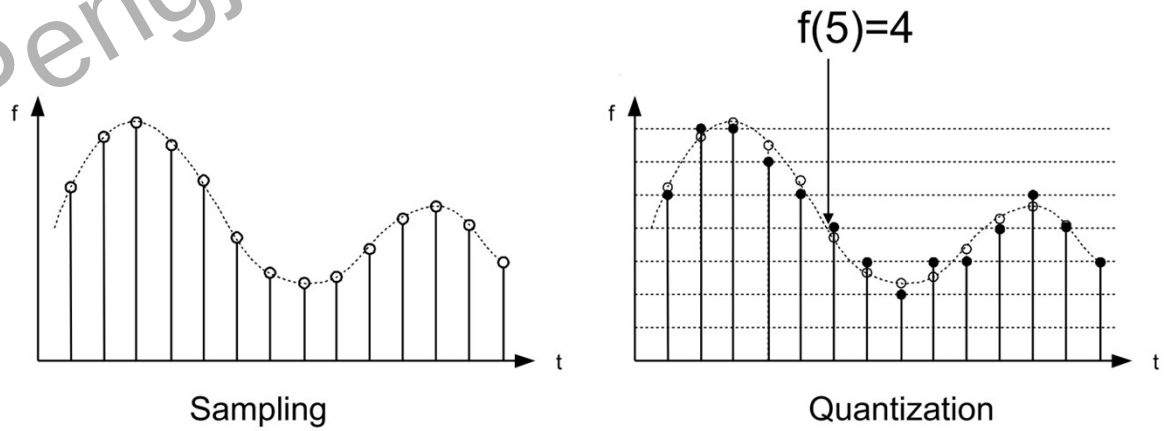
http://gr.xjtu.edu.cn/web/pengjuren

Analog Signal and Digital Signal

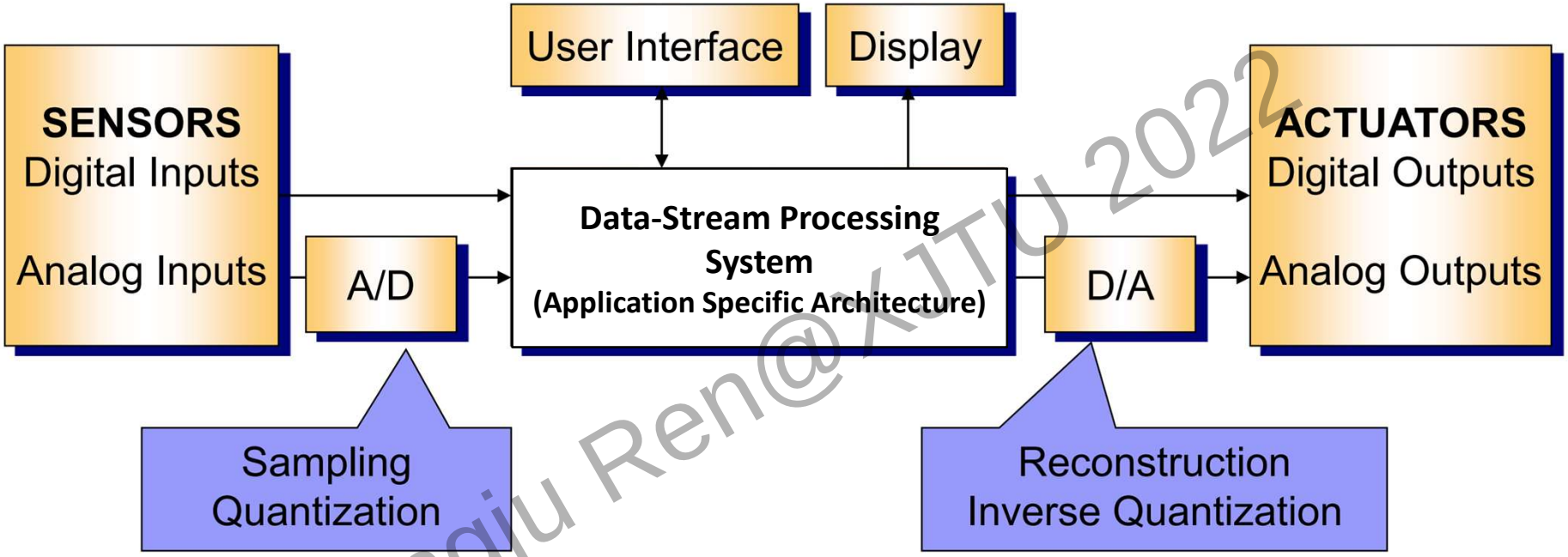
Real world signal
Infinite accuracy on time and magnitude



Get after sampling and quantization
Finite accuracy on time and magnitude



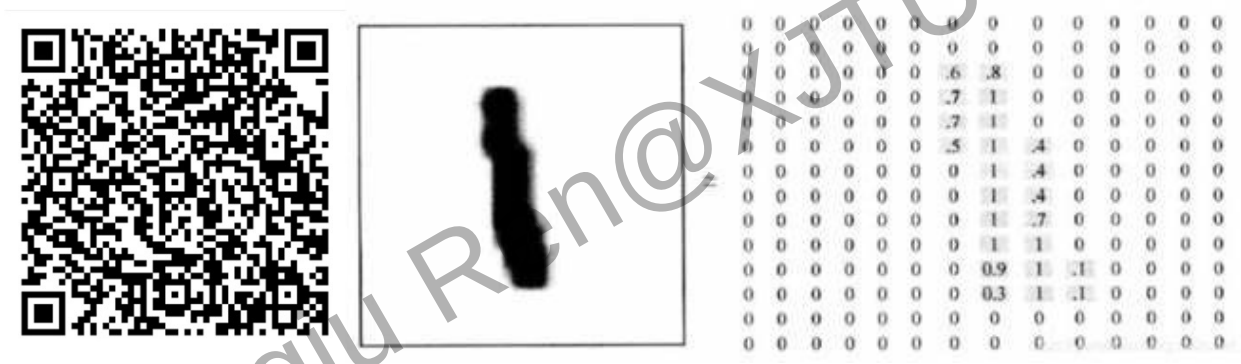
Typical Data-Stream Processing Systems



- Real-time throughput requirement
- Data-driven property
- Non-terminating program

Some Definitions

Signal: "a detectable physical quantity or impulse (such as a voltage, current, or magnetic field strength) by which messages or information can be transmitted"



Signal Processing: " to subject to examination or analysis"

Example: Detection, Alignment

Data streaming Algorithms

Algorithm: "a step-by-step procedure for solving a problem or accomplishing some end"

- Data streaming Algorithms (e.g. *DFT, DCT, DNN, Compression, Encryption, NLP*)
  - Numerically intensive
    - Number representation
    - Bit-width
  - Limited or no control-flow (branching)
  - Streaming/Samples:
    - From ADC, File storage, Camera …
    - Rate often determined by physical factors

Example: Analysis (Filtering: FIR, IIR, Transforms: DFT, DCT)

Comparison (Image recognition, Speech recognition)

Architecture

Architecture: "The manner in which the components of a computer or computer system are organized and integrated"

- Fully custom hardware
  - Difficult to design, best performance
- Semi-custom hardware
  - Use standard modules (Mux, Multiplier)
- Programmable hardware
  - FPGAs
- Domain specific programmable
  - DSPs, GPUs
- General purpose programmable
  - CPUs

Mapping

Sorry … No relevant dictionary definition

1. Understand constraints of algorithm and architecture
2. Specify which elements of algorithm will be implemented by which elements of architecture
3. Measure the effectiveness and usefulness of the mapping
4. Modify algorithm or architecture with the aim of improving some metrics
5. Repeat from Step 1 …

Approach

- Hardware accelerators: custom modules for specific functions
- Code initially targets pure software on general purpose processor
  - Profile and identify bottlenecks
  - Justify use of custom hardware
- Bus interfacing and memory interface design important factors

Learning Objectives

- Describe characteristics of computationally intensive algorithms
- Identify bottlenecks or critical computations/communication
- Map an algorithm onto a given architecture and measure metrics of quality
  - Identify modifications to the **algorithm** to improve quality
  - Identify modifications to the **architecture** to improve quality

Pre-requires

- Digital Signal Processing fundamentals and definitions
  - Filters, Transforms bare minimum
  - Compression, recognition etc. *helpful but not essential*
- Hardware design fundamentals
  - Timing: critical path analysis, maximum operating frequency
  - Power dissipation sources: Capacitance and effect of technology scaling *necessary*
- Programming
  - C/C++ (no need for advanced OOP or classes)
  - Verilog and hardware design: to understand implementation aspects

Review of Digital Circuit

- Gates
- Combinational Logic
- State or Memory
- Sequential logic（systems）
- Finite State Machines (FSMs) ***core concept***
  - Programmable FSM

Timing and Power of Digital Circuit

- Timing:
  - C: low => fast
  - C: high=> slow
  - V: low => slow
  - V: high=> fast
- Power:
  - C: low => low
  - C: high=> high
  - V: low => very low
  - V: high=> very high

Characteristics of Data Stream Algorithms

Metrics

- Estimate compute **requirements** for some problems
- Estimate compute **capabilities** of some architectures

Metrics

- Data transfer / communication
  - #bits transferred over a wire/bus per unit time: Mbps, Gbps
- Common operations
  - Multiply-accumulate: MAC – MMAC/s, GMAC/s, TMAC/s
  - Floating point operations per second: FLOPS — MFLOPS, GFLOPS, TFLOPS
  - Instructions: MIPS (Million Instructions Per Second)
  - Non-standard “operations”: OPS(Ops/second)
- Memory bandwidth
  - Number of bytes/words transferred per unit time: MB/s, GBps
  - Transfers/second for wide buses: MT/s, GT/s

Example: Audio Processing

- Input: audio sampled at 48 KSps, 16-bit/sample
- Operation: apply 30-tap FIR filter
- Estimate:
  - Bit-rate
  - Multiply-Accumulate rate
  - Memory access rate

Example: Image recognition —— AlexNet

- Layers:

  224x224x3 input -> (48+48) kernels of 11x11x3 stride 4: **55x55x48x2**

  -> (128+128) kernels of 5x5x48, max-pool stride 2: **27x27x128x2**

  -> (192+192) kernels of 3x3x256, max-pool stride 2: **13x13x192x2**

  -> (192+192) kernels of 3x3x192: **13x13x192x2**

  -> (128+128) kernels of 3x3x192: **13x13x128x2**

  -> (2048+2048) FC

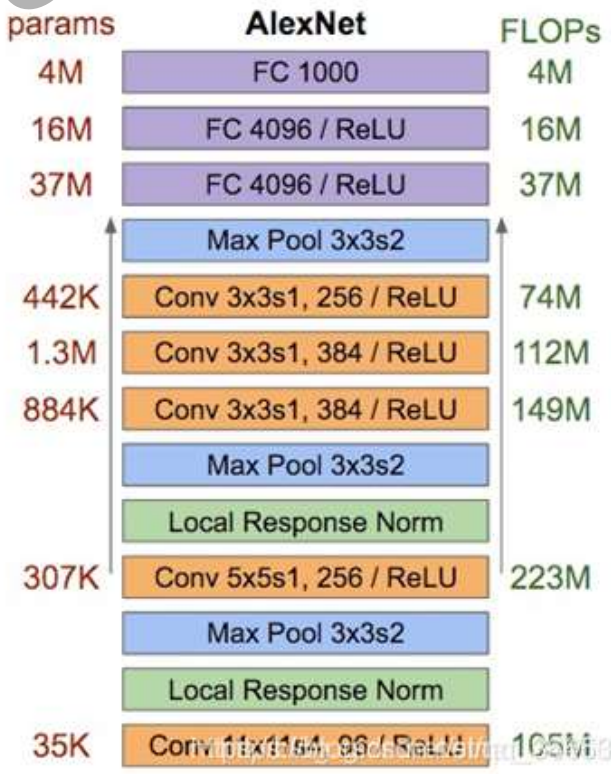
  -> (2048+2048) FC

  -> 1000 FC

- Estimate

  #N of parameters **23MB**

  MFLOPs **727MFLOPs**



Architecture components (1)

- Computational units
  - ALU: Arithmetic and Logic Unit
  - Integer operations always present
  - Floating point co-processors in most modern systems
  - Not in micro-controllers, low-power devices
  - Specialized MAC units: multiply-accumulate
  - Hardware DSP slices: customizable MAC units
- Clock frequency
  - Determined by complexity of hardware
    - **Critical path**: longest combinational logic path
  - Directly proportional to power consumed: more toggling of signals
  - Over-clocking: requires higher voltage operation
    - Compounded effect on power consumption
- Bus bandwidth *(Next Slice)*

Architecture components (2)

- Bus bandwidth
  - Data transfer usually done with multiple wires in sync to form parallel buses
  - Note: off-chip transfer often better using serial buses:
    - Syncing multiple wires to same clock over long distances hard
  - Off-chip: capacitances >> on chip caps
    - Delay and Power increase
  - Wide buses
    - More bits/second
    - Harder to sync wide bus, but throughput improvement worth it

Architecture Example：Arduino

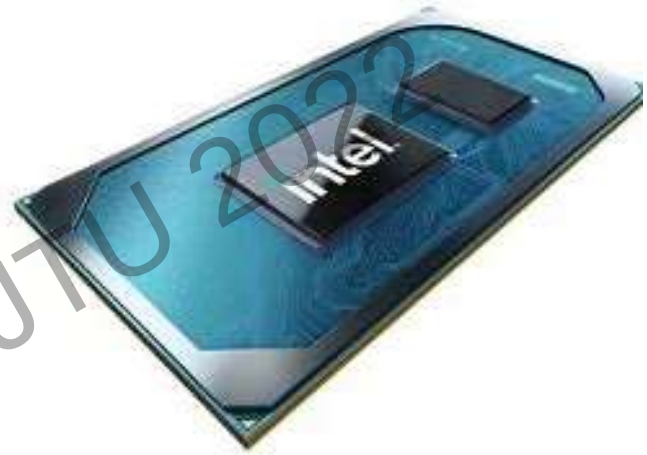
- 16Mhz Clock
- 2KB SRAM
- No floating point unit

=>

- 1 word/cycle: 32MB/s (16-bit word)
- 16 MIPS (less in practice，why?)
- << 16 MFLOPs (why?)

Intel i7-1185G7

- 3.0 GHz(turbo)
- 4 Core/8 Thread
- DDR4 (3200M)



=>

- 32 SP FLOPs/core/cycle => ~800GFLOPS peak – measured ~400
- Memory bandwidth ~ 40GB/s (3200x2x8x80%)

NVIDIA Titan RTX

- 4608 processor cores @ 1.35GHz(base)
  @ 1.77GHz(boost)
- GDDR6 memory 384-bit@14Gbps

=>

- 2 FP32/cycle => 16.3 TFLOPs
- ~672GB/s memory bandwidth

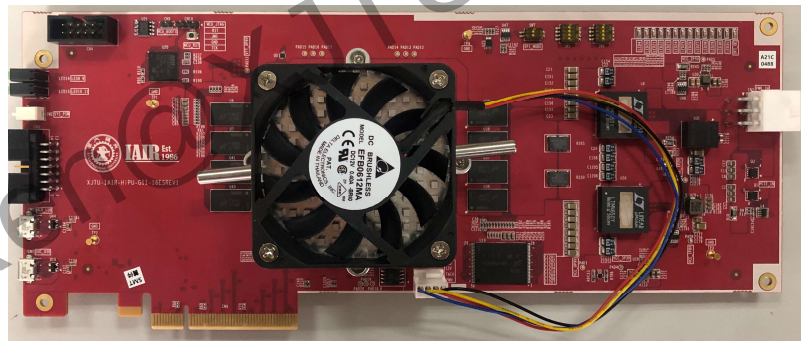
FPGA/Custom hardware

- DSP slices to build FP: 4
- Up to 12,800 slices on high end FPGA (VU13P) => 3200 FP units
- @300MHz => 960 GFLOPS

- Local BRAM resources: ~ 1000 of 8-bitwidth /1 K depth
  - 300GB/s local memory bandwidth
  - But GPUs can do even better than this with local memory
- So Why FPGA ? Allows custom hardware tweaking

HiPU ASIC

- 13 core @1 Ghz
- 512 MAC@INT8（finish 1 multiply and 1 add per cycle)
- 13TOPs



- Two DDR4 sockets
- 2x64bits*2666Mhz=42.6GB/s Memory bandwidth

Multi-objective Optimization

- Speed/Time
- Power
- Area
- Cost

Next Lecture：Graphical Representations

AlexNet：How accurate？

https ://github.com/jcjohnson/cnn-benchmarks

- AlexNet estimate: approx. 0.75GFLOPs
- NVIDIA Pascal TitanX: State of the art from 2016-estimated peak of 10TFLOPS
- Time ~ 5 ms（50 GFLOPs）
- Utilization: **~1.5%**

Where did the remaining 98.5% go ?