A Short Presentation on Digital Signal Processing

IEEE Circuits and Systems Society, Silicon Valley

Shiv Balakrishnan

March 17, 2014
Start of ‘Modern’ Digital Signal Processing

- (Re)Discovery of FFT in mid-60s (big deal?)
  - Cooley & Tukey at IBM
  - Bell Labs: speech, J.Flanagan et al
- Initial apps: filtering, spectral estimation
- Implementation by:
  - HW – big, expensive
    - 19” rack for ~4 2nd-order IIR sections
  - Off-line implementations on minicomputers and mainframes
- Graduate courses in a few schools (MIT, Georgia Tech, Rice et al)
Early example of Coding and Multirate Filters: CD Audio c.1981

- Reed-Solomon coding for Error Detection and Recovery
  - Play frisbee with the CD: no problem. Well, almost.
  - Interpolation by 4x before output DAC (why?)

- DSP approach to polynomial curve-fitting or interpolation (Schafer & Rabiner, 1973)
- Prototype filter with L sub-filters
- Nth-Band FIR filters and how to design them (Mintzer)
- Elegant Polyphase interpretation (Belanger, Bonnerot et al)
- Practical issues with Finite Word Length
Filter Design in the Sampled World (1)

- Initial procedures: mapping of analog filter design to the sampled data space
  - $X(f) \leftrightarrow X(z)$
  - $x(t) \leftrightarrow X(f)$ becomes $x_n \leftrightarrow X(z)$
  - Left-Half Plane maps to Unit Circle
    - System poles outside either one spell trouble....
Digital Filter Design (2)

- Map y-axis to boundary of unit circle
- Several ways to then map s-domain analog filters to corresponding z-domain filters
  - Impulse Invariant Transform
  - Bilinear Transform (perhaps most common for pole-zero i.e. IIR filter design)
  - Properties maintained fairly accurately i.e. Butterworth filter retains its Maximally Flat property
- All-zero (FIR) filters designed directly in the digital domain
  - Chebychev (MiniMax) criterion: elegant solution due to Parks & McClellan, probably the most-used filter program. Makes use of the Remez exchange algorithm to solve the problem with minimum computation
  - Windowed filter design: desired frequency response, appropriate window function \( \rightarrow \) inverse DFT \( \rightarrow \) (Finite) Impulse Response
Digital Filter Design (2)

- Map y-axis to boundary of unit circle
- Several ways to then map s-domain analog filters to corresponding z-domain filters
  - Impulse Invariant Transform
  - Bilinear Transform (perhaps most common for pole-zero i.e. IIR filter design)
  - Properties maintained fairly accurately i.e. Butterworth filter retains its Maximally Flat property
- All-zero (FIR) filters designed directly in the digital domain
  - Chebychev (MiniMax) criterion: elegant solution due to Parks & McClellan, probably the most-used filter program. Makes use of the Remez exchange algorithm to solve the problem with minimum computation
  - Windowed filter design: desired frequency response, appropriate window function \(\rightarrow\) inverse DFT \(\rightarrow\) (Finite) Impulse Response
Digital Filter Design (3)

- What happens if we want to add other constraints e.g. time-domain, overshoot or step response?
  - Not trivial, but can use optimization SW packages which implement Linear Programming

- A couple of other issues rear their ugly heads viz.
  - Quantization of filter coefficients
  - Roundoff noise in fixed point arithmetic
  - Effects: noise (tolerable with reasonable wordlengths)
  - Recursive (IIR or pole-zero) filters can have Limit Cycles, usually a major problem
    - Preventing limit cycles in 2nd order sections well understood, hence common implementation is cascade of such sections

- Can one design knowing quantization of filter coefficients a priori?
  - Yes, using Mixed Integer-Linear Programming but very, very compute-intensive!
  - In practice, design in floating point, quantize, iterate to get desired response

- New result from Georgia Tech: Remez exchange algorithm for fixed point filter design. This should make the quantization problem much easier!
Enough of Digital Filters: Let’s Move On

- Early ’80s saw the emergence of programmable machines to implement signal processing algorithms
  - TI had tasted success with the innovative Speak-n-Spell product in the late ’70s. This was not a user-programmable machine, though.
  - Microprocessors (8086, 6800 – class) were way too slow to handle many signals of interest
  - First Harvard-architecture DSPs from TI, NEC, AT&T (internal use), Motorola: fixed point, could handle speech, audio etc. in real time.
    - If interested in stories from this hoary past, talk to people like Ray Simar, Gene Frantz or Wanda Gass.
  - TI had staying power and ultimately was the market maker: took > 5 years
  - Forward Concepts (Will Strauss) began publishing data on sales of Digital Signal Processors and fixed-hardware implementations, thereby giving DSP its own ‘market’ which could be tracked
Applications Start to Appear All Over

- Early adopters and buyers of any significant size were typically military e.g. beam forming for sonar at the Naval Research Center or radar work at MIT’s Lincoln Labs
- Talent pool was in DoD space, academia; a few volume applications like the first cellular (GSM) phones.
- Even in late ’80s / early ’90s, not many DSP folk in Silicon Valley
- Along came MPEG video, data and audio compression and other multimedia apps in the PC world
- The ’90s saw DFTs, lossy compression, coding etc. become lingua franca of many chip architects and designers
- Since mid-to-late ’90s, communications apps, both wireline and wireless have driven the progress of DSPs and the technology
Applications Start to Appear All Over

- Early adopters and buyers of any significant size were typically military e.g. beam forming for sonar at the Naval Research Center or radar work at MIT’s Lincoln Labs
- Talent pool was in DoD space, academia; a few volume applications like the first cellular (GSM) phones.
- Even in late ’80s / early ’90s, not many DSP folk in Silicon Valley
- Along came MPEG video, data and audio compression and other multimedia apps in the PC world
- The ’90s saw DFTs, lossy compression, coding etc. become lingua franca of many chip architects and designers
- Since mid-to-late ’90s, communications apps, both wireline and wireless have driven the progress of DSPs and the technology
The Fixed Point DSP Dilemma

- At some point, fixed point DSPs had proven their usefulness and price/power/performance
- Programming these beasts was (is?) not for the faint of heart
  - Compilers could not provide enough optimization
    - Inner loops in hand-assembly
    - Either Assembly and C programmers had to learn DSP theory (hard, in spite of courses by now taught routinely in undergrad EE programs)
      or
    - DSP (algorithm) folk had to learn assembly and C programming (somewhat easier, but no bed of roses either)
- Situation today (completely opinion): C compilers have come a long way, TI is industry-leading but:
  - Still a ways to go to make optimization easy
  - Gains at algorithmic or system level tend to be much greater than code fine-tuning
  - The Holy Grail of going automatically from system-level design e.g. MatLab to optimized DSP code remains just that: a Holy Grail, coming one of these days to a design shop near you
  - Claims are made of efficient RTL-to-FPGA push-button design but it is still in development (and is an easier problem than the above)
Meanwhile, Floating Point DSPs are starting to be reasonable in some applications, and are (or should be) much easier for an optimizing compiler
  - Relative ease of use cited by users in selecting FP parts

General-purpose processors have added DSP extensions, so the solution mix is getting muddy

DSPs have added OS support and memory controllers so the distinctions are blurring

High-end applications are increasingly using FPGA + DSP solutions, even though interfaces are hard to design and debug

“DSP” apps are blurring too: filters, FFT, DFT, MPEG, JPEG, coding, error correction (?), encryption (?), …
Going Forward

- DSP market though << uProcessor market should see higher CAGR
- Applications are increasing in the Internet era
- Architectures of DSPs and uProcessors are converging in some ways: VLIW, multi-threading, hardware accelerators, fast context-switching etc.
- Ecosystem of 3rd party products, libraries, IP cores getting established; again, TI is way ahead of the DSP competition
- RISC + DSP architecture (OMAP, others) well established
- Multiprocessor systems will pose a challenge. Seeing a RISC controlling 4 DSPs : haven’t we been there before? Recall the 320C80 product; was too hard to program and some would claim that to automatically find coarse-grain parallelism in an algorithm is practically unsolvable
- Bottom line: no shortage of challenging stuff to work on
Some trends in the DSP space

- Applications have continued apace
  - Wireless and wired communications
  - All PHYs have a lot of ‘heavy duty’ signal processing e.g. equalization and dispersion compensation
  - Multimedia applications continue to grow especially in handheld devices like Smart Phone

- FPGAs have emerged as an interesting platform due to performance, high speed I/Os etc
  - Price/performance hard to beat in simple ‘core’ filter-type algorithms
  - Number of MACs in high-end part is staggering; problem lies in making use of them
  - Development tools continue to improve and are ‘free’
Aside: computation requirements of Up/Down filters implemented on FPGAs

- Modern radios (e.g. SDR) will usually have Decimation filters in Receive path and Interpolation filters in Transmit path
  - Architecture of Xilinx gate arrays map efficiently to Transpose form FIR filters while Altera parts map to Direct form.
  - However, to use FIR filter symmetry, one must use Transpose form in Decimators and Direct form in Interpolators (or the other way around)
  - Thus either way one is left with an inefficiency, which swamps any tool-based “QoR” improvements!
Changes in the DSP market

“DSP Market” began to fragment

Data harder to track as

- Some segments (e.g. Wireless) showed volumes far greater than others
- DSP capability was more often buried inside ASICs
- It became more useful for people to see data broken out into vertical segments like Video
- Even within a company like TI, the wireless “ASICs” dominated the “Catalog” DSP
DSP market by category (Forward Concepts)

DSP SILICON MARKET BY TYPE

2008: $27.2 Billion (WW)

- CELLPHONE
  - DSP ASSPs: 29%
  - DSP SoC/FASICs: 51%
- NON-CELLPHONE
  - DSP ASICs: 3%
- FPGA & PARALLEL DSPs
  - MPUs & OTHER: 2%
- DISCRETE DSP CHIPS: 11%
DSP silicon by market segment: 2008 (Forward Concepts)
Programmable DSPs continue to intersect GPPs

- As capabilities of General Purpose Micros were continually enhanced with DSP features, the distinction is increasingly blurred
  - ARM chips are increasingly capable
  - TI has OMAP parts which don’t have a TI DSP part at all…
  - On top of which there are parts from Tensilica et al which have very specific instructions giving great performance in some cases
Trends in Programmable DSPs (contd.)

- The trend to heterogeneous cores on a single die continues, if only to satisfy the ASIC-like nature of many DSP-intensive SoC’s
  - Hard to match hard-wired silicon for price/performance/power in standards-based applications as in Wireless
- There are apparently chips with up to 4 processors with different instruction sets!
Another trend: Digital-assisted Analog Design

- One well known example is from the Wireless space where Linearization of Power Amplifiers (PAs) is crucial in many OFDM systems. (As is reducing Peak-to-Average power ratio but that is a different problem)
  - Solution is usually “pre-distortion” of some kind
- More aggressive approach suggested by M.Horowitz (Stanford) and others
Digital-assisted Analog (from Horowitz et al)

- Analog design getting harder
  - More bits, faster, lower power budget
  - $F_t$ may be better but matching is worse, $V_{dd}$ range smaller
  - $>50\%$ of chips have some analog

- Digital power scales better, for same energy tens of thousands of gates are “free”
Digital-assisted Analog (from Murmann via Horowitz et al)
Digital-assisted Analog (from Murmann via Horowitz et al)

Example: Pipeline ADC Prototype

- Open-loop amplifier in first, most critical stage
- Statistics based system ID
  - allows continuous parameter tracking
- Judicious analog/digital co-design
  - Only two correction parameters (linear and cubic error)

~8400 Gates (0.042mm² in 0.13µm)

[Murmann, Boser, JSSC 12/2003]
DSP in recent applications

- Software Defined Flexible Transponders
- Chromatic Dispersion, Polarization-mode Dispersion compensation
- Timing Phase Recovery, Frequency Offset Estimation, Carrier Phase Estimation
- Multiple baud rates, bandwidths an modulation formats for high spectral efficiency transmissions
- Current transponders for long haul systems are based on single carrier transmission, partly because unlike wireless channels, the fiber-optic channel is largely frequency-flat
DSP applications going forward

- The applications of Digital Signal Processing are:
  - Ubiquitous in many day-to-day products like cell phones, multimedia devices, home routers etc
  - The $$ value of the worldwide market has risen though the data is fragmented
  - Safe to predict new and more demanding applications of the technology realized in HW/SW/FW and Applications

- Thank you for your time!