NGCodec: Using High Level Synthesis and SDAccel to Develop Best-in-class HEVC/VP9 Video Compression

Presented By

Name  Adam Malamy
Title  Co-founder & VP Technology
Date  October 2 2018
NGCodec Product

FPGA based Video Compression

x.265 very slow ~ 1 frame/sec
NGCodec HEVC 60 frames/sec and better VQ
Similar results for NGCodec VP9

2.1M lines HW Verilog RTL source code
331K lines HW C++ source code
128K lines HW C++ verification code
87K lines algorithmic reference model
Real Time Cloud Video Transcoding

e.g. twitch
Value of HW Acceleration

Bandwidth and storage costs
(Service provider CDN & consumer data plan)

Operating Expenses

Quality of experience
(Startup time, visual quality, stalls)
NGCodec Mission

To own Cloud Video Encoding

- Based on FPGA acceleration, with no load on host CPU
- Provide best VQ, latency, channel density, cost per RU
- Support all major video standards (AVC, HEVC, VP9, AV1)
- Make our solutions look like software encoders
- Deliver the same VQ for live as slow off-line SW encoders

Partners:
## Encoder Basic Stats

<table>
<thead>
<tr>
<th>Device</th>
<th>Virtex UltraScale+ 9 (VU9P)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>1182240 LUTs, 6840 DSPs, 4320 BRAMs, 960 URAMs</td>
</tr>
<tr>
<td>Used</td>
<td>461038 LUTs, 2736 DSPs, 1824 BRAMs, 698 URAMs</td>
</tr>
<tr>
<td>Frequency</td>
<td>200MHz, Power, ~20W</td>
</tr>
<tr>
<td>Performance</td>
<td>1080p60 real time encoding</td>
</tr>
</tbody>
</table>

© Copyright 2018 Xilinx
Use of HLS in Design Flow

- 12 Major design components written HLS
- 10 or 15 more smaller HLS modules
- Top level of design is Verilog RTL
- Fifos, Memories, module start/done logic in RTL
HLS Design Methodology

- Everything in Green is in C/C++
- Everything in Yellow is handled in vivado_hls
- Reduced design time
  - After learning curve
- Reduced verification time
- Reduced EDA costs
- Reduced staffing requirements
  - Especially for verification
What is SDAccel?

- Designed for PCIe interface to FPGA accelerators
- DSA is custom per board and handles all I/O - hardware abstraction
- Kernel is the Designer’s IP and can be used across boards

*Device Support Archive
SDAccel Advantages for NGCodec

- Our business is cloud-based video encoding
- We have three major deployments so far, all with different PCIe hardware
- We can use common encoder kernel, i.e. a common design, for all boards
- Reduces board-specific debug by 90%
  - Differences like DDR frequency still remain
Some Problems with SDAccel

- DSA is large - occupies $\frac{1}{2}$ SLR, 16% of VU9P
- Restricts visibility into Microblaze
  - No run-time debugging
  - We do our debugging first in a non-SDAccel build
Next Steps for NGCodec

- Currently our encoder can encode 1080p60 in real time and occupies 50% of a VU9P
- Next step - double the throughput - 2x1080p60 in real time on the VU9P
  - With the same quality
- How do we accomplish this?
Method 1 - Run twice as fast

- Change pipelining to run modules at 400MHz rather than 200MHz
- Works well on modules with no feedback loop
- More optimally uses DSP and BRAM resources
- HLS is very good at this type of redesign

200 MHz Design Original

400 MHz Design Redesigned
Method 2 - Process 2x data at 200MHz

Some designs have feedback loop, cannot start calculation n+1 until n completed

Increasing clock frequency does not help

But can be clever and process two independent sets of data in the same time as processing one

HLS is not so good at this type of acceleration
Designing with HLS vs. Designing with RTL

- Vivado_HLS reduces design time and verification time
- 90% of the time it is a benefit once you become proficient with the tool
- There is a level of complexity at which old-fashioned RTL design still wins out
  - Max frequency
  - Min resources
  - Complicated dependencies
Adaptable. Intelligent.

Thank You For Attending

NEXT GENERATION VIDEO COMPRESSION

© Copyright 2018 Xilinx