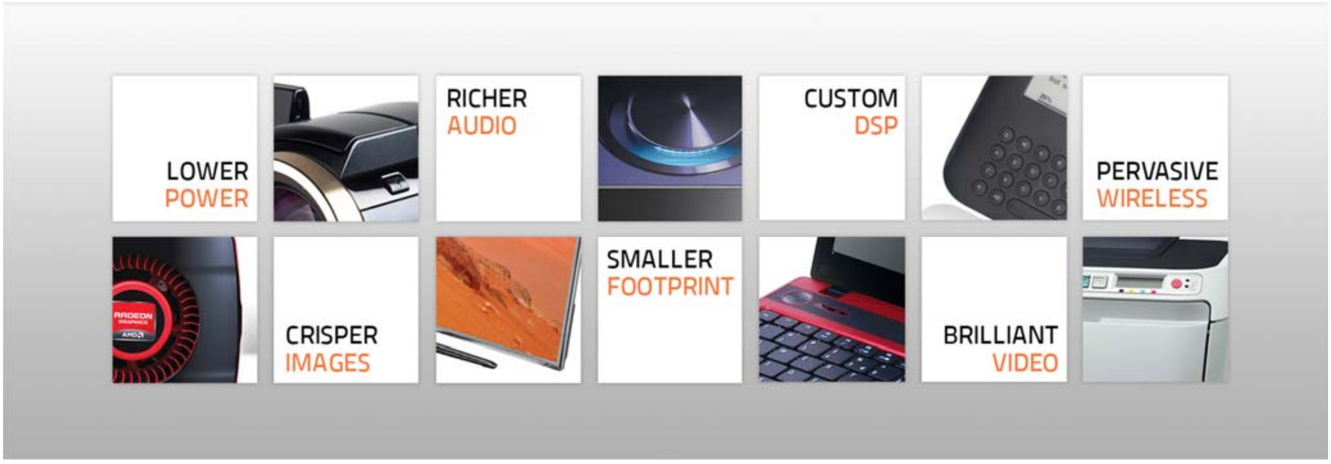
Grand Challenge Scaling - Pushing a Fully Programmable TeraOp into Handset Imaging

Chris Rowen
Cadence Fellow/Tensilica CTO



Outline

- Grand challenge problem for the next decade: video and vision intelligence
- An example problem: bilateral image filtering
- A new platform for imaging – IVP
- Scaling IVP

Imaging/Video Becoming Platform Differentiator

Advanced ISP

High dynamic range (HDR) image capture

Low-light image enhancement

Scene analysis

tal
m

Still-image & Video Capture

Handsets, Tablets PCs, DSCs

Video pre- and post-processing

Stabilization

Computer Vision

DTV, Tablets, PCs, Consumer Gaming, Glasses

Face recognition & tracking

Feature detection/classification

Face detection and recognition

Gesture control

Scene analysis

Video Post-Processing

DTV, STB, Mobile

Content-Adaptive scaling

Sharpening

Video enhancement

Auto (ADAS)

Advanced Driver Assistance Systems

Lane departure warning

Traffic sign detection / recognition

Automatic high beam

Front collision warning

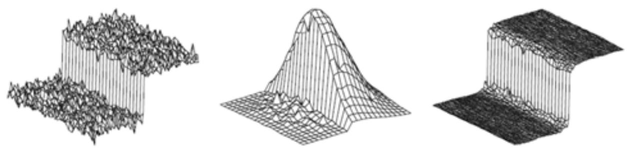
Pedestrian detection and tracking

Bilateral Filter

- Highly selective, edge preserving filter
  - Noise reduction, edge preserving blur, 3D depth filtering and many other apps
- Uses dot product of two kernels:
  - Spatial kernel
  - Image gradient kernel
- Gradient kernel varies adaptively with image content
  - Filter kernel needs to be re-calculated every pixel
  - Normalization requires a division
- ~500 RISC operations per pixel across multiple components
- To do 4K x 2K @ 60fps:

  > 500M pixels per second

  250 billion operations per second



$$I_{new}(x,y) = \sum_{j=y-\frac{n}{2}}^{y+\frac{n}{2}} \sum_{i=x-\frac{m}{2}}^{x+\frac{m}{2}} w(i,j,x,y)\, I(i,j)$$

$$w(i,j,x,y) = \frac{1}{\sum w}\, e^{-\frac{1}{2}\left(\frac{d(i,j,x,y)}{\delta_i}\right)^2}\, e^{-\frac{1}{2}\left(\frac{g(I(i,j),I(x,y))}{\delta_c}\right)^2}$$

$$d(i,j,x,y) = \sqrt{(i-x)^2 + (j-y)^2}$$ Spatial component

$$g(I_1, I_2) = |I1 - I2|$$ Gradient component

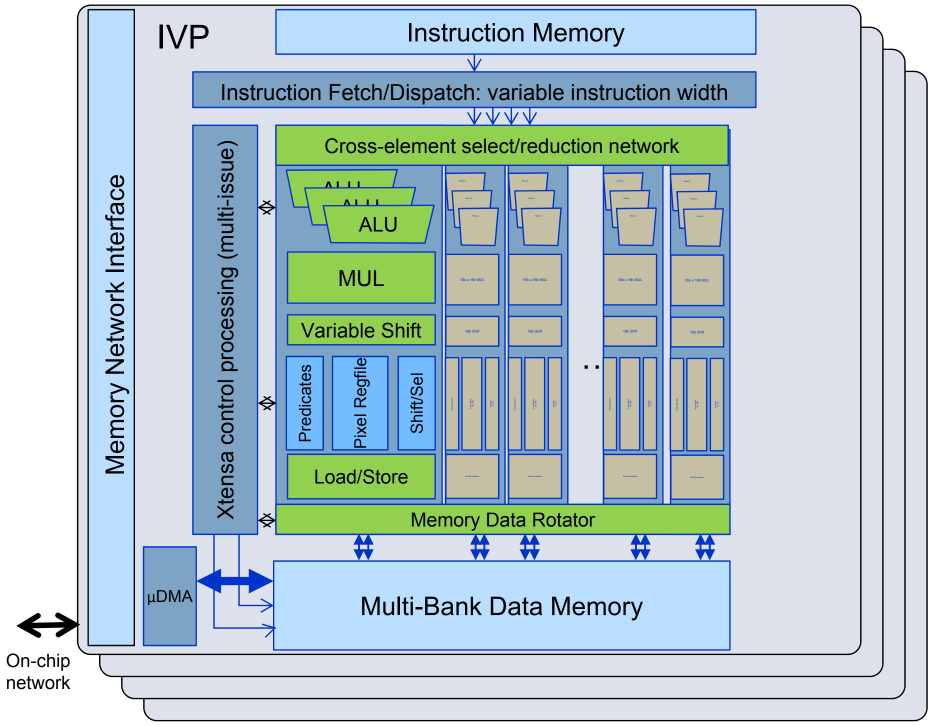
Target Platform: IVP

The Problem: Extreme computation and energy demands for

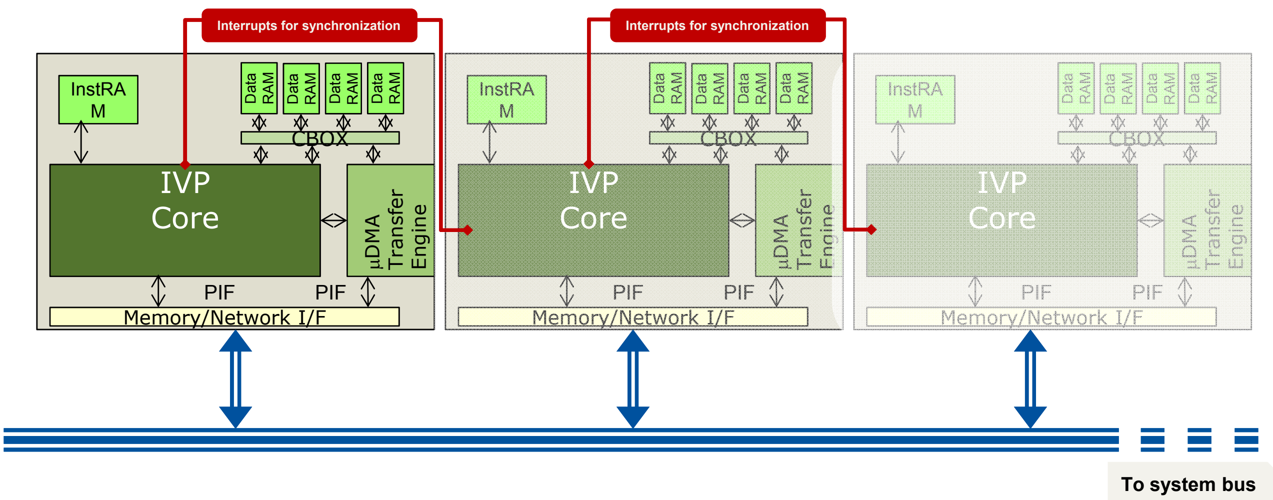
- Advanced imaging
- Vision
- Gesture
- Video Post-processing

The Solution:

- IVP: Many parallel “element engines” + Xtensa control programmed as SIMD uniprocessor
- Each element engine – many operations per cycle:
- Deep DSP pipeline for high clock frequency
- Powerful data reorganization:
- Predicated architecture
- Mature SIMD/VLIW software tools and libraries



Scaling IVP



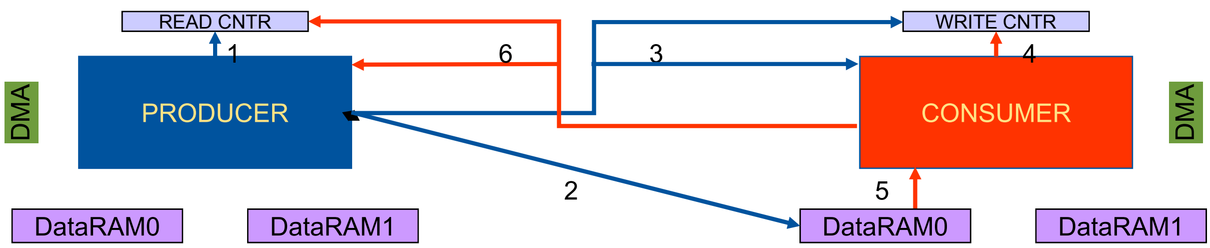
- Flexible compute pipeline / parallel execution
- Control via interrupts
- Data via uDMA
- Peak 16b operation rate approaching 1 TeraOp

Multicore Considerations

- Cores can communicate as follows:
  - Directly with neighboring cores via direct shared memory and uDMA, managed via interrupts.
  - Read/write system memory directly or, more commonly, via uDMA
- uDMA:
  - Can write to the local memory of neighboring cores.
  - Can read and write system memory and local core memory.
- Synchronization is done via interrupts and shared memory
  - Standard API provided via “Xtensa Communication Protocol”.
  - Using interrupts allows easy power management of inactive cores.
  - Overhead easily absorbed for common imaging tasks

A channel-based communications on distributed shared memory

- Unidirectional communication channel between a producer-consumer task
- Consumer allocates buffer space in its local memory: Data allocated and consumed in FIFO order
- Read counter at producer and write counter at consumer
  - Track read/write addresses in the buffer
  - Updated by remote core after read/write
  - Local variables
- Requires direct writes/DMA from producer to consumer
  - Ability to notify each other – interrupts/MMIO registers
- OS neutral: API to register callbacks for enabling/disabling interrupts, task sleep/wakeup – tested on XOS/XTOS.

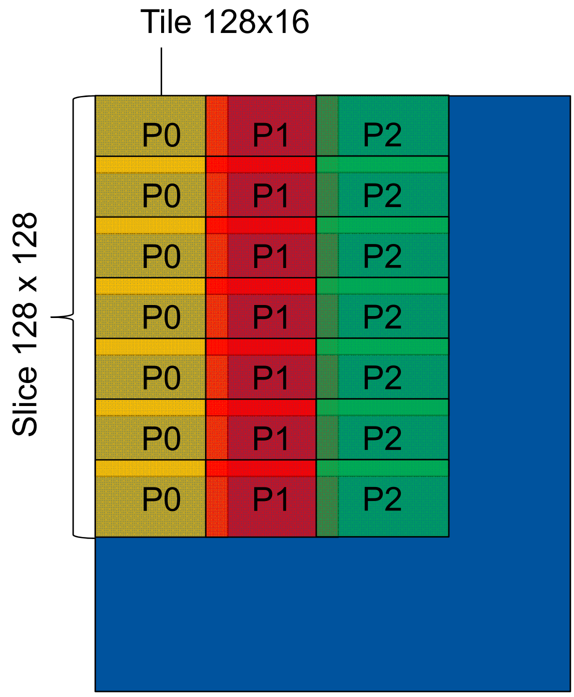


1) Allocate: check if buffer space available in consumer
2) Transfer: DMA data from producer to consumer at read cntr index
3) Commit: Update write cntr & interrupt consumer
4) Get block: check if data available
5) Consume at write cntr index
6) Release: Update read cntr & interrupt producer

8

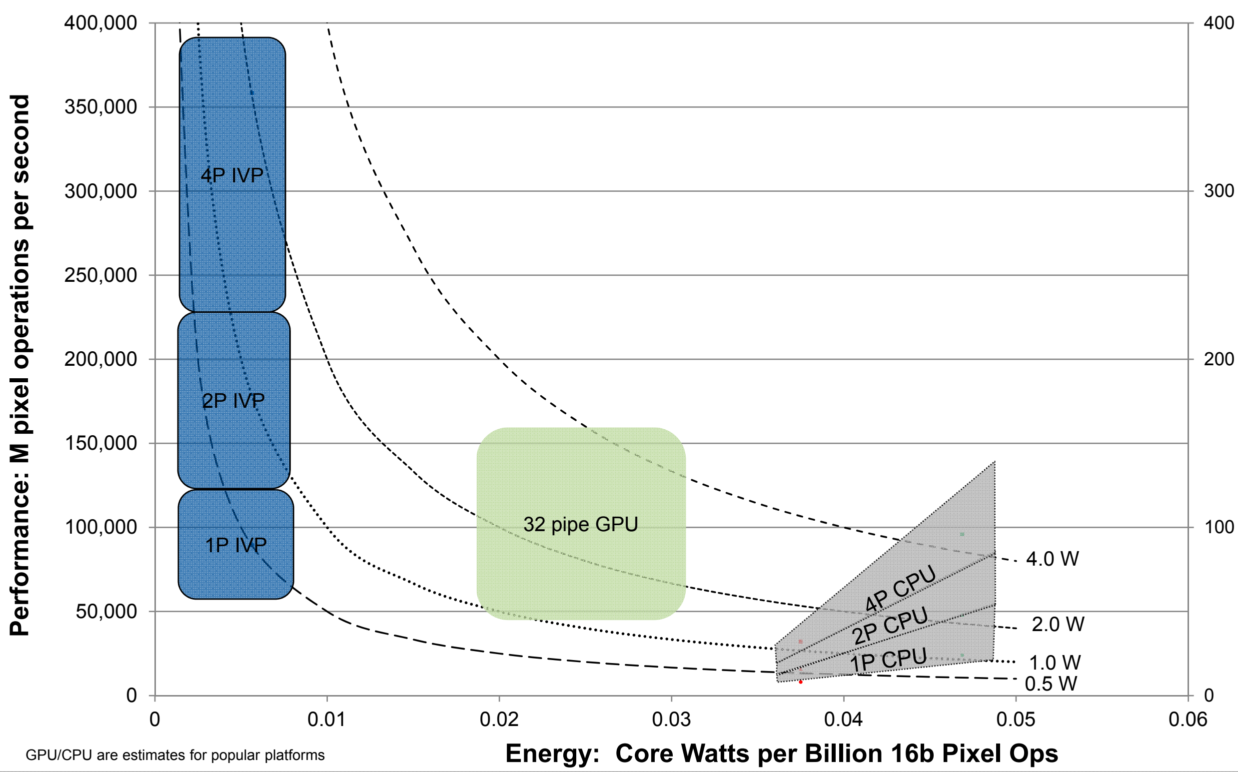
Using Multiple IVPs for Bilateral Filter

- Ample parallelism in complex filters
- Use uDMA to fetch and send tiles from/to the image in main memory.
  - Each tile can be computed independently
  - Overlap region, especially overlap between horizontally-adjacent cores, fetched twice
  - Tiles are fetched to local memory using a double buffering
  - Processed tiles are sent back to memory via the uDMA, again offloading the processor.
  - Leverages single-core "Tile Manager" programming model
- Daisy-chained interrupt notification of new data availability, and task completion
- Reduced overhead with increased slice size



Complementary Processing Opportunity

More Ops, Lower Power per Ops



GPU/CPU are estimates for popular platforms

Results

- IVP takes general-purpose image processing to new efficiency level: <0.005 W per B 16b pixel ops
- Imaging suitable for homogeneous distributed shared local memory MP scaling
- MP app just a modest effort from single-core app – a few engineer days
- Many options for multi-core imaging programming:
  - Image partitioning into parallel sub-tasks
  - Producer-consumer task processing chains
  - Dynamic task pool allocation