embedded VISION SUMMIT

Porting Applications to High-Performance Imaging DSPs

Gary Brown, Imaging/Video BU, IPG

cadence

October 2, 2013

Application Drivers:

complexity ➔ programmability

pixel-operation rate ➔ parallelism

embedded VISION SUMMIT

Still Image and Video Capture

Handsets, Tablets PCs, DSCs

Stabilization

Digital zoom

High dynamic range (HDR) image capture

Scene analysis

Face recognition and tracking

Low-light image enhancement

Video pre- and post-processing

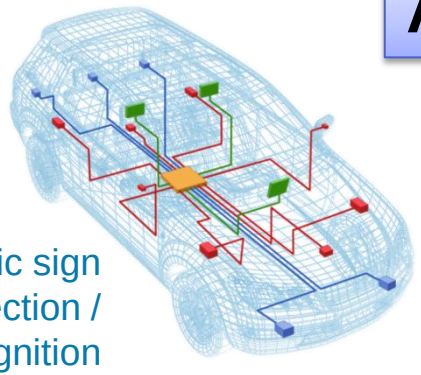
Computer Vision

DTV, Tablets, PCs, Consumer Gaming

Face detection and recognition

Gesture control

Auto (ADAS)



Advanced Driver Assistance Systems

Lane-departure warning

Traffic sign detection / recognition

Front-collision warning

Automatic high beam

Video Post-Processing

DTV, Mobile

Scaling

Sharpening

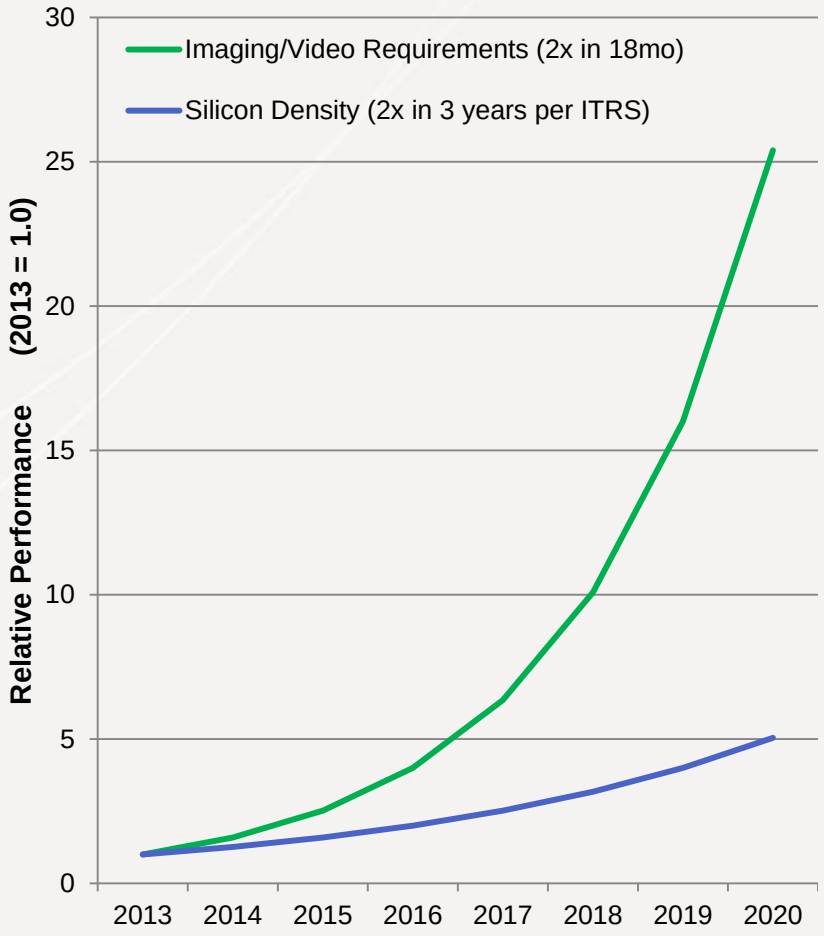
Video enhancement

cadence

Moore's Law vs. Imaging

- Moore's Law—silicon scaling: 25%/year
- Imaging performance drivers:
  - Still image res: 8 Mpixel ➔ 20 Mpixel **2.5X**
  - Still image to video enhancement: e.g. HDR: 8 Mpixel@1fps➔HD1080 (2 Mpixel@30fps) **7.5X**
  - HD1080 ➔ UHD (4Kx2K) video pre/post-processing **4X**
  - Advanced video: feature extraction, tracking, and identification **10X**
- Overall, imaging/video performance must scale by 2X every 12-18 months
- Power budget remains fixed

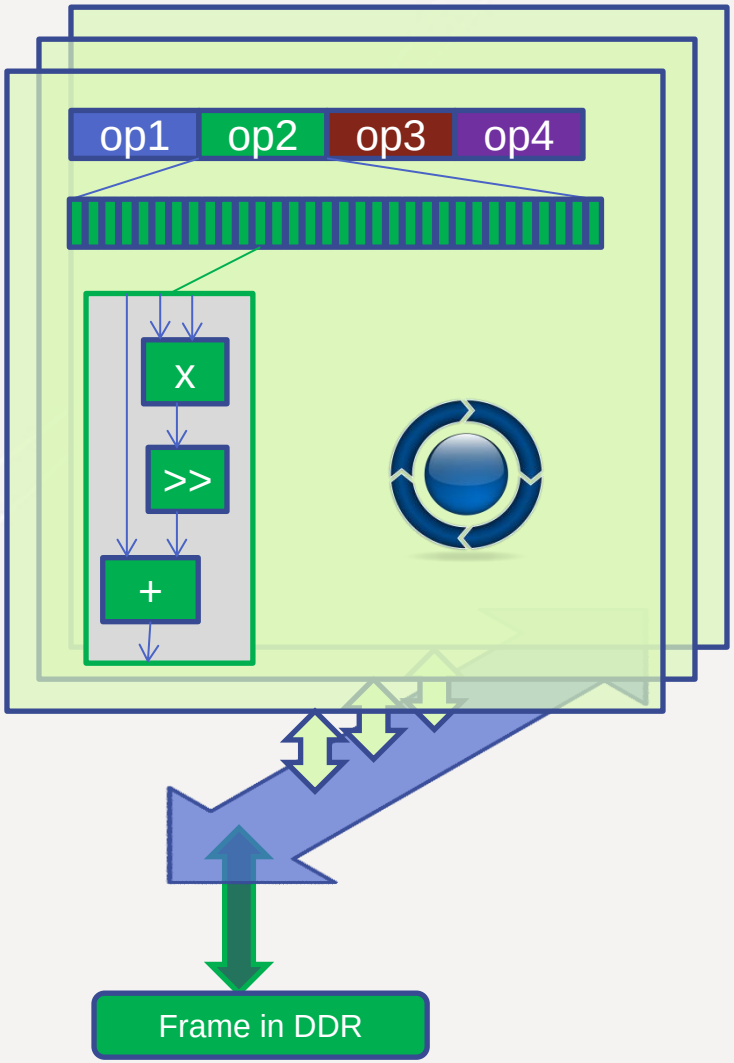
Implication:
New architectures will emerge to solve new imaging/video problems



Imaging Throughput Driven by Parallelism

Six components of imaging performance

1. Instruction issue parallelism: VLIW or superscalar operations
2. Data parallelism: SIMD
3. Compound operations: Specialized pipelines doing multiple ops per SIMD element per operation
4. Multi-core
5. MHz
6. Compute/data reference concurrency: prefetch/DMA



Platform Types for High-Performance Imaging

	CPU	GPU	Hardwired Function Block	Imaging DSP
Range of Data Types	8b,16b,32b,32b FP,64b FP	8b,16b,32b,32b FP,64b FP	Various	8b,16b,32b,32b FP,64b FP
Optimal Data Type	32b	32b FP	8..16b	16b
Vector Width	x4-x8	x1-x2	Varies	to x32
Vector Operation Issue	1-2/cycle	1-2/cycle	NA	To 4/cycle + control ops
Parallel Cores	1-4	1-32	Varies	1-4
Energy per Imaging Op	High	Medium	Very low	Low
Imaging Sweet Spot	Heavy general-purpose compute, light imaging	Heavy graphics compute, light imaging	Simple, fixed-imaging pipeline	Complex imaging

Porting and Tuning Applications on a High-Performance Imaging DSP—Key Considerations

1. For lowest energy and highest performance, use the lowest-resolution computation consistent with target output quality. Fixed point typically 3-4X lower energy than floating point.
2. Identify fundamental sequential dependencies in algorithm—restructure to expose as much data parallelism as possible.
3. Map out frame data usage to minimize bulk traffic between DDR and computational resources. Restructure for sequential or "tiled" access. Plan for DMA/pre-fetch.
4. Understand library and software development resources to ease programming—for example:

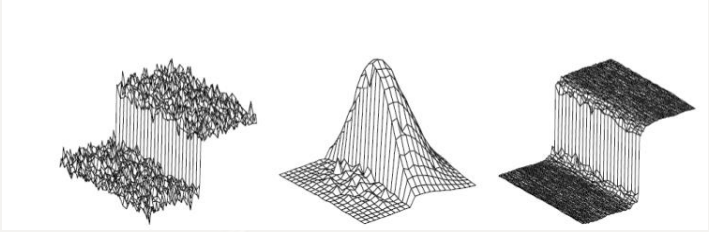
- Standard imaging libraries
- DMA/tile manager
- Examples library
- Full application example source
- C runtime libraries with POSIX I/O
- Auto-vectorization with vectorization feedback
- Extended imaging type system with extended C operators
- Intrinsics imaging functions
- C/C++ source debugging
- Physical and virtual prototyping boards and models
- Pipeline-level and ISA usage profiling

Porting and Tuning Applications on a High-Performance Imaging DSP—Likely Steps

1. Measure reference cycle performance and image quality
2. If necessary, convert floating point types to lowest fixed-point scalar equivalent that meets image quality need
3. Identify any inherent loop dependencies and restructure access patterns to move dependencies from inner loops to outer loops
4. Decompose deep table lookups into computed function of shallow table lookups
5. Compiler with auto-vectorization or adopt vector data types
6. If necessary, add vector reorganization operations to maximize vector usage
7. If necessary, look at VLIW packing and adjust operation selection
8. Use DMA or “tile manager” library to pre-load/post-store data in background
9. If MP configuration, partition data and add calls to communications library to distribute computation across processors

Example 1:
Tuning an Application: Bilateral Filter on IVP

- Pixel adaptive, edge preserving filter
- Two filter kernels
  - Spatial kernel—always fixed
  - Intensity kernel—coefficients change based on pixel intensity variations
  - Coefficients are pre-computed and stored as 16b fixed-point values
- Intensity kernel poses some challenges for SIMD processors because of adaptive nature
- All other parts of filter can be easily vectorized



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

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

Spatial component

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

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

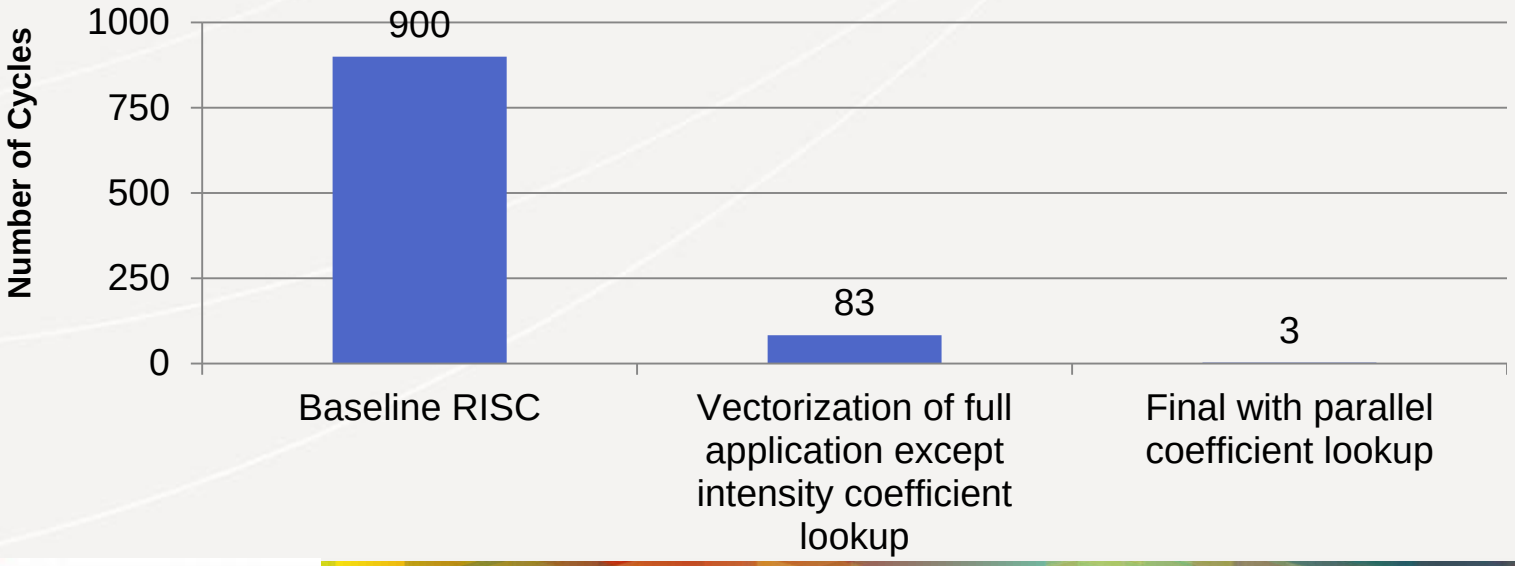
Gradient component based on pixel intensity

Reference: Bilateral Filtering for Gray and Color Images, C Tomasi & R. Manduchi, 1998 IEEE International Conference on Computer Vision

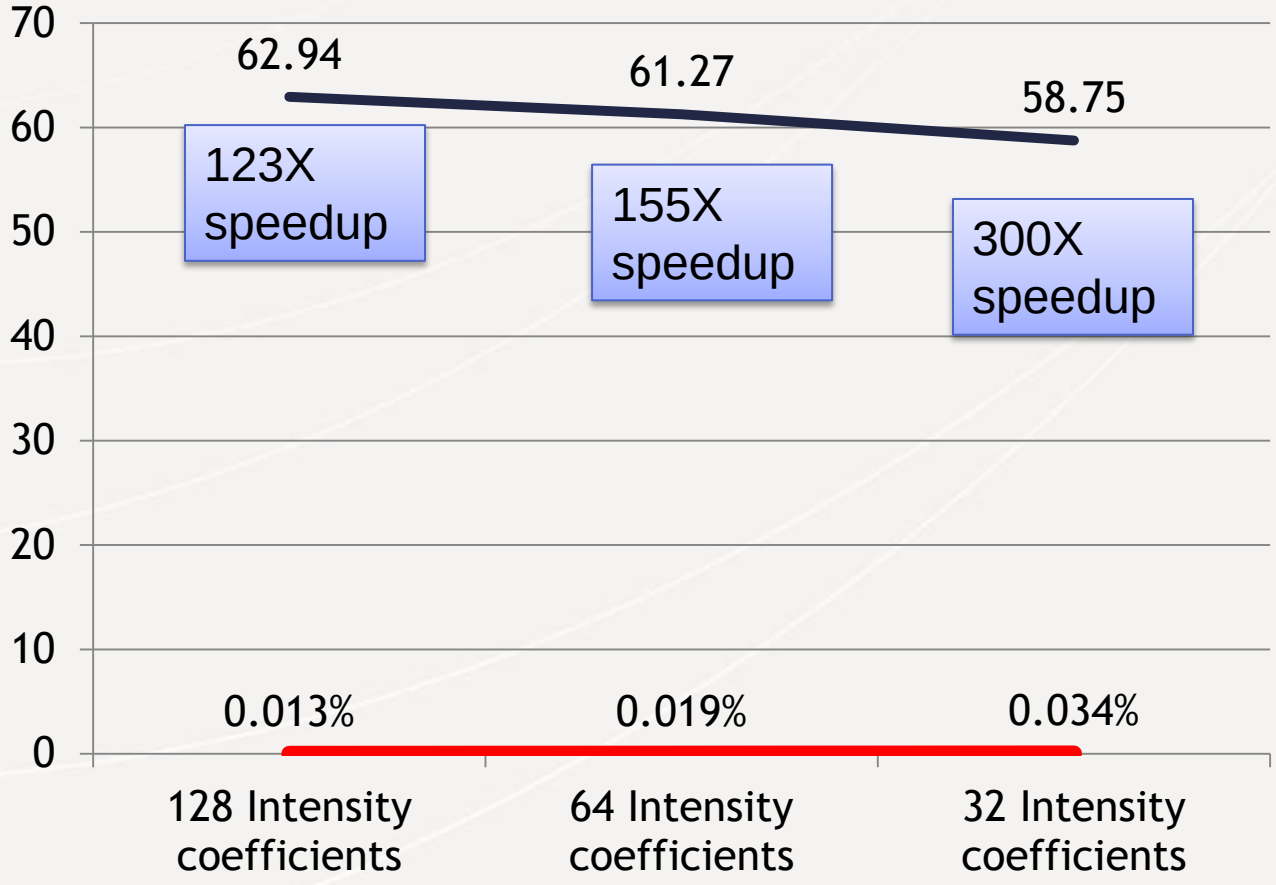
Example 1 (Continued)
Tuning an Application: Bilateral Filter on IVP

Compare performance of

1. A baseline RISC implementation of 5x5 bilateral filter
2. Vectorization of all computations except for intensity coefficient lookup **(10X faster than baseline RISC)**
3. Vectorization of intensity coefficient look-up, with optimized parallel lookup operation **(300X faster than baseline RISC)**



Quality vs. Performance Comparison when 32, 64, or 128 Different Intensity Coefficients Used



All 3 schemes have high PSNR compared to a floating point implementation

Cadence Imaging IP Offering

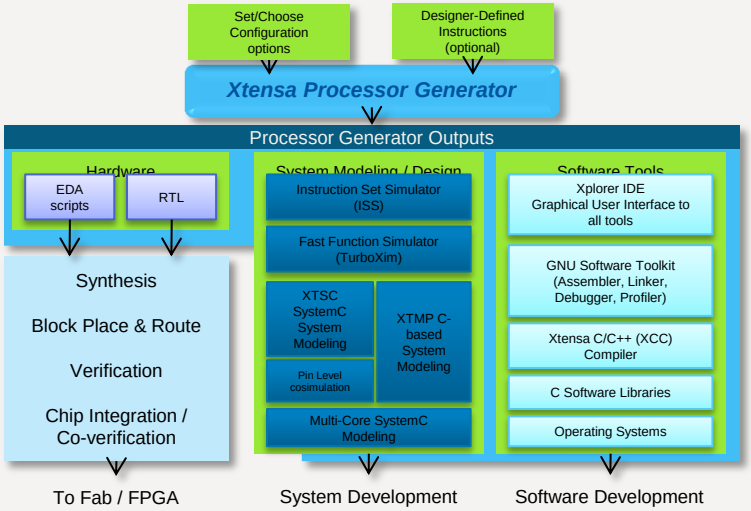
- Leading embedded DSP IP, part of Cadence's IP business
- Shipped over 2 ½ B cores in silicon to date
- Leader in embedded imaging space
  - Over 750M chips shipped in imaging/video to date
- Launched new flagship IVP imaging/video IP in August 2012
  - Traction in mobile and consumer applications
  - Programmable VLIW, vector architecture

What Cadence Provides

- **IVP: Imaging/Video DSP**
  - Complete hardware ready for integration
    - RTL, EDA scripts, testbenches
  - Software modeling tools
  - Cycle-accurate simulator; fast functional simulator; system modeling; pin level cosim
  - Development environment
    - Integrated development environment (IDE)
    - Software toolkit (assembler, linker, debugger, profiler)
    - Compiler with auto-vectorization
    - C software libraries
  - Documentation, support
  - FPGA emulation environment
  - Access to 3rd party network of application developers

Cadence® Tensilica® Xtensa™ application-specific imaging cores:

- Xtensa configurable, extensible IP core, customized with instruction set extensions
- Web-based Xtensa Processor Generator for configuring the core
  - Outputs the hardware, system modeling, and software tools matching the customized core



Bilateral Filter:
What to Look for in Generated Code

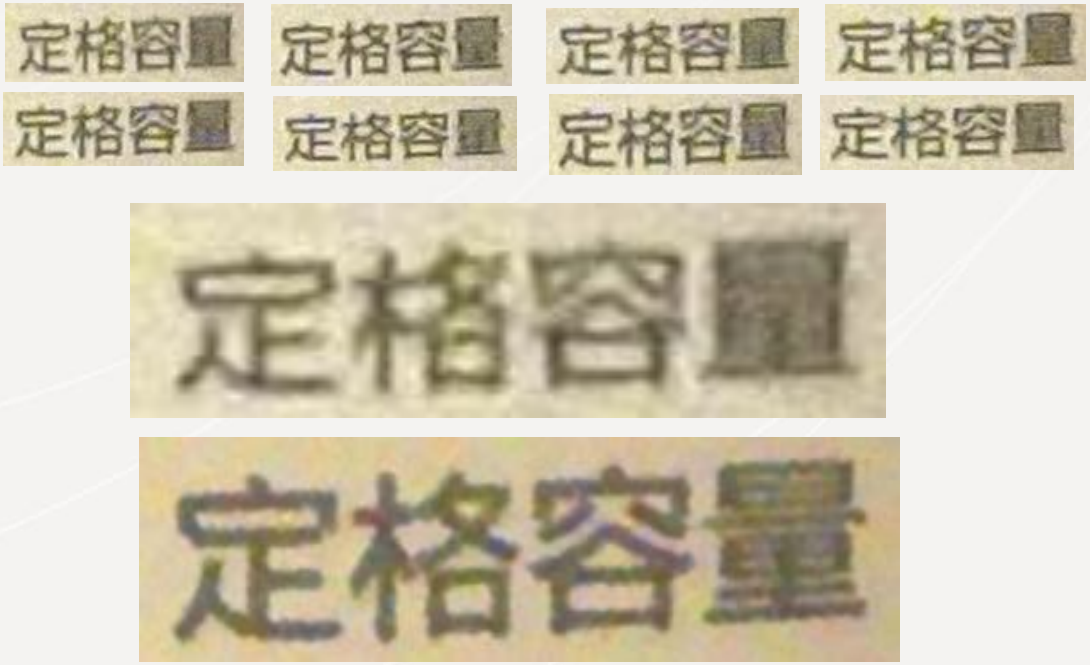
Slot 0	Slot 1	Slot 2	Slot3
{ivp_abssub32x16 v15,v4,v7;	nop;	ivp_mul32x16packp v14,v14,v1;	ivp_abssub32x16 v2,v11,v7}
{ivp_movvsv s3,v2,8;	ivp_lv32x16.i v15,a1,0x1c0;	ivp_abssub32x16 v2,v9,v7;	ivp_movvsv s2,v15,8}
{ivp_movvsv s1,v2,8;	nop;	ivp_mul32x16packp v13,v13,v1;	ivp_sel32x16i v10,v6,v3,1}
{ivp_abssub32x16 v1,v3,v7;	nop;	ivp_mul32x16packp v15,v15,v1;	ivp_sel32x16 v2,v12,v14,s2}

- Full vectorization—32 elements per operation
- 3-4 operations per issue bundle
- At or near multiply-bound or load/store bound, whichever is tougher
- Heavy use of imaging-optimized operations (absolute difference)

Example 2:
Precision Challenge in Computational Imaging

- 8-bit pixels are still very common today, although trend is towards increased pixel resolution
- For high quality advanced imaging, precision grows quickly during processing
- It is common to preserve greater than input pixel resolution at intermediate processing steps such as high quality fusion and pyramidal decomposition or when processing is done in frequency domain (such as with 2D DCT or FFT)
- Ability to deal with growing precision and high performance for 16-bit pixel data is key to good performance for computational imaging

Example 2:
Precision Challenge in Computational Imaging



- Superresolution (and other multi-image computation) requires high-precision alignment and weighted fusion of multiple frames:

$$I_{out}(x,y) = \sum_{j=1}^{N} F_j(I_j, W_j, x, y)$$

Superresolution application courtesy:

Almalence

Example 2:
Precision Challenge in Computational Imaging

- There should be no loss of information while combining/merging images
  - Superresolution is possible due to the presence of 'aliased components'
  - And these aliased components are very low in amplitude—usually deeply buried within the image noise
- Multiple images (around 10) are fused together, and each has 8-bit precision
  - Need 11 to 12 bits of precision to avoid loss of information in summation
- Not only multiple images are summed, but also contributions from multiple input pixels are used to compute single output pixel
  - Combined with the fact that contribution from each image is weighted individually—the required precision needs 3 to 5 more bits during summation (so requirement becomes 14 to 17 bits)
- Also, there are pixel interpolations embedded within nonlinear function F(), which further raises precision requirement by a few more bits
- Need to keep intermediate results at higher than 16-bit precision

Superresolution application courtesy:

Almalence

Wrap-up

- Imaging has almost infinite appetite for more computing throughput at low power
- Rapid change in applications and performance needs implies:
  - ➔Emergence of new platforms
  - ➔Big focus on ease of porting
- Successful porting wants a structured method
- Bilateral filter and SuperResolution examples show major steps
- Resources:
  - Articles: www.embedded-vision.com
  - Application examples: http://www.almalence.com/
  - Company Information: www.cadence.com www.tensilica.com
  - Brief on Vector and SIMD Processors: http://meseec.ce.rit.edu/756-projects/spring2013/2-2.pdf