Signal Processing Libraries for High Performance Embedded Computing

GE IP HPEC Center Of Excellence, Billerica, MA

March 2014
Agenda

1) Common Hardware architectures for HPEC.
2) Programming challenges.
3) Potential solutions.
4) VSIPL – Industry standard abstraction API:
   - Concept
   - Functional coverage
   - API example
5) Demo – VSIPL deployed on CPU & GPU in a unified demo application.
MOSA (Modular Open Systems Approach)

- Bringing HPC silicon into embedded market
- High performance, low latency CPU & GPU clusters
- Open Standard Middleware - Common APIs

- DoD Common Back End Processor
- Open System Architecture (OSA)
- Designed for affordable change
- Open interfaces and APIs

Reducing software costs, simplifying tech refresh
Signal & image processing for radar, sonar, Multi-INT, EW
Common HPEC Architectures in Mil/Aero
Bringing HPC technology to the embedded market
Common HPEC Architectures in Mil/Aero

Bringing HPC technology to the embedded market
Problems faced by application developers

Rapidly evolving hardware environments
Spiraling software costs due to tech refresh.
Need to protect software development.

• **Portability**: -
  - Different processing paradigms.
  - Different operating systems.

• **Performance**: - Take advantage of processing resources available.

• **Productivity**: – Fast to develop & debug.
Potential solutions

Parallel computing extensions to C / C++
CUDA – NVIDIA specific, limited support beyond GPUs.
OpenCL – Standard becoming more widely supported.
OpenACC – Higher abstract level via compiler directives.

Abstraction libraries
OpenCV
FFTW, cuFFT, NPP
cuBLAS, CULA, MAGMA
VSIIPL – An Industry standard abstraction library

Vector, Signal an Image Processing Library

- High level of hardware abstraction.
- Common ‘C’ API – e.g. Intel, PPC and GPU
- ‘Under the covers’ optimization by vendor – no need for user expertise in hardware acceleration.
- CPU implementations can be multi-threaded – automatically distribute computation over multiple cores.
- GPU utilize CUDA or OpenCL
- Managed by the OMG (Object Management Group)
- For full spec see http://www.omg.org/spec/VSIPL/

C++ API also specified but not widely supported (VSIPL++)

For more info: http://www.omg.org/hot-topics/vsipl.htm
VSIPL Programming model

Highly portable

Data is in ‘User Space’ or ‘VSIPL space’

Data in VSIPL space cannot be modified by anything other than a VSIPL function.

Admit to VSIPL Space
- Process data

Release to User Space

Architecture agnostic.
VSIPL – Functional coverage

- Vector Math
  - Basic vector operations – (vmul, vadd, vma, vsqrt, dot product & many more)
  - Transcendental functions- (vsin, vcos, vsinh etc)
  - Selection operations – (vmax, vmin etc)

- Data sorting, interpolation, conversion & copying, random number generation

- Signal processing
  - 1D, 2D & 3D FFTs
  - 1D & 2D Convolutions
  - Correlations & histogramming.
  - FIR Filters
  - Windowing

- Matrix math & Linear Algebra
  - Matrix operations (Matrix products etc)
  - Linear system solvers (Cholesky decomp, LUD, QRD, SVD)

<table>
<thead>
<tr>
<th>BLAS level 1</th>
<th>BLAS level 2</th>
<th>LAPACK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

imagination at work
VSIPL – Specification Profiles

- VSIPL CoreLite – Basic profile, approximately 80 signal processing & math functions.

- VSIPL Core 1.0 – over 512 functions.

- VSIPL Core 1.1, 1.2, 1.3 and 1.4 – incremental functional additions.
  - Maintaining compatibility with previous profiles.
/* create input data block and view and admit to VSIPL Space*/
vsip_cblockbind_f(cc_re, cc_im, N, VSIP_MEM_NONE);
vsip_cblockadmit_f(cc_block, VSIP_TRUE);
cc = vsip_cvbind_f(cc_block, 0, 1, N)

/* create output data block and view */
vsip_cblockbind_f(dd_re, dd_im, N, VSIP_MEM_NONE);
vsip_cblockadmit_f(dd_block, VSIP_TRUE);
dd = vsip_cvbind_f(dd_block, 0, 1, N);

/* create FFT specification */
fftccop = vsip_ccfftop_create_f(N, 1.0, VSIP_FFT_FWD, 1, VSIP_ALG_TIME);

/* perform fft */
vsip_ccfftop_f(fftccop, cc, dd);

/* destroy view and release output data to User Space*/
vsip_cvdestroy(dd);
vsip_blockrelease_f(dd_block, VSIP_TRUE);
Spectral analysis demo using VSIPL

Typical signal processing stream used for spectral analysis of a signal.

Basis for many applications: Radar processing, signal intelligence, software radio, voice recognition, vibration detection, medical imaging etc.

- Application framework created by our AXISView graphical tools.
- Data movement between threads using our AXISFlow communications library (API).
- Signal Processing using our AXISLib VSIPL libraries.
Spectral analysis demo - GUIs

Output signal display GUI
Displays processed signal in Freq domain.
Show GFLOPs for FFT & FIR operations.

Signal Generation GUI
Creates signals + noise.
Simulates A/D converter front-end.
VSIPL – Commercial implementations

GE Intelligent Platforms – AXISLib (Intel, PowerPC, NVIDIA GPU)
http://defense.ge-ip.com/products/axislib/p3585

RunTime Computing Solutions – VSI/Pro (Intel, AMD, PowerPC, GPU)
http://www.runtimecomputing.com/products/vsipro/

NASoftware – VSIPL (Intel, PowerPC, MIPs)
http://www.nasoftware.co.uk/home/index.php/products/vsipl

Also, there is an open source reference implementation called TASP:
Image processing demo – Using OpenCV

Optic Flow – CPU vs GPU

Computes optic flow of features identified in image.
Summary

- Abstraction libraries and tools enable developer to build portable, scalable applications.

- The use of industry standard APIs avoid vendor lock-in and facilitate technology refresh.

- Enable building high performance applications that can be deployed across heterogeneous CPU & GPU architectures.

See us at our booth in the Exhibition hall.

Thank you for listening. Questions?