Manycore Processor for Video Mining Applications

Jan. 25th 2013
Yukoh Matsumoto, Hiroyuki Uchida, Michiya Hagimoto, Yasumori Hibi, Sunao Torii, Masamichi Izumida

TOPS Systems Corp.
More Energy-Efficiency required for Next-Generation Embedded Systems

- **Next-Gen Embedded Systems**: Requires more performance (100’s GOPS~1 POPS)
- **Power Consumption**: Already reached upper limit (~W)

Ref: NDEO Technology Roadmap 2009, I-48p, Fig. 1-6

Energy-Efficient Computing goes to Heterogeneous & Manycore

TOPS Systems Corp.  www.topscom.co.jp
“ML and 3D Object Recognition on Image Stream”

Computer Vision (CW) = EYE (sensor) + CEREBRUM (decision)

Why we need Hetero Manycore?
- Conceptually: Machine Learning (ML) ≡ Functional Configuration
- High Perf. Requirement: More than 1 TOPS
- Inherent Parallelism: More than 99% of processing
- Several types of Proc.: Huge cost with Hardwires Implementation
  - Resolutions: VGA, XGA, SGGA, FHD, 2K, 4K, etc.
  - Algorithms: SIFT, Optical Flow, Ransac, Viola & Jones, Model based Recognitions, etc.
  - Others: Multi-Medium Streams (MPEG-2, MPEG-4, H.264, etc.)

Key requirement is High Performance with Flexibility
What is Video Mining System

- Feature Extraction (Low Level)
  - Camera(s)
  - Video (MPEG, H.264, …)
  - Video Stream
  - Motion Stream
  - Audio Stream

- Shot boundary Detection
- Keyword Detection (Mid Level)
  - \( x_2 \), \( x_3 \), …, \( x_n \)
- Event Detection (High Level)
  - Recognized Images, Extracted Images

- Video Decoder + \( \alpha \)
  - Frame
  - Shot
  - Scene
  - Video Sequence
  - Highlight
  - Person
  - Dog

Data Parallel
- Image Stream by frames, Motion Streams (MVs)
- Search target, event detection

Task Parallel
- Recognition Algorithms
- Viola Jones, SIFT, SVM, Matching, optical flow, etc.
- Matching, Editing

- Gaussian Filter, Integral Map, Motion Vectors, Motion Compensation, Entropy
  - Decode, iDCT, de-blocking Filter, etc.

Data Parallel
- \( \sim n \) streams, 2-D image blocks
- Task Parallel

More than 99% of application can be parallelized
## Parallelisms in Algorithms for Video Mining

<table>
<thead>
<tr>
<th>Application</th>
<th>Objective</th>
<th>Algorithm</th>
<th>Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video Analysis</td>
<td>Prediction of Motion Vector</td>
<td>Optical Flow</td>
<td>Line</td>
</tr>
<tr>
<td></td>
<td>Specific feature detection and extraction</td>
<td>SIFT(^1)</td>
<td>Data Partitioning</td>
</tr>
<tr>
<td></td>
<td>Detection of human, and tracking</td>
<td>Cascaded Haar Like</td>
<td>Pixel Level</td>
</tr>
<tr>
<td></td>
<td>Line detection for field separation</td>
<td>Huff</td>
<td>Line</td>
</tr>
<tr>
<td></td>
<td>Elimination of error from continuous frames</td>
<td>Ransac(^2)</td>
<td>Sample Data</td>
</tr>
<tr>
<td>Human Search</td>
<td>Face detection from several angles</td>
<td>Vector Face Detection</td>
<td>Pixel Blocks</td>
</tr>
<tr>
<td></td>
<td>Extraction of features on faces</td>
<td>Model Based Face</td>
<td>Task Level</td>
</tr>
<tr>
<td></td>
<td>Specific feature detection and extraction</td>
<td>SIFT</td>
<td>Data Partitioning</td>
</tr>
<tr>
<td>Video Editing</td>
<td>Segment Extraction</td>
<td>Graph based</td>
<td>Grid Level</td>
</tr>
<tr>
<td></td>
<td>Detection of Motion Vector</td>
<td>Block Matching</td>
<td>Line</td>
</tr>
</tbody>
</table>

*\(^1\) SIFT: Scale-Invariant Feature Transform
*\(^2\) Ransac: Random Sample Consensus

Many type of parallelisms are inherent in algorithms for Video Mining
Goals of SMYLEvideo Manycare

- Real-Time Processing: 1TOPS~
- Scalability: 10fps, 20fps, 30fps
- Programmability: Software Based Implementation
- Flexibility: OpenCV (Computer Vision)
  - SIFT, Optical Flow, Ransac, Viola & Jones,
  - Model based Recognitions, SVM, etc.
- Low Power: ~1.5W
- Low Clock Frequency: ~100MHz

High Performance, Programmable, Scalable

TOPS Systems Corp.  www.topscom.co.jp
System Level Architecture of SMYLEvideo

- **Distributed Processing** with KPN
  - Non-Shared Memory Processes
  - Zero-Overhead Message Passing Mechanism

- **Combination of Parallelisms**
  - Distributed Parallel Processing (Task, Pipeline)
  - Data Parallelism (High-Level, Instruction Level)

- **Stream Processing (Core)**
  - Kernel
  - Stream-In (Read Message)
  - Stream-Out (Write Message)

- **Optimization of Core**
  - Support Stream Processing: background Stream
  - Complex Inst: Reduction of Kernel cycle
  - FIFO support mechanism
  - Reduction of energy for instruction/data supply
SMYLEvideo: Basic Architecture

- **Application Domain Specific Scalable Heterogeneous Manycore**
  - Local Inst. Memory Configuration
  - Loop Buffer Configuration
  - Instruction Extension (Decoder, ALU, LD/ST)
  - Data Register Configuration
  - FIFO Register Configuration
  - Local Data Memory Configuration

**Scalar Processor Core**
- Fetch / Loop Buffer (*)
- Inst. Decoder (*)
- GP Reg File
- Data Reg File (*)
- Interrupt & FIFO Controller
- MUX
- ALU (*)
- LD/ST (*)
- Local Data Mem (*)

**Cluster**
- L1 Instruction Memory (SPM)
- L1 Data Memory (SPM)
- Local I Mem (*)
- (* Configurable)
- Interrupt & FIFO Controller
- MUX
- ALU (*)
- LD/ST (*)
- Local Data Mem (*)

**SMYLE for Video Mining Chip**
- Memory Controller
- L2 Data Memory (SPM)
- I/O

**NoC for data and event communication**
Partitioning
OpenCL vs. Distributed Processing

- **OpenCL (CPU centric)**
  - Bottleneck
    - Processing on Host
    - Increasing communication with Host
  - Hard to express distributed processing

- **Distributed Processing**
  - Scalability
  - Can combine with Data Parallel Processing

Take an approach of Distributed Processing for removing bottlenecks
Software Partitioning
Sequential to Distributed Processing

Issues
- Cannot utilize Multi-Core / Many-Core
- Requires High Performance for Video Mining

Advantages
- Utilize Multi-Core / Many-Core resources
- Easy to balance the load

Investigation has done on Many Algorithms; Viola & Jones, SVM, SIFT, etc.
Goal: High-Performance & Low-Power Programmable Accelerator (Energy-Efficient, Low Cost, Flexible, Scalable)

Approach: Low Clock Frequency

**Power consumption**

\[ P_{\text{Total}} = P_{\text{Dynamic}} + P_{\text{Static}} \]

\[ = \frac{1}{2} \alpha C V^2 f \alpha + I_{\text{Leak}} V \]

- \( C \): Load capacity
- \( V \): Source voltage
- \( f \): Frequency
- \( \alpha \): Switching rate
- \( I_{\text{Leak}} \): Leakage current

**Performance**

Performance = \( \text{OPC} \times f \)

- \( \text{OPC} \): Operation Per Clock

High Performance @ Low Clock Frequency drives Low Power
Approach to reduce clock frequency with Architecture-Algorithm Co-Design

- (1) KPN based distributed processing
- (2) Stream Processing on Stream Processor
- (3) Inter-core communication
- (4) Memory Hierarchy optimization
- (5) Exploit locality by SW optimization
- (6) Increase Operations / clock by Domain Specific Instructions

10fps = $\frac{10M\text{cycle}}{\text{fr}}$
Hide overhead of Stream-In and Stream-Out

- Hide cycles for inter-process communication (Stream-In and Stream-Out)
Inter-Core FIFO : Register Bank Sharing

- Reduction of Memory Access Bandwidth and its Energy

Reduce memory access time and Energy!
Reduction of memory traffic

Path for Message Passing

- Via Mem
- Via Reg

Software to implement FIFO on memory
Software to implement FIFO on register

Total Memory access to Local Memory

- Via
- Via Reg

Significant Reduction of Memory Traffic: more than 30%
ZOMP
Zero-Overhead Message Passing Mechanism

- Remove cycles and memory access for checking FIFO counts and synchronization

<Hardware>
- FIFO Configuration Reg
- FIFO Counter

<Software>
- Prefix Instruction
  - FISYNCh: Check & Wait for Event
  - FOWAITn: Check & Wait for Output
  - FOINT: Event Generation
  - FIINT: Event Generation

No overhead: Just Add Prefix instructions on Stream-In & Stream-Out
Memory Access reduction by Distributed Stream Processing

- **Memory Centric Processing**
  - Each core works data on External Memory
  - Integration of processors and memories

- **Distributed Stream Processing**
  - Core to Core Stream passing
    - On-Chip memory
    - Register Sharing

Increase scalability by reducing memory bandwidth requirement
Frame based vs. Block based Processing

Frame based processing

Input Frame
Gaussian DOG
Feature BIN,MAG
Direction
Feature Values
Matching Voting

DRAM
DRAM
DRAM
DRAM

+0 fr
+1 fr
+2 fr
+3 fr
+4 fr
+5 fr

Block based processing

Input Frame
Gaussian DOG
Feature Extraction
MAG,BIN Generation
Direction
Feature Values
Matching Voting

Block Data
Block Data
Block Data
Block Data
Block Data

+0fr
+1fr
+2fr
+3fr

Smaller latency with Smaller Data

Ex). SIFT
## Frame based vs. Block based Processing

<table>
<thead>
<tr>
<th></th>
<th>Frame based Processing</th>
<th>Block based Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Memory Usage</td>
<td>22Mbytes</td>
<td>3.1Mbytes</td>
</tr>
<tr>
<td>Cluster Local Memory Usage</td>
<td>0.15Mbytes</td>
<td>0.8Mbytes</td>
</tr>
<tr>
<td>Recognition Latency</td>
<td>167mSec(5.1frame)</td>
<td>100mSec(3frame)</td>
</tr>
</tbody>
</table>

### Memory Bandwidth Requirement for SIFT

- **Frame based Processing**: 800 Mbyte/sec
- **Block based Processing**: 100 Mbyte/sec

- **80% Reduction**

Memory Usage: 1/7, Memory Bandwidth Requirement: 1/5
SMYLEvideo Configuration

Cluster 0

L1 Instruction Memory (SPM)

QVP C0 → QVP C1 → QVP C2 → QVP C3 → SVP C4 → SVP C5

L1 Data Memory (SPM)

Cluster 1

L1 Instruction Memory (SPM)

SVP C0 → SVP C1 → QVP C2 → QVP C3

L1 Data Memory (SPM)

10 cores (SIFT: 15fps)

SMYLE for Video Mining Chip

CPU ↔ Mem Ctrl

L2 Data Memory (SPM)

I/O

NoC for data and event communication

QVP : Quad V processor (256-bit core)
SVP : Single V processor (64-bit core)

Scalable Performance and Functionality with adding Clusters

TOPS Systems Corp.  www.topscom.co.jp
Conclusions

- **Manycore** will play a crucial role in extending the roadmap for enabling the next generation **SoCs** required for “**Video Mining**” one of Computer Vision systems.

- **Zero-Overhead Message Passing Mechanism (ZOMP)** can efficiently increases the system performance and scalability of Manycore processors.

- **Block based distributed processing** drastically reduces memory access bandwidth and increases room for higher performance on Manycore processors.

- **SMYLEvideo** provides scalability in performance and functionality with its clustered architecture.