

# **CE11: Parallelization of HHI\_TRANSFORM\_CODING (Fixed Diagonal Scan) (JCTVC-F128)**

**Vivienne Sze, Madhukar Budagavi**

**Texas Instruments Inc.**

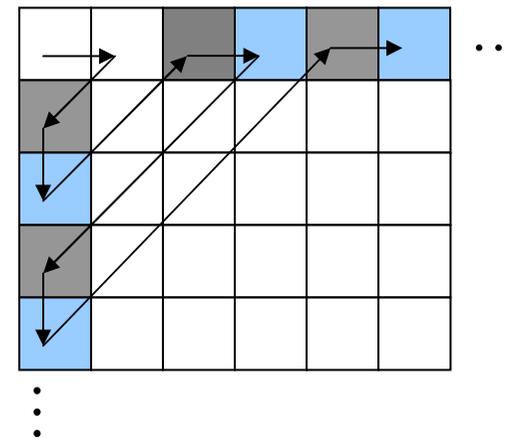
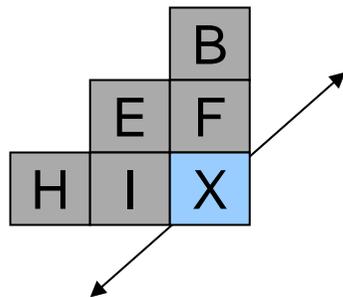
**Joint Collaborative Team on Video Coding (JCT-VC)  
of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**

**6th Meeting: Torino, IT, 14-22 July, 2011**

# Motivation

- Increase throughput of CABAC while maintaining high coding efficiency
- For parallel multi-bin decoding, minimized dependency across bins
  - Dependencies require speculative computations
- For transforms larger than 8x8
  - use zig-zag scan
  - context selection depends on neighbors

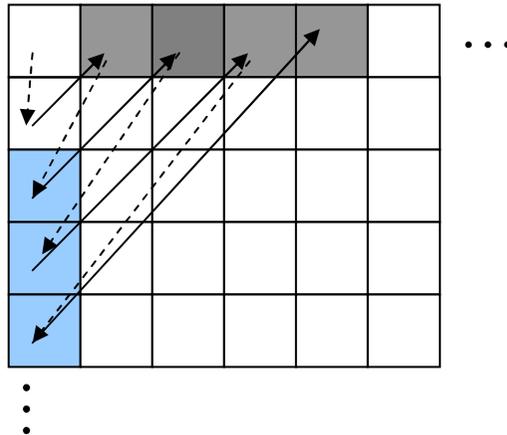
JCTVC-D260 (adopted in HM-2.0) – eliminated diagonal dependencies; context selection of X (blue), depends on neighboring values (grey)



Process blue and grey in parallel at edges; remove dependency on most recently processed neighbor

# Reduce Dependency

- For CABAC, remove dependency on previously decoded bin by processing with fixed diagonal scan



# Experiment Results

- HM-3.0 under common conditions
- Simulation platform is LSF equipped with Intel(R) Xeon(R) CPU X5570@2.93GHz 64 bits Linux machines
- Results cross-checked by I2R (F149), Qualcomm (F293), Sony (F134)

## Coding efficiency impact for High Efficiency

Intra	Random Access	Low Delay
-0.1	-0.1	0.0

# Conclusions

- Use diagonal fixed diagonal scan rather than zig-zag for CABAC at edges of transform to eliminate dependency on previously decoded bin
  - Reduce speculation required for parallel processing
- Coding efficiency impact of -0.1%
- Recommend for adoption into HEVC test model
  - Draft text available in contribution

# Outline

- Introduction
  - Process multiple bins in parallel for increased throughput
    - Need to minimize dependency between bins that will be processed in parallel.
  - Current context selection method for higher coding efficiency
    - Dependency between bins are edges of transform
- Approach
  - At edge of transform, reduce dependency
  - Helps even when more than 2+ bins in parallel (delay exponential growth)
- Coding Efficiency
- Conclusion