High Efficient Architecture for 3D-HEVC DMM-1 Decoder Targeting 1080p Videos

[1,2]Gustavo Sanchez, [3]Luciano Agostini, and [1]César Marcon
[1]*Pontifical Catholic University of Rio Grande do Sul – Porto Alegre, Brazil*
[2]*IF Farroupilha – Alegrete, Brazil*
[3]*Video Technology Research Group (ViTech) – Federal University of Pelotas (UFPel) – Pelotas, Brazil*
gfsanchez@acad.pucrs.br, agostini@inf.ufpel.edu.br, cesar.marcon@pucrs.br

***Abstract*— This paper presents an efficient hardware design for the Depth Modeling Mode 1 (DMM-1) decoder of the 3D-High Efficiency Video Coding (3D-HEVC). The designed architecture uses a lossless wedgelet memory compression technique to reduce the used memory, and a well-balanced parallelism level to allow the desired throughput at the minimum possible power consumption and area usage. The architecture was synthesized for the 65nm ST standard cells technology, using 4,047 gates and consuming 0.95mW. It is capable of processing 1080p videos at 30 frames per second, decoding all allowed block sizes and wedgelets patterns. Besides, the proposed architecture saves 10.7% of area and 29.1% of power when compared with a version without memory compression. At the best of the author's knowledge, this is the first work with a dedicated hardware design targeting the DMM-1 decoder.**

Keywords—Depth Modeling Modes, 3D-HEVC Decoding, Memory Aware, Hardware Design.

I. INTRODUCTION

The 3D High Efficiency Video Coding (3D-HEVC) [1] has provided several benefits for 3D video coding, and the major part of these benefits comes from the usage of Multiview Video plus Depth (MVD) [2] data format. MVD associates a depth map to each texture view, which must be together encoded by a 3D-HEVC encoder. At the decoding side, view synthesis techniques allow the interpolation of texture views based on depth maps information and the generation of new high-quality virtual texture views located between the original views [3].

An example of a texture view and its associated depth map extracted from MicroWorld video sequence [4] is presented in Fig. 1(a) and Fig. 1(b), respectively. Notice that the depth map contains different characteristics than the texture view. In the texture view, there are smooth transitions between its pixels, while in depth map there are wide areas with homogeneous pixels values (objects bodies and background), and areas with a sharp change in pixel values (objects borders).

Since depth maps have distinct characteristics compared with texture components, different encoding tools have been created to codify depth maps with higher efficiency. Depth Modeling Modes (DMMs) [5], Segment-Wise Direct Component Coding (SDC) [6] and Depth Intra Skip (DIS) [7] are examples of new coding tools used at depth maps intra-frame prediction. The combination of these encoding tools is capable of achieving a reduction in the bitrate necessary to encode homogeneous regions and preserving the depth maps edges, which are crucial to generate high-quality virtual views.

Considering that 3D-HEVC is a relatively new HEVC extension and that the hardware design for depth maps coding has only been superficially explored, this work presents the design of a DMM-1 decoding hardware. All hardware designs found in the literature for the new 3D-HEVC depth maps coding tools are focusing on the encoder side. Sanchez et al. [8] and Amish et al. [9] designed a DMMs architecture for the encoding process. A DMM-4 encoder architecture was designed in [10] and a DMM-1 encoder architecture was designed in [11]. Afonso et al. [12] designed an encoder architecture for DIS. At the best of our knowledge, this is the first work presenting a decoding architecture for these new depth maps coding tools.

The presented architecture uses a technique proposed in our previous work [13] that reduces the DMM-1 patterns memory requirement, saving area and power. The architecture was also designed with a balanced parallelism level to reach the desired processing rate at the lowest possible cost in area and power, and requiring a low bandwidth.

Fig. 1. (a) Texture view and its associated (b) depth map extracted from *MicroWorld* video sequence [4].

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

II. DMM-1 Decoding Algorithm

The DMM-1 decoding algorithm is more straightforward than the encoding one. Fig. 2 exemplifies the decoding of a 4×4 depth block. For a given block size, the input of the DMM-1 decoding algorithm requires the number of the selected pattern, the Constant Partition Value (CPV) of each region and the residual block.

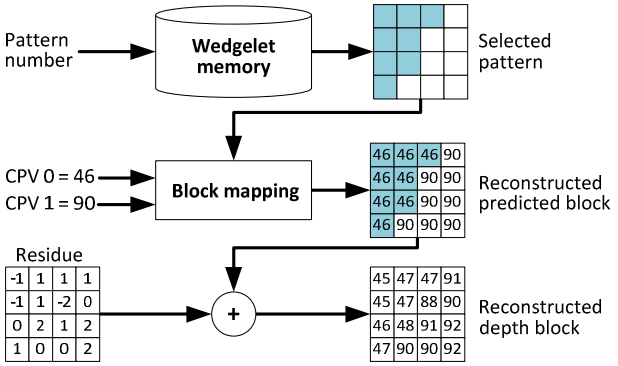

Fig. 2. Example of a 4×4 depth block decoding with the DMM-1 algorithm.

While the encoding process requires the evaluation of the entire wedgelet memory (or at least the initial set), the decoding process only requires the access of the wedgelet indexed by the received pattern number, retrieving the DMM-1 pattern that was selected in the encoding process. The CPV of each region is mapped into this DMM-1 pattern to reconstruct the predicted block. Finally, the residues are added to the predicted block, generating the reconstructed depth block.

A. Delta CPV Computation

The DMM-1 algorithm can produce CPVs that require several bits for storage and transmission. The 3D-HEVC standard adopted the delta CPV, which computes the predicted CPV using the information of the neighbor pixels, to mitigate this problem. Fig. 3 shows the samples used in delta CPV computation, where the encoding block is detached in red.

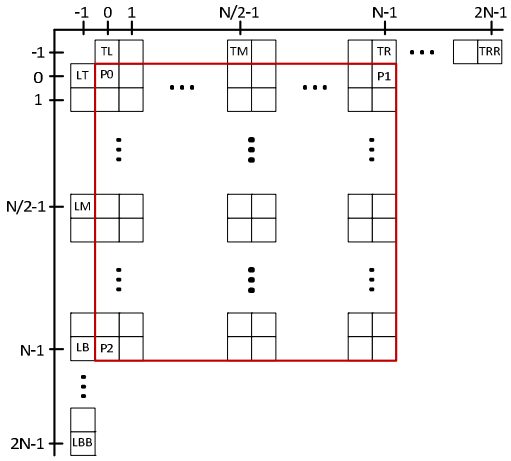

Fig. 3. Encoding block and its boundaries samples.

The following information is required to obtain the predicted CPV: (i) the binary pattern information of the three corners positions (P0, P1, and P2); (ii) three samples of the upper Prediction Unit (PU) block (TL, TM, and TR); (iii) one sample from the upper right PU (TRR); (iv) three samples from the left PU (LT, LM, and LB); and (v) one sample from the left bottom PU (LBB). When some of these samples are not available (i.e., the necessary neighbor pixel has not been codified yet), the nearest available sample is used instead of that.

Fig. 4 illustrates the pseudo-code for delta CPV generation, which uses two Boolean variables bT and bL to determine if the pattern corner samples P0 and P1, and P0 and P2 are different, respectively. Subsequently, the algorithm generates ref_0 and ref_1 and assigns these references to the correct prediction CPV (i.e., pred_0 and pred_1 in the pseudo-code).

```
1.  bT ← P0 ≠ P1 ? TRUE : FALSE
2.  bL ← P0 ≠ P2 ? TRUE : FALSE
3.  if bT = bL
4.      ref_0 ← average(TL, LT)          → Cases 1 and 2
5.      if bL = 1
6.          ref_1 ← average(LB, TR)      → Case 2
7.      else
8.          if abs(TRR – TL) > abs(LBB – LB)
9.              ref_1 ← TRR              → Case 1a
10.         else
11.             ref_1 ← LBB              → Case 1b
12. else                                 → Case 3
13.     ref_0 ← bL ? TM : LM
14.     ref_1 ← bL ? LB : LR             → Case 4
15. pred_0 ← bL ? ref_1 : ref_0
16. pred_1 ← bL ? ref_0 : ref_1
```

Fig. 4. Pseudo-code for delta CPV computation.

Case 1 is defined when the binary patterns are equal; then, ref_0 is computed using the average value of TL and LT. In this case, ref_1 is obtained from LBB or TRR, according to the highest absolute difference between TRR and TL or LBB and LB; only Case 1 uses LBB and TRR. Case 2 occurs when the binary pattern of P0 differs from the binary patterns of P1 and P2. Therefore, ref_0 is also obtained from the average of TL and LT, while ref_1 is obtained averaging TR and LB. Case 3 arises when P0 and P1 have the same pattern, which differs from the one of the P2. In this case, TM is selected as ref_0 and LB as ref_1. Finally, Case 4 occurs when P0 and P2 have the same pattern, and P1 contains a different one. This last case selects LM as ref_0 and TR as ref_1.

After computing the predicted CPV, the original CPV is subtracted from the predicted CPV, generating the delta CPV. Then, only delta CPV should be transmitted in the bitstream. In the decoding process, the predicted CPV can be computed using the same algorithm employed in the encoding process. It requires accessing the wedgelet memory to obtain P0, P1, and P2 values, to request the same sample values of neighbor samples used by the encoder. The predicted CPV is added with the delta CPV, recovering the original CPV.

B. Efficient DMM-1 Patterns Storage

The works [8], [13], and [14] propose techniques to reduce the wedgelet memory requirements. Shuying et al. [14] propose to store only the 16×16 patterns and dynamically create some of the remaining patterns. Sanchez et al. [8] designed a DMMs encoder architecture and removed the DMM-1 patterns obtained in the refinement. The solution of these two works cannot be adapted to the DMM-1 decoder since both solutions eliminate some patterns in their memory and then, if one of

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

these patterns are necessary, the decoder will not be able to retrieve it. Our previous work [13] proposed four lossless pattern storage techniques focusing on reducing the DMM-1 storage requirements. Since the decoder must access random positions inside the pattern memory, only two of the proposed solutions by our previous work can be applied to the decoder, namely: First Bit and Change (FB&C) and Huffman Coding.

FB&C can achieve similar results than Huffman Coding and is much simpler to implement; consequently, FB&C was selected to be implemented in the architecture designed in this work. FB&C relies on the fact that DMM-1 divides each block into two and only two regions. Thus, DMM-1 does not represent interlaced ones and zeros in a single line leading to an inefficient storage model if $N \times N$ bits are stored per pattern. Fig. 5 illustrates the FB&C technique that codifies each line of the pattern copying the 1st bit content of the original line to the coded one. The remaining of the encoded line contains the position that there is a bit change. FB&C can save 63.2% of the memory requirement (from 180 Kbits to 66 Kbits).

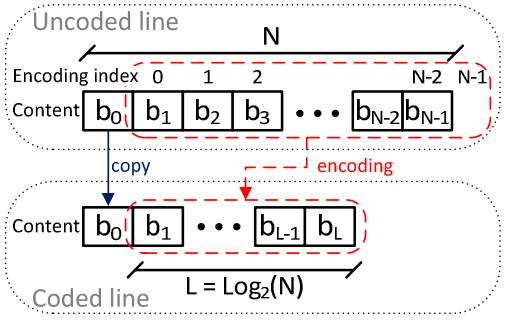

Fig. 5. FB&C encoding model.

III. Design of the DMM-1 Decoder Architecture

Fig. 6 shows the high-level block diagram of the designed DMM-1 decoder architecture, which contains six main modules: DC Reconstruction (rebuilds the original CPVs), Register bank, Block Reconstruction, Wedgelet Decision, Wedgelet Memories, and Control. The architecture has a low input bandwidth, requiring at most only 36 bits per cycle, which is interesting to create a low consumption decoder. The architecture also has a parallelism level adjusted to process 1080p @ 30fps minimizing the area usage and the power consumption.

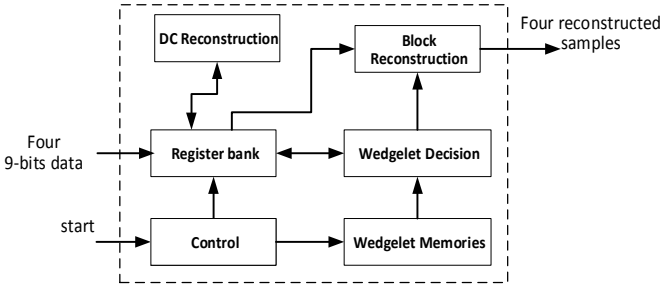

Fig. 6. High-level block diagram of the DMM-1 decoder.

The architecture inputs were limited to four parallel 9-bits input samples, which are stored in the register bank before being used by the other architecture modules. This limitation was performed focusing on achieving a low-bandwidth communication with the main memory of the decoder, which can help to increase the entire decoder efficiency.

Fig. 7 shows that these input samples bring a different kind of information according to the clock cycle. The start bit raises during a single clock cycle to signalize the decoder that it should start processing. In the next clock cycle, the start bit falls and the information of deltaDC_0, deltaDC_1, block size (blocksize) and the pattern number (pattern_num) are delivered to the decoder. Next clock cycle, TL, TM, TR, and TRR, which are the samples placed above the current block required for DC prediction, are delivered to the decoder. During this same cycle, the Wedgelet Memories are accessed bringing the information of P0 and P1 (i.e., the upper corners of current block pattern) and stored it in the Register bank. Next clock cycle, LT, LM, LB, and LBB are delivered to the decoder and P2 (the lower left corner pattern) is stored in the Register bank.

As all information required by DC reconstruction module is already available in the register bank, during the next clock cycle this module computes the reconstructed DCs, delivering this information for the Register bank. Finally, at each clock cycle, the module receives four samples of the residue and compute the reconstructed four samples by accessing the corresponding position in the Wedgelet Memories.

A. DC Reconstruction Module

Fig. 8 depicts the block diagram of the DC Reconstruction architecture that implements the algorithm described in Fig. 4. All data required in this process (i.e., P0, P1, P2, DELTA_DC0, DELTA_DC1, TL, TM, TR, TRR, LT, LM, LB, LBB) are previously stored in the decoder Register bank. This module takes one clock cycle to generate DC_REC0 and DC_REC1, representing the first DC computed by the encoding process.

B. Block Reconstruction Module

Fig. 9 presents the Block Reconstruction architecture, which receives, at each cycle, four samples of residues from the Register bank and the binary pattern of those positions from the Wedgelet Memories. Moreover, this architecture receives a fixed value for DC_REC0 and DC_REC1 that were stored in the Register bank after the DC Reconstruction step.

According to the binary pattern of each sample, the architecture decides through a multiplexer if DC_REC0 or DC_REC1 should be added with the residues, delivering the reconstructed sample to the output of the decoder.

IV. Synthesis, Results and Discussion

The designed architecture was synthesized with the traditional wedgelet storing approach (i.e., storing all binary information) and using FB&C to achieve a considerable memory reduction. Table I shows the synthesis results for standard cells 65nm ST technology.

Table I. Synthesis results of the DMM-1 Decoder.

Solution	Traditional	FB&C
Area (gates)	4,533	4,047
Decoding block sizes	All	All
Frequency (MHz)	38.9	38.9
Cycles per block	10/22/70/262	10/22/70/262
Processing rate HD 1080p fps	30.0	30.0
Power (mW)	1.34	0.95

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

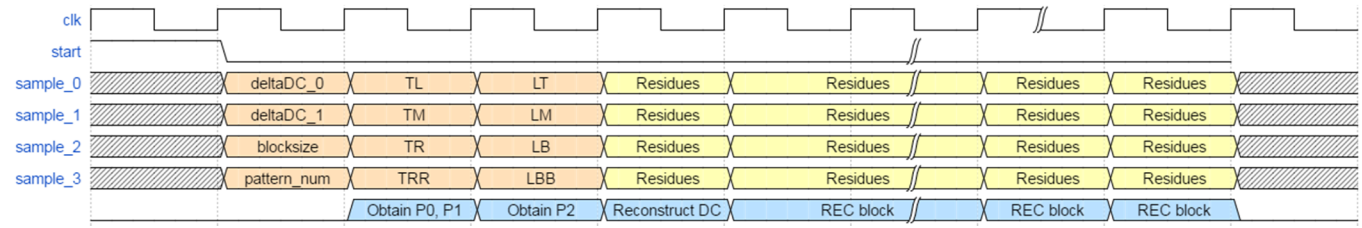

Fig. 7. Input information of the DMM-1 decoder

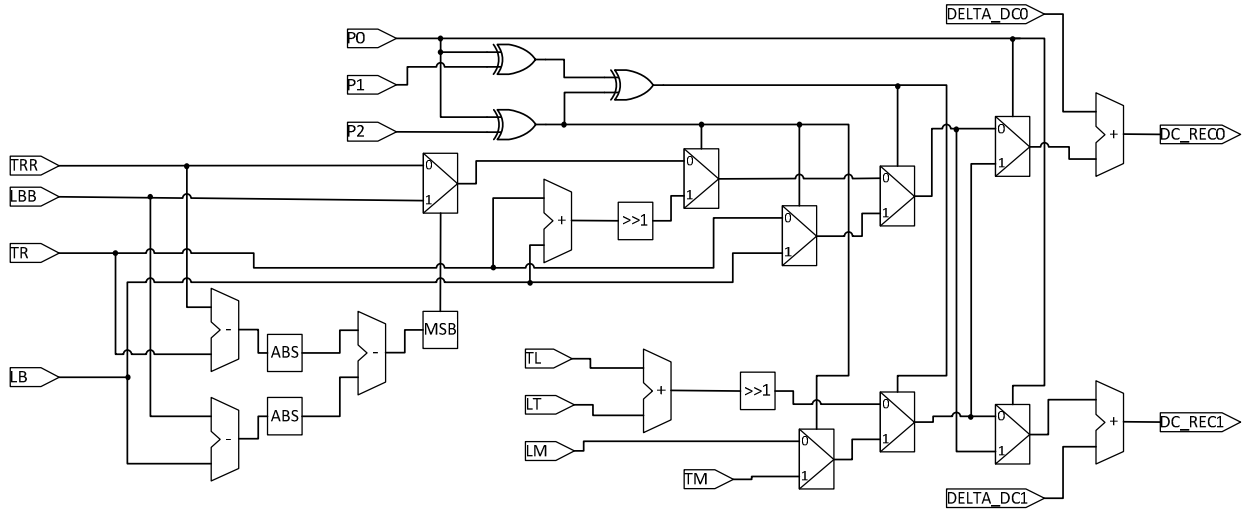

Fig. 8. DC Reconstruction architecture.

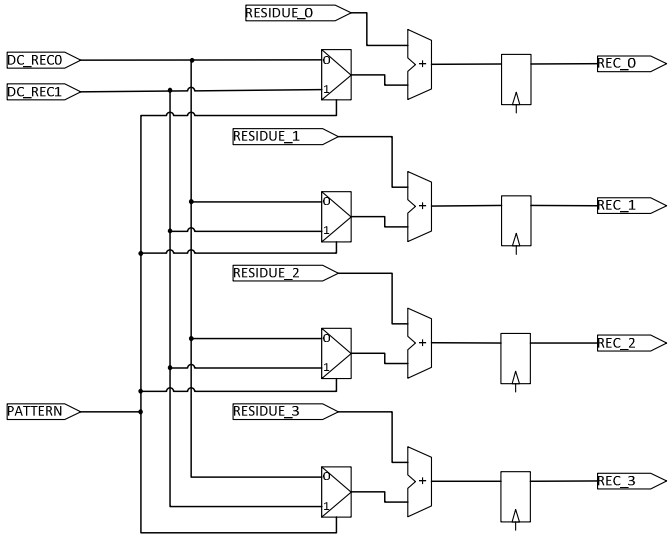

Fig. 9. Block Reconstruction architecture.

The DMM-1 decoder architecture used 4,047 gates and reached 0.95mW of power consumption at 38.9MHz. This architecture can process 1080p frames at 30 frames per second.

The DMM-1 architecture processing rate was calculated considering the maximum possible number of DMM-1 blocks inside a 1080p frame. Notice that this processing rate is the worst case and it is much higher than that necessary since DMM-1 block selection is shared with other modules, such as HEVC intra-frame prediction and inter prediction.

The DMM-1 decoder with traditional memory approach consumes 4,533 gates and dissipates 1.34 mW of power. When applying FB&C, the architecture area and power dissipation are reduced in 10.7% and 29.1%, respectively. This result demonstrates that reducing the area for the wedgelet memory brings significant benefits for hardware design regarding area occupation and power dissipation.

V. Conclusions

This paper presented an efficient hardware design for the 3D-HEVC DMM-1 decoder. The architecture used a memory-aware technique that reduced 63.2% of the required memory. Besides, the hardware was designed to decode 30 1080p frames per second using the minimum possible number of gates and the minimum possible I/O bandwidth (36 bits per cycle). The design strategies allowed the architecture to reach a power consumption of 0.95 mW when running at 38.9 MHz. The results also showed that the used memory compression caused a reduction of 10.7% in area and 29.1% in power dissipation when compared to a DMM-1 decoder without memory compression.

Acknowledgment

This paper was achieved in cooperation with Hewlett-Packard Brazil Ltda. using incentives of Brazilian Informatics Law (Law nº 8.248 of 1991). Authors also would like to thanks CAPES (processes 88881135737/2016-01 and 88881119481/2016-01), CNPq and FAPERGS Brazilian research agencies to support the development of this work.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

REFERENCES

[1] G. Tech, Y. Chen, K. Muller, J. Ohm, A. Vetro, Y. Wang. "Overview of the Multiview and 3D extensions of High Efficiency Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, v. 26, n. 1, pp. 35-49, Jan. 2016.

[2] K. Muller, P. Merkle, T. Wiegand. "3D Video Representation using Depth Maps," *Proceedings of the IEEE - Special Issue 3D Media Displays*, v. 99, n. 4, pp. 643-656, Apr. 2011.

[3] P. Kauff, N. Atzpadin, C. Fehn, M. Muller, O. Schreer, A. Smolic, R. Tanger. "Depth Map Creation and Image-based Rendering for Advanced 3DTV Services Providing Interoperability and Scalability," *Image Communication*, v. 22, n. 2, pp. 217-234, Feb. 2007.

[4] Mitsubish Electric Research Laboratiores. Available at: ftp://ftp.merl.com/pub/tian/NICT-3D/, access in Nov. 2016.

[5] P. Merkle, K. Muller, D. Marpe, T. Wiegand, "Depth Intra Coding for 3D Video based on Geometric Primitives," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, v. 26, n. 3, pp. 570-582, Feb. 2015.

[6] H. Liu, Y. Chen, "Generic segment-wise DC for 3D-HEVC depth intra coding," in *Proc. IEEE International Conference on Image Processing (ICIP)*, pp. 3219-3222, 2014.

[7] J. Lee, M. Park, C. Kim, "*3D-CE1: Depth Intra Skip (DIS) Mode*," document JCT3V-K0033, Geneva, Switzerland, Feb. 2015.

[8] G. Sanchez, C. Marcon, L. Agostini. "Real-Time Scalable Architecture for 3D-HEVC Bipartition Modes," *Journal of Real-Time Image Processing (JRTIP)*, v. 13, n. 1, pp 71-83, Mar. 2017.

[9] F. Amish, E. Bourennane. "An Efficient Hardware Solution for 3D-HEVC Intra-Prediction," *Journal of Real-Time Image Processing (JRTIP)*, pp. 1-13, Jan. 2017.

[10] G. Sanchez, B. Zatt, M. Porto, L. Agostini. "A Real-time 5-views HD 1080p Architecture for 3D-HEVC Depth Modeling Mode 4," *Symposium on Integrated Circuits and Systems (SBCCI)*, pp. 1-6, Sep. 2014.

[11] G. Sanchez, L. Agostini, F. Mór, C. Marcon. "Low-area scalable hardware architecture for DMM-1 encoder of 3D-HEVC video coding standard," *Symposium on Integrated Circuits and Systems (SBCCI)*, pp. 36-40, Sep. 2017.

[12] V. Afonso, L. Audibert, M. Saldanha, R. Conceição, A. Susin, M. Porto, B. Zatt, L. Agostini. "Low-Power and High-Throughput Hardware Design for the 3D-HEVC Depth Intra Skip," *International Symposium on Circuits and Systems (ISCAS)*, pp. 1-4, 2017.

[13] G. Sanchez, C. Marcon, L. Agostini. "Energy-aware Light-weight DMM-1 Patterns Decoders with Efficiently Storage in 3D-HEVC," *Proceedings of the Symposium on Integrated Circuits and Systems (SBCCI)*, pp. 1-6, 2016.

[14] M. Shuying, Y. Wang, C. Zhu, Y. Lin, J. Zheng. "Reducing Wedgelet Lookup Table Size with Down-sampling for Depth Map Coding in 3D-HEVC," *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1-5, Oct. 2015.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE