Novel VLSI Implementation of Peano-Hilbert Curve Address Generator

Yan Wang, Shoushun Chen and Amine Bermak
Smart Sensory Integrated Systems Lab
Electronic and Computer Engineering Department
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong, SAR.
Email: wongyin,eechenss,eebermak@ust.hk

***Abstract*— This paper presents a fast algorithm for generating Hilbert address for hardware implementation with low storage requirement. This work avoids the use of recursive functions as compared with Quinqueton's work, and eliminates complicated bit manipulations as proposed by Butz, and does not use any look-up-tables as implemented by Kamata. Each address can be obtained in one clock cycle by one-to-one mapping using a simple incremental counter and cascading of multiplexers. The merit of our method is that it achieves very high speed when computing the Hilbert address which requires little memory storage.**

I. Introduction

An Italian mathematician G. Peano discovered a series of curves that can continuously traverse all points in a space [1]. Later, Hilbert [2] worked out one of the simplest curves in two-dimensional space, which is now the so called *Hilbert curve*. This curve is widely used in image analysis as well as image compression applications [3] [4] [5] [6] [7] as the scanning path. There are quite a few algorithms for N-dimensional Hilbert scanning, for example, the Quinqueton [8] algorithm and the Butz [9] algorithm, to name a few. The Quinqueton algorithm uses recursive functions to compute all the addresses for the Hilbert scanning path. However, in general, recursive function is usually computationally expensive and not easy to be implemented in hardware. By using this algorithm, all the Hilbert addresses should be precalculated and stored in memory for hardware application, which is not an economical solution. In contrast, the Butz algorithm uses quite complicated bit manipulations such as shifting, exclusive OR to get the address in real time. Although the addresses of the Hilbert scanning path is not obtained by recursive functions, the Butz algorithm itself is already too complex to be hardware friendly and it cannot generate the address in just one clock cycle. Kamata [10] proposed a new algorithm to generate the addresses of the Hilbert scanning path by the use of look-up tables. However, this algorithm consumes huge amount of memory to generate all the addresses, which is not suitable for hardware implementation.

In this paper, we propose a novel method for generating Hilbert scanning address in hardware. This technique is a direct one-to-one mapping of the output of an incremental counter to the Hilbert scanning address. Each address can be obtained in just one clock cycle using the proposed method. The prototype chip for Hilbert address generation is fabricated.

The remainder of this paper is organized as follows. Section II presents the analysis of Hilbert scanning. Section III detailed and discussed the implementation of Hilbert address generation system. Section IV concludes this work.

II. Hilbert Scanning Analysis

A typical sized square image can be decomposed into different levels of quadrants through Hilbert scanning with different resolutions. The length and the width of the square image are usually a power of two. Figure 1 illustrates an 8×8 image that is decomposed into different levels of hierarchies with Hilbert scanning in the resolution of 2×2, 4×4, and 8×8. From this figure, it is clear that when the resolution gets larger and larger, the basic scanning pattern replicates and rotates in a certain manner. By examining the addresses

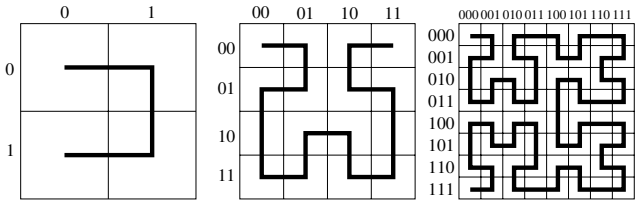

Fig. 1. Hilbert Scan of a square image with different resolution: (a) 2×2, (b) 4×4, and (c) 8×8.

of the Hilbert scanning path, the construction rule for Hilbert address generation can be generalized.

The addresses of the 2×2 image are as follows: 00→01→11→10. The addresses of the 4×4 image are as follows: 00·[00→01→11→10] → 10·[00→10→11→01] → 11·[00→10→11→01] → 01·[11→10→00→01] where the operator · indicates concatenation operation. The x- and y-coordinates of the physical address can be obtained by re-grouping the odd and even bits of the above presented address. For example, from the address 01·10 ⇔ ***0*1*1*0**, it is easy to get the x- and y- coordinates by selecting the odd and even bits respectively. Therefore the x-coordinate is *01* and the y-coordinate is **10**.

Following the same rule, the addresses of the 8×8 image can be represented as: 00·{00·[00→01→11→10]} →

$00\cdot\{10\cdot[00\rightarrow10\rightarrow11\rightarrow01]\} \rightarrow 00\cdot\{11\cdot[00\rightarrow10\rightarrow11\rightarrow01]\} \rightarrow$
$00\cdot\{01\cdot[11\rightarrow10\rightarrow00\rightarrow01]\} \rightarrow 01\cdot\{00\cdot[00\rightarrow10\rightarrow11\rightarrow01]\} \rightarrow$
$01\cdot\{01\cdot[00\rightarrow01\rightarrow11\rightarrow10]\} \rightarrow 01\cdot\{11\cdot[00\rightarrow01\rightarrow11\rightarrow10]\} \rightarrow$
$01\cdot\{10\cdot[11\rightarrow01\rightarrow00\rightarrow10]\} \rightarrow 11\cdot\{00\cdot[00\rightarrow10\rightarrow11\rightarrow01]\} \rightarrow$
$11\cdot\{01\cdot[00\rightarrow01\rightarrow11\rightarrow10]\} \rightarrow 11\cdot\{11\cdot[00\rightarrow01\rightarrow11\rightarrow10]\} \rightarrow$
$11\cdot\{10\cdot[11\rightarrow01\rightarrow00\rightarrow10]\} \rightarrow 10\cdot\{11\cdot[11\rightarrow10\rightarrow00\rightarrow01]\} \rightarrow$
$10\cdot\{01\cdot[11\rightarrow01\rightarrow00\rightarrow10]\} \rightarrow 10\cdot\{00\cdot[11\rightarrow01\rightarrow00\rightarrow10]\} \rightarrow$
$10\cdot\{10\cdot[00\rightarrow01\rightarrow11\rightarrow10]\}$.

From the addresses listed above, the underlining rules of the Hilbert scanning path can be revealed. This space filling curve is constructed in a hierarchical and quadrant based manner. The traverse of the Hilbert curve in different hierarchies and in different quadrants in the same hierarchy follows different patterns, as can be seen from Figure 2. The most significant bits of the address reveal the pattern of traverse in the highest layer while the least significant bits reveal that of the lowest layer of quadrant. For the case of 8×8 image, the most significant two bits change in the sequence of 00→01→11→10, which is represented as a bold solid line in Figure 2, where the thin solid line represents the track of the middle hierarchy and the change in the middle two bits of the address, while the dashed line represents the scanning path in the bottom layer and the change in the least significant two bits.

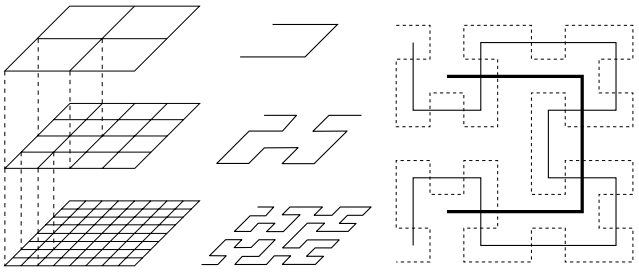

Fig. 2. Hierarchical Structure of the Hilbert Scan.

In order to analyze the address generation process for Hilbert scanning path , it is better to define the basic scans as the elements of this space filling curve. An image is defined as I_n, which is in the size of $n \times n$, where $n = 2^k$ and $k \geq 1$. This defined image I_n is a square image thus its length and width are equal; and both of them are in the size of a power of two. Therefore, this image can be decomposed into multiple levels of hierarchies.

Let $I_n(0,0)$, $I_n(0,1)$, $I_n(1,1)$, and $I_n(1,0)$ represent the upper left, upper right, lower right, and lower left quadrants of I_n. The basic scans of Hilbert curve are denoted as RR, -RR, -CC, and CC respectively, they are defined as follows and are illustrated in Figure 3:

RR: $(0,0) \rightarrow (0,1) \rightarrow (1,1) \rightarrow (1,0)$,
-RR: $(1,1) \rightarrow (1,0) \rightarrow (0,0) \rightarrow (0,1)$,
-CC: $(1,1) \rightarrow (0,1) \rightarrow (0,0) \rightarrow (1,0)$,
CC: $(0,0) \rightarrow (1,0) \rightarrow (1,1) \rightarrow (0,1)$.

The name RR, -RR, -CC, and CC indicate the direction of movement of the scanning path from its 2^{nd} position to its 3^{rd} position in a quadrant. RR means positive row-to-row movement, while -RR means negative row-to-row movement.

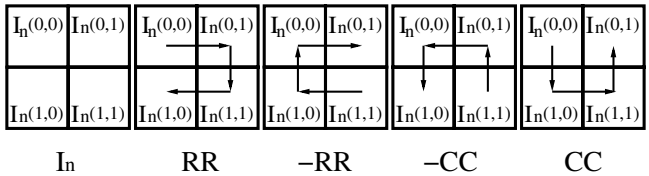

Fig. 3. Square Image I_n and its four basic scans: RR, -RR, -CC, and CC.

Similarly, CC and -CC indicate positive and negative column-to-column movement, respectively. The notation of (a,b) indicates that the x- and y-coordinates are a and b, respectively, where a and b are binary integers. These four basic scans are the key elements for constructing Hilbert space filling curve and are discovered by exhaustive search in this curve.

The relationship between the upper and lower hierarchies are also found through comprehensive analysis; this is illustrated in Figure 4. When the scanning in the upper layer hierarchy is RR, the scans in the lower hierarchy are CC, RR, RR, and -CC for each quadrant, respectively. When the scanning in the upper layer hierarchy is -RR, the scans in the lower hierarchy are -CC, -RR, -RR, and CC for each quadrant, respectively. When the scanning in the upper layer hierarchy is -CC, the scans in the lower hierarchy are -RR, -CC, -CC, and RR for each quadrant, respectively. When the scanning in the upper layer hierarchy is CC, the scans in the lower hierarchy are RR, CC, CC, and -RR for each quadrant, respectively.

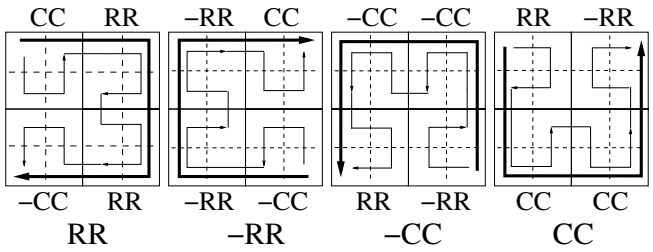

Fig. 4. Standard Recursive Scanning.

One interesting feature of the Hilbert scan is that when the total number of levels of hierarchy is odd number, the scanning in the highest level is **RR**. While if the number of levels are even, the scanning in the highest level is **CC**. The number of levels is directly related with the size of the square image. When the size of an image is $2^k \times 2^k$, the value k is equal to the number of levels of hierarchy. This rule does not always need to be satisfied if the starting point of the Hilbert curve is not fixed to the up and left corner of an image. However, it is included in our generalized construction rules to facilitate Hilbert address generation for CMOS image sensor applications, where the starting point of a pixel array is always set to be up and left corner by convention.

To conclude, Hilbert space filling curve has the following characteristics that govern the construction of its scanning path. These are the generalized Hilbert curve construction rules.

1) When $k = 1$, the four basic scans are defined:

Name	Basic Scan Path
RR:	$(0,0) \rightarrow (0,1) \rightarrow (1,1) \rightarrow (1,0)$
-RR:	$(1,1) \rightarrow (1,0) \rightarrow (0,0) \rightarrow (0,1)$
-CC:	$(1,1) \rightarrow (0,1) \rightarrow (0,0) \rightarrow (1,0)$
CC:	$(0,0) \rightarrow (1,0) \rightarrow (1,1) \rightarrow (0,1)$

2) When $k > 1$, the scan in the highest level is:

k	Scan mode in the Highest level
k is odd	RR
k is even	CC

3) The relationship between the upper and lower scan mode:

Upper Layer Scan Mode	Lower Layer Scan Mode
RR	CC→RR→RR→-CC
-RR	-CC→-RR→-RR→CC
-CC	-RR→-CC→-CC→RR
CC	RR→CC→CC→-RR

From the construction rules of the Hilbert scanning path described above, it can be easily seen that when the size of a square image is known, the number of levels of hierarchy could be easily calculated and thus the scan mode of the highest hierarchy can also be determined. While the scan mode of the highest hierarchy is established, all the scan modes of the lower hierarchies will be fixed. That means all the Hilbert address information is predetermined and can be obtained through propagating this construction process.

III. Hilbert Address Generation Implementation

The Hilbert Address Generation Algorithm should comply with the construction rules of the Hilbert Space Filling Curve. For Hardware implementation, Image size is always predetermined, therefore the circuitry for Hilbert address generation can be tailor-made. The process to obtain the Hilbert scanning address is synchronous with the output of an incremental counter in our hardware implementation. If the scan mode of the current hierarchy is known, by looking at the counter output, the Hilbert address of the current hierarchy and the scan mode of the lower hierarchy can be determined by the iterative construction process of the Hilbert scan.

Each address along the Hilbert space filling curve can be generated in one clock cycle through using a simple incremental counter and multiplexers. The basic scan modes RR, -RR, -CC, and CC are represented by 00, 01, 10, and 11, respectively. The process of Hilbert address generation for a 4×4 image is illustrated here as an example. Here the value of k is equal to $\sqrt{4} = 2$, which is an even number. There are totally 16 addresses that require 4 bits to fully represent them. Therefore, a 4-bit counter Cnt[3:0] is used to generate the address. According to the construction rules, the scan mode in the highest hierarchy is CC. The four consecutive scan mode in the lower hierarchy are RR, CC, CC, and -RR in each quadrant respectively. The most significant two bits of the counter output Cnt[3:2] is used to indicate in which quadrant and by which mode the image is being scanned. When the Cnt[3:2] is equal to 00, the lower hierarchy scanning mode is RR and the most significant two bits of the output Hilbert address Add[3:2] is 00. The Cnt[3:2] is equal to 01, the lower hierarchy scanning mode is CC and Add[3:2] is 10. The remaining cases are all listed in the following table:

UP Hierarchy	CC			
Cnt[3:2]	00	01	10	11
Add[3:2]	00	10	11	01
Low Hierarchy	RR	CC	CC	-RR

The least significant two bits Cnt[1:0] is used to indicate the lower two bits address information for a pixel that belongs to a certain sub-quadrant, according to the given scan mode. For example, when the scan mode is RR, the mapping between the value of Cnt[1:0] and Add[1:0] is listed in the following table:

RR

Cnt[1:0]	00	01	10	11
Add[1:0]	00	01	11	10

The overall system is therefore consists of a simple counter and multiplexers. These multiplexers are used to map the value of Counter output and the generated Address. Table I is the mapping between the upper hierarchy scan mode and the lower hierarchy scan mode as well the mapping of counting value and the address. This mapping is implemented in our system by simply using multiplexers.

TABLE I

Direct Mapping System developed by Multiplexers for Hilbert Address Generation

Input		Output	
Upper Scan	Cnt[n+1:n]	Add[n+1:n]	Lower Scan
RR(00)	00	00	CC (11)
	01	01	RR (00)
	10	11	RR (00)
	11	10	CC (11)
-RR(01)	00	11	-CC (10)
	01	10	-RR (01)
	10	00	-RR (01)
	11	01	CC (11)
-CC(10)	00	11	-RR (01)
	01	01	-CC (10)
	10	00	-CC (10)
	11	10	RR (00)
CC(11)	00	00	RR (00)
	01	10	CC (11)
	10	11	CC (11)
	11	01	-RR (01)

The direct mapping system reported in Table I is used to generate 2-bit address. It only consists of multiplexers, therefore the mapping task can be finished in one clock cycle. In order to get the address of more than 2 bits, this mapping module should be cascaded. The overall cascaded system can also complete the mapping task in just one clock. In the bottom layer of Hilbert scanning, there is no need to get the scan mode of the lower hierarchy. Therefore, the mapping between the upper and lower scan modes could be removed in the bottom layer. Therefore, the overall Hilbert address generator is constructed by cascading $k - 1$ number of full mapping

TABLE II

COMPARISON OF COMPUTATIONAL COMPLEXITY AND MEMORY STORAGE WITH OTHER ALGORITHMS

	Computational Costs		Storage (bytes)
	Generating all addresses	One-to-One mapping	
Our Method	$O(2^{2k})$	$O(1)$	$13k+9$
Kamata	$O(2^{2k})$	$O(k)$	$8+2(1+2^2+2^4+\cdots+2^{2(k-1)})+2^{2k}$
Quinqueton	$O(k\cdot 2^{2(k+1)})$	$O(k\cdot 2^{2(k+1)})$ in worst case	$2k$
Butz	$O(2^{4k+2k\cdot 2^k})$	$O(2k)$	10k

systems and 1 reduced mapping system for an image in the size of $2^k \times 2^k$.

The generated address Add[k-1:0] should then be rewired to obtain the x- and y-coordinates for pixel readout. The even bits of Add[k-1:0] will be grouped together to form the x-coordinate while the odd bits of Add[k-1:0] will be grouped together to form the y-coordinate. This task is merely simple rewiring, therefore it takes no time to execute. The Hilbert address generation system is implemented in VLSI chip with integrated CMOS image sensor as shown in Figure 5

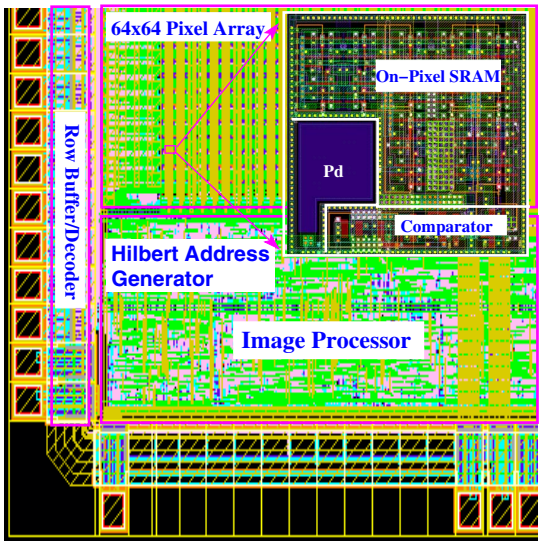

Fig. 5. CMOS image sensor with Hilbert Address Generator

The multiplexers used in the mapping system require the mapping information to be stored in registers. The full mapping system requires $(4{+}16{+}16{+}16){\times}2 = 104$ 1-bit registers for storing the mapping information as shown in Table I. The reduced mapping system requires merely $(4{+}16{+}16){\times}2 = 72$ registers for storing the mapping information. For a $2^k \times 2^k$ sized image, a total number of $104 \times (k-1) + 72$ registers are required. It can be translated into $13k+9$ bytes memory storage.

The computational complexity and memory storage comparison results for generating Hilbert address for a $2^k \times 2^k$ sized square image are presented in Table II. The Quinqueton algorithm takes much longer time for computing the one-to-one mapping than other methods due to its use of recursive functions. Butz algorithm is still very complicated in generating one-to-one address mapping due to its use of five basic bit operations. It also takes the longest time to compute all addresses. Kamata algorithm can obtain the one-to-one mapping slightly less complicated than that of the Butz algorithm, however it consumes huge amount of memory access to store the look-up-table. Our method can generate every address through one-to-one mapping in just one clock cycle with hardware consumption slightly more than the Butz algorithm. In general, our method outperforms other methods in both computational complexity and memory storage, which is highly desirable for hardware implementation.

IV. CONCLUSION

This paper presents an extremely fast algorithm for Hilbert address generation that can be easily implemented in hardware. The key feature of this work is that it can generate every Hilbert address in one clock cycle by using a simple incremental counter and multiplexers which requires less hardware storage compared with other works [8] [9] [10] , none of which can accomplish the task in just one iteration. This simple and efficient algorithm is the most cost-effective technique up to date to generate Hilbert address for square images. The generalized Hilbert address generation algorithm for rectangular images will be our future work.

ACKNOWLEDGMENT

This work was supported by the Research Grant Council of Hong Kong SAR, China, under Project HKUST610405.

REFERENCES

[1] G. Peano, "Sur une courbe qui remplit touteune aire plane", *Math. Ann.*,vol. 36, pp. 157-160, 1890

[2] D. Hilbert,"Über die stetige Abbildung einer Linie auf ein Flächenstück", *Math. Ann.*, Vol. 38, pp. 459-460, 1891

[3] S. Kamata, N. Niimi, and E. Kawaguchi, "A Gray Image Compression Using a Hilbert Scan", *Proceedings of ICRP*, pp. 905-909, 1996

[4] P. Kocharoen, K. M. Ahmed, R. M. A. P. Rajatheva, and W.A.C.Fernando, "Mesh-based video coding for low bit-rate communications", *IEEE Transactions on Consumer Electronics*, Vol 52, Issue 2, pp. 611-620, May 2006

[5] Y. K. Wang; H. Kuroda, "Hilbert scanning search algorithm for motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol 9, Issue 5, pp. 683-691, Aug. 1999

[6] N. Memon, D. L. Neuhoff, S. Shende, "An analysis of some common scanning techniques for lossless image coding", *IEEE Transactions on Image Processing*, Vol 9, Issue 11, pp.1837-1848, Nov. 2000

[7] M. Petrou, H. Peixin, S. Kamata, C. I. Underwood, "Region-based image coding with multiple algorithms", *IEEE Transactions on Geoscience and Remote Sensing*, Vol 39, Issue 3, pp. 562-570, Mar 2001

[8] J. Quinqueton and M. Berthod, "A locally adaptive Peano scanning algorithm", *IEEE Trans. Pattern Anal. Machine Intell.*, col. 3, pp.403-412, July 1981.

[9] A. R. Butz, "Convergence with Hilbert's space filling curve," *J. Comput. Syst. Sci.*, vol. 3, pp. 128-146, 1969.

[10] S. Kamata, *et Al.* "A New Algorithm for N-Dimensinal Hilbert Scanning", *IEEE Transactions on Image Processing*, vol. 8, No. 7, July, 1999