A Low-Power Geometric Mapping Co-Processor for High-Speed Graphics Application

Selwyn Leeke and Koushik Maharatna
University of Bristol, Bristol, UK
selwyn.leeke@gmail.com; Koushik.Maharatna@bristol.ac.uk

Abstract— **In this article we present a novel design of a low-power geometric mapping co-processor that can be used for high-performance graphics system. The processor can carry out any single or a combination of transformations belonging to affine transformation family ranging from 1-D to 3-D. It allows interactive operations which can be defined either by a user (allowing it to be a stand-alone geometric transformation processor) or by a host processor (allowing it to be a co-processor to accelerate certain graphics operations). It occupies a silicon area of 6 mm^2 and consumes 40 mW power when synthesized with 0.25µm technology.**

I. INTRODUCTION

The demand for more realism in 3D graphics system in terms of quality, interactivity and simulation of physical effect calls for the design of high-performance graphics subsystems. The main design criterion of such system is strong support of arithmetic functionality to perform the required kernels over the high bandwidth data stream [1, 2]. To accommodate future scaling of performance, it is envisaged that introduction of arithmetic modules which provide high throughput for a variety of different arithmetic function will be necessary [3, 4]. Apart from the quality and speed of operation, the trend to incorporate graphics systems in mobile and portable devices makes power dissipation an important design criterion. The massive computational requirement for a graphics processor results in significant power dissipation. One way to reduce power dissipation is by assigning computationally intensive arithmetic functions to specially designed low-power ASIC co-processors that can be used together with a host processor. In this work we are mainly interested to develop such a low-power co-processor that can be included in the geometry subsystem of a graphics system; for carrying out the geometrical mapping (affine transformation) operation on the input streaming data according to the user specification. Typically, such a co-processor should be able to perform translation, scaling, shearing and rotation operations on the input data. The operation may take place from 1-D to 3-D according to the user specification. Thus interconnections between different sections of the co-processor should be configurable and flexible enough for carrying out the operations in any order specified by the user.

In this work we propose a module-based 16-b fixed-point novel low-power co-processor that can satisfy the above requirements. Newly developed novel CoOrdinate Rotation DIgital Computer (CORDIC) processor [5] has been used to design the rotation section of the co-processor. Novel adder design has been used to realize the scaling and shearing sections. The co-processor can act as a stand alone geometric transformation processor where the inputs can be provided by the external user or can be interfaced with a host processor to carry out the same task. When synthesized in 0.25µm technology the processor dissipates 40mW power at 80 MHz frequency and occupies 6 mm^2 area.

The rest of the paper is structured as follows: in Section II we describe the architecture of different arithmetic modules comprising the co-processor and Section III describes its top-level design and overall controlling scheme. The implementation results are discussed in Section IV and conclusions are drawn in Section V.

II. MODULE DESIGN

The main arithmetic operations required for the target co-processor are addition (translation), trigonometric multiplication (rotation) and arithmetic multiplication (scaling and shearing). In general multiplication is the most area and power hungry component in a VLSI design. Although trigonometric multiplication can be realized by using CORDIC technique which adds elegance and flexibility in the design; from the low power perspective it is necessary to select appropriate arithmetic multiplication scheme. To do this under the constraint of area and power requirement we have considered three approaches for 16×16-b multiplication: conventional array multiplier, linear CORDIC [6] and shift-and-add based multiplication. All the architectures are synthesized in 0.25µm CMOS library for comparing them on a uniform platform. Since adders are the fundamental entities for multiplication - particularly for shift-and-add technique - we paid special attention to reducing the critical path of 16-bit adder by employing Carry-Select Adder (CSA) in conjunction with the concept presented in [7]. Here the sums corresponding to the input carry = '1' can be computed from the sum considering '0' input carry using an *add*-1 circuit. This enables one to implement a high-speed adder circuit without significant hardware overhead. But the add-1 circuit proposed in [7]

cannot be applied to a standard cell-based design since manipulation in the transistor level is required. In our design we have used a new modified add-1 circuit based on the standard logic gates. Fig. 1 shows the circuit diagram of such a 4-bit CSA as an example. The resultant circuit is slightly slower than the conventional CSA by one AND gate and a 2-to-1 multiplexer delay but cut down the hardware cost approximately by 92% compared to a conventional CSA.

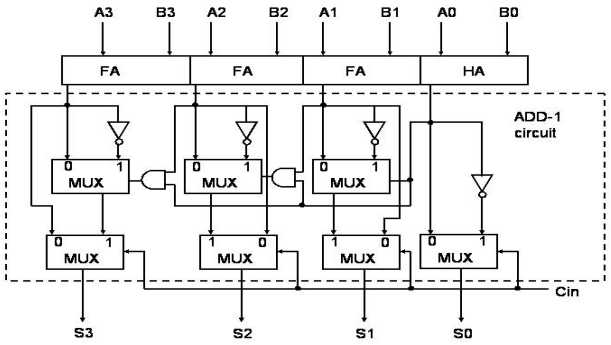

Fig. 1 The proposed CSA structure

The comparison of all the three multiplication schemes shows that the shift-and-add technique offers 20% and 9.7% reduction in power and area respectively compared to the conventional array multiplier. The linear CORDIC-based approach results in the worst performance consuming more than two times the power and four times the area of the shift-and-add based multiplier. Thus for construction of the scaling and shearing units we have used the shift-and-add based approach that utilize the newly proposed adder. Unless otherwise mentioned, from this point on we will refer this structure as "multiplier" in this paper.

The entire architecture is designed in a modular way where each unit is responsible for performing a specific task. Each of these units can be configured internally to perform its operation either in 1-D, 2-D or 3-D using a "dimension code-word" (physically; some signals). Each of these units is also equipped with two signals indicating the presence of valid data at their input and output. These signals are used in conjunction with the dimension code word to shut down the unused parts of a unit by gating the external clock. In the forthcoming subsection we describe architecture of different units.

Translator: Translation is simply the addition of a vector to a collection of co-ordinates and is the simplest unit here. Its only contents are a 16-b adder for each dimension, thus three in total. Each of the adders has been constructed using the newly proposed adder described above.

Scaling: The scaling unit multiplies co-ordinate values by a set of user defined coefficients. Three 16×16-b multipliers have been used for this purpose. For better controllability and power reduction this unit is partitioned in its 2-D and 3-D modes of operation. Each of these modules is provided with a gated clock. The dimension code shown in Table 1 is used to control these gated clocks and to configure the entire unit internally.

Dimension code	Nature of operation
00	XY
01	XZ
10	YZ
11	XYZ

Table 1 Dimension code-word for scaling and shearing unit

Here X, Y, Z denotes the principal axes of a Cartesian coordinate system and XY means scaling along X and Y axes. For 1-D operation the 2-D mode of operation can be used with one of the scaling coefficient set to one. This arrangement simplifies the logic circuit required for controlling the scaling unit.

Shearing: To achieve fast operation and power reduction we have split the shearing unit in two functional parts *viz*., multipliers and adders and introduce single-stage pipelining between them as shown in Fig. 2 where the a's, b's and c's are the weighting coefficients defined by the user.

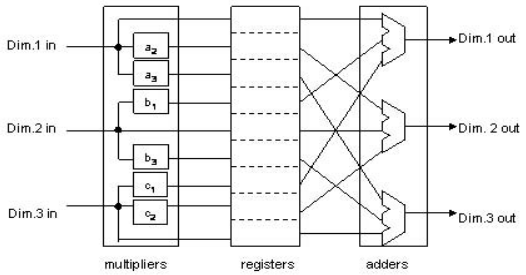

Fig. 2 Structure of the shearing module

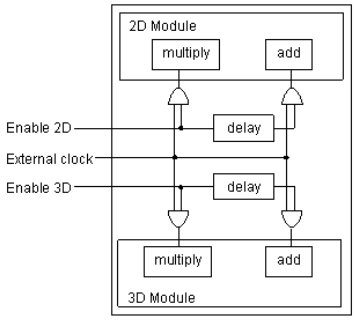

Fig.3 Enable signals and gated clocks in shearing unit

The Dim* denotes the data at the input and output. On top of that this unit is partitioned in 2-D and 3-D modules as shown in Fig. 3. Each of these modules is provided with a gated clock which can be activated by asserting Enable* signals. A single register delay has been used for the enable signals in the adder sections in order to balance the effect of the internal pipeline register. Like the scaling unit, the shearing unit also uses a similar type of dimension code-word (Table 1) which is used to control the enable signals for the 2-D and 3-D sections separately.

Rotation: To design the rotational unit we have used a newly developed 16-b pipelined CORDIC module [5]. Unlike the conventional CORDIC it eliminates the requirement of scaling and reduces the number of required iteration by 50% on an average. Three such CORDIC modules are used to carry out rotation ranging from 1-D to 3-D. A typical affine rotation requires data along two axes to be rotated by the user defined angle while the data corresponding to the third axis passes through without rotation [8]. A FIFO of same length as the CORDIC pipeline (here 16) is provided within the basic Processing Element (PE) for this purpose. The simplified block diagram of the rotation unit is shown in Fig. 4. In order to carry out all the possible rotation operations the data at the input of each PE needs to be rearranged. To do this, we have provided multiplexers at the input of each of the PE. From the nature of the rotation operation we have identified three unique kinds of data links between the PEs corresponding to rotation about different axes as shown in Fig. 5. The appropriate link type is selected using a 4-b dimension code-word according to the operation sequence specified by the user or the host processor.

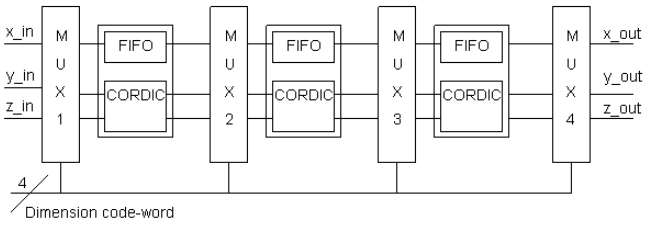

Fig.4 Architecture of the rotational unit

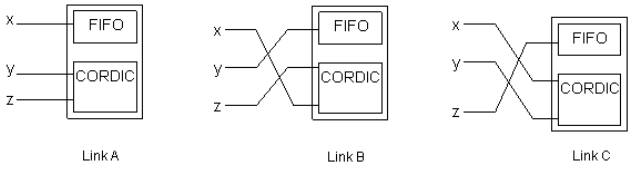

Fig. 5 The unique link connections for different rotation

Table 2 summarizes the relationship of different link types (configuration of the Multiplexers shown in Fig. 4) with the dimension code-word. Here MUX4 acts as the output multiplexer. Depending upon the dimension code-word MUX4 selects the appropriate output from either of MUX1, MUX2 or MUX3. As before X, Y, Z denotes the principal axes of rotation. The meaning of XYZ in Table 2 is that the object is first rotated about the Z axis and then about the Y axis and finally about the X axis. Appropriate enable signals are derived from the dimension code-word to shut down the unused CORDIC modules by gating the external clock signal and thereby reducing unwanted power dissipation.

Dimension Code-word	Axis of rotation	Link type of the multiplexers		
		MUX1	MUX2	MUX3
0001	X	A	-	-
0010	Y	B	-	-
0011	Z	C	-	-
0100	YX	A	B	-
0101	XY	B	C	-
0110	XZ	C	B	-
0111	ZX	A	C	-
1000	YZ	C	C	-
1001	ZY	B	B	-
1010	ZYX	A	B	C
1011	ZXY	B	C	C
1100	YXZ	C	B	B
1101	YZX	A	C	C
1110	XYZ	C	C	C
1111	XZY	B	B	B

Table 2 Relationship of link types with axis of rotation

III. SYSTEM ARCHITECTURE AND CONTROL

The overall block diagram of the co-processor is shown in Fig. 6.

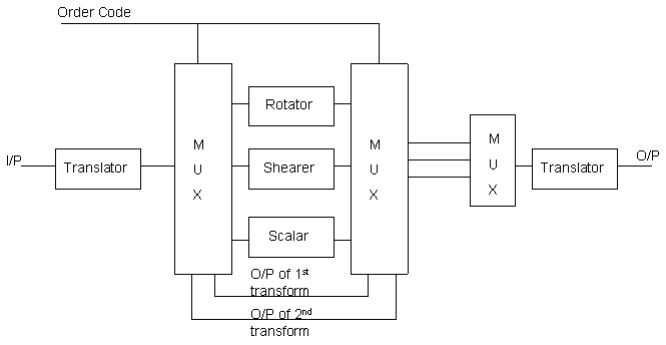

Fig. 6 The overall architecture

In this arrangement we have provided two translator units at the input and output. These are necessary since the operation may take place along a non-principal axis. The entire co-processor is controlled by a hierarchical coding scheme. At the top level the co-processor is controlled by an "order code" generated either by the host processor or given by the user. This code essentially decides the sequence of operations the co-processor needs to perform. The multiplexers at the input and output are configured accordingly to route the data to the appropriate unit in appropriate order. Table 3 shows the summary of the order code and its relationship with the sequence of operation.

Order Code	1st Operation	2nd Operation	3rd Operation
0000	-	-	-
0001	Rotate	-	-
0010	Scale	-	-
0011	Shear	-	-
0100	Rotate	Scale	-
0101	Rotate	Shear	-
0110	Scale	Rotate	-
0111	Scale	Shear	-
1000	Shear	Rotate	-
1001	Shear	Scale	-
1010	Rotate	Scale	Shear
1011	Rotate	Shear	Scale
1100	Scale	Rotate	Shear
1101	Scale	Shear	Rotate
1110	Shear	Rotate	Scale
1111	Shear	Scale	Rotate

Table 3 Transform order code

This top level order code is concatenated with the dimension code-words of the individual unit to form a 12-b instruction bit field as shown in Fig. 7.

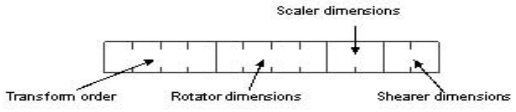

Fig. 7 Structure of instruction bit field

From the perspective of the end user, only specification of this instruction bit field is necessary. Once it is defined the co-processor configures itself internally according to the functions described in Table 1, 2 and 3 and outputs the final results. In case of using the co-processor with a host procesor, this instruction bit field is generated by the host processor.

IV. IMPLEMENTATION RESULTS

The co-processor is synthesized using 0.25µm CMOS library. The maximum achievable frequency is 80 MHz at 3.3V supply. The overall area required by the co-processor is 6 mm^2. The rotator unit and the shearing unit consume 46% and 35% of the total area respectively. The scaling unit is the next biggest consuming 14% followed by the translation unit (< 1%). The control circuitry including the hardware requirement for clock gating consumes approximately 4% of the total area.

Power dissipation has been analyzed using Synopsys' Prime power. The total power consumption of the processor at 3.3V supply voltage and 80 MHz frequency is 40 mW. The power consumption of the rotational and shearing unit is 52% and 35% respectively whereas the scaling module consumes 11% of the overall power. The rest of the power is consumed by the translation unit and the control circuitry.

V. CONCLUSIONS

In this article we have proposed a novel geometric mapping co-processor that can act as a stand-alone unit or can be used with a central processor for high-performance graphics application. The processor's pipelined nature fascilitates high-throughput while the strategies adopted for power reduction results in significantly less power consumption. The hierarchical control structure of the co-processor is extremely simple, is hardware efficient and enables one to apply clock gating more effeciently. Although the processor designed here is a fixed-point one, without loss of generality it can be extended for floating-point. It is envisaged that when fabricated with the current generation of technology the co-processor will exhibit significant speed advantage.

REFERENCES

[1] N. Ide et al., "2.44 GFLOPS 300 MHz floating-point vector-processing unit for for high-performance 3-D computer graphics computing", *IEEE J. Solid-State Circuits*, vol. 35, no. 7, pp. 1025 – 1033, Jul. 2000.

[2] E. Lindholm et al., "A user programmable Vertex engine", *Proc. ACNSIGGRAPH* 2001, pp. 149 – 158, 2001.

[3] W. J. Dally, P. Hanrahan, M. Erez and T. J. Night, "Merrimac: supercomputing with streams", *Proc. Supercomputing Comf.*, Nov. 2003.

[4] U. J. Kapasi et al., "Programmable stream processors", *Computers*, vol. 36, no. 8, pp. 54 – 62, Aug. 2003.

[5] K. Maharatna, A. Troya, S. Banerjee and E. Grass, "New virtually scaling free adaptive CORDIC rotator", *IEE Proc. Computers and Digital Techniques*, vol. 151, no. 6, pp. 448 – 456, November 2004.

[6] J. S Walther, "A unified algorithm for elementary functions", *Proc. Joint Spring Comput. Conf.*, vol. 38, pp. 379 – 385, Jul. 1971.

[7] T. Y. Chang and M. J. Hsiao, "Carry-select adder using single ripple-carry adder", *Electronics Letters*, vol. 34, no. 22, pp. 2101 – 2103, October 1998.

[8] F. H. Hill Jr., *Computer Graphics*, Prentice Hall NJ, 1990.