Algorithmic Level Low-Power VLSI Design Applied to the RGB to HSI Conversion

A.Th. Schwarzbacher[1,2], J.B. Foley[2] and J.T. Timoney[3]

[1]*Dublin Institute of Technology, Dublin, Ireland.*
[2]*Trinity College, Dublin, Ireland.*
[3]*NUI, Maynooth, Irland*

Abstract: In the early 90s the first portable computing applications became widely available. At this time, lower performance than the mains dependant counterparts was accepted. However, today user demand small lightweight portable systems with long battery operating times without any compromise in computing performance. To meet these customer demands, a consequent low power approach has to be taken starting at the first design decision. This paper will present the implementation of a low-power real-time image processing circuit for the transformation of a camera signal into the human perception code.

1 Introduction

The aim of this paper is to present a system level approach for the low power implementation of computationally intensive algorithms. Kender's algorithm for faster computation of hue [1], as shown in Table 1, was chosen as a way to investigate general valid methods of reducing the power consumption at the first stages of the VLSI design cycle. The traditional method of reducing the supply voltage [2] was not applicable for this task as the design was to be mapped into an ASIC library. Here voltage scaling is only possible in a very limited range. It therefore became clear that the only possibility to minimise the power consumption in such a design was to reduce the active capacitance.

To enable a detailed analysis, the design was split into the three main paths, one for computing hue, saturation and intensity. As shown in Figure 1, these paths were again subdivided into smaller blocks, each containing a typical set of implementation problems. These blocks could then be investigated separately for potential reductions in power consumption. In the hue path, the main task

Kender's Algorithm for Faster Computation of HUE:

```
if ((R > B) and (G > B)
    $hue = \frac{\pi}{3} + \arctan\left(\frac{\sqrt{3}\times(R-G)}{R-B+G-B}\right)$
else if (G > R)
    $hue = \pi + \arctan\left(\frac{\sqrt{3}\times(B-G)}{B-R+G-R}\right)$
else if (B > G)
    $hue = \frac{5\times\pi}{3} + \arctan\left(\frac{\sqrt{3}\times(R-B)}{R-G+B-G}\right)$
else if (R > B)
    $hue = 0$
else
    'achromatic'
```

Saturation:

$$saturation = 1 - \frac{3\times\min(R,G,B)}{R+G+B}$$

Intensity:

$$intensity = \frac{R+G+B}{3}$$

Table 1: The HSI Algorithm using Kender's Algorithm for the Faster Computation of Hue

was to implement the alternating function of the arctan using unsigned arithmetic. This was achieved through the use of sign detection in the first stage of the design. This resulted in smaller logic in the remaining stages [3]. The second task was the implementation of the arctan function itself. The standard implementation for all trigonometric functions is the CORDIC algorithm [4]. However the CORDIC is a general purpose algorithm. Due to the range of the arctan it was possible to implement it using a LUT [5]. This already yields power savings of a factor of 20. The transformation of the LUT also pointed into the direction of a hardware optimised approximation algorithm. This algorithm could realise the arctan function by using only one adder and one constant shift operation. Here a comparison with the CORDIC algorithm showed a reduced power consumption by a factor of 25 [5]. Furthermore, the other features such as the maximum frequency of operation or area requirements could be significantly reduced. Therefore, it was shown that the main strength of the CORDIC algorithm is in the area of mathematical multiprocessors rather than in single function implementations. An investigation of the control pipeline showed a significant power inefficiency. Firstly, different coding styles were applied to the design. However, the result was unsatisfying as theoretically superior codes produced a higher power consumption than standard approaches. A detailed analysis showed that these power reduced codes had in fact a higher power consumption in the clock network. Hence, an alternative to the traditional shift register implementation was developed. Using this new shift register it was possible to show in a general study that even for small designs, such as those needed for the control bus, power savings of up to 30% could be achieved [6]. For larger shift register this figure increases even further.

The first design decision of the saturation path was the reuse of terms already computed in the hue path. This resulted in smaller logic, the use of balanced structures and a reduction in pipeline stages. Following this, four different implementations were considered [7]. Theoretically, all implementations have theoretical advantages. However, the direct implementation of the saturation path showed the best power performance. This indicates the difference between the software-optimised algorithms and hardware-optimised once. Although the direct implementation was not the smallest possible mathematical function, the block diagram nevertheless showed that it will be the smallest in terms of functional blocks. The investigation of the direct implementation also highlighted that this functions superiority in terms of power consumption.

The final path to be implemented was the intensity algorithm. As in the case of the saturation algorithm, it was possible to reuse terms previously calculated for hue and intensity. Therefore, only one division by three had to be implemented. To achieve such a constant divider, various mathematical algorithms were developed [7]. An overall investigation of six different algorithms showed that the algorithm proposed by Petry [8] gave the best power to area/speed performance. It was therefore used in this design. The accuracy of the intensity was also investigated. Here it was possible to replace the least significant bit by a constant ONE. This resulted in smaller logic and smaller power consumption while improving the accuracy by 33% [6].

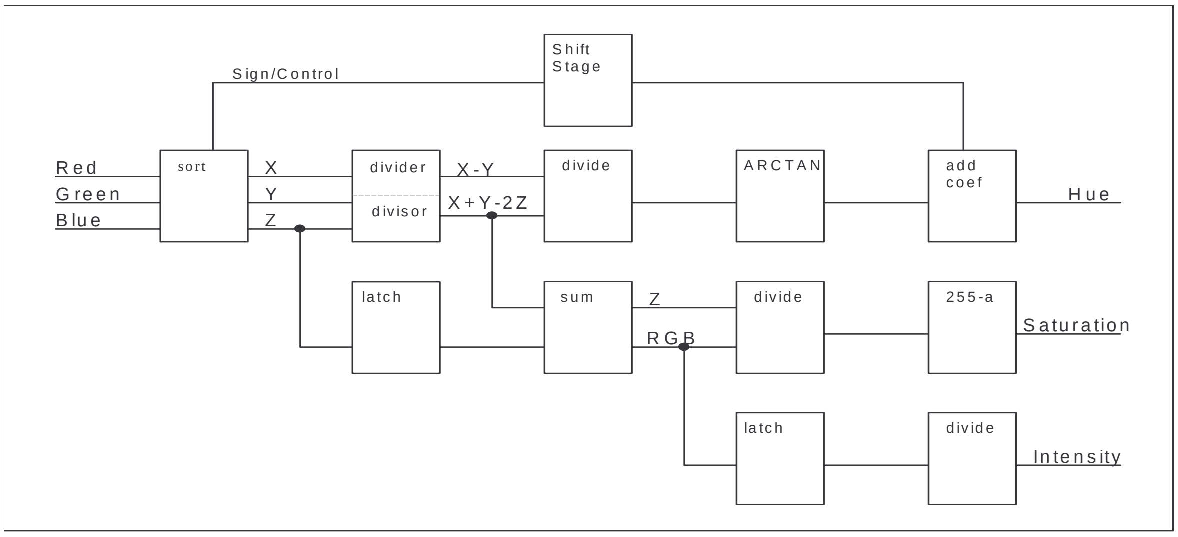

Figure 1: Clock Diagram of the RGB to HSI Converter

2 Circuit Features

This chapter will present the features of the implementation of the image processing algorithm. Table 2 summarises the performance of the low power design and a direct implementation of the algorithm presented in Table 1.

Performance of the RGB to HSI Converter		
Technology	ES2 07µm (industrial)	
Supply Voltage	5V (±0.5V)	
Input Signals	Red, Green, Blue 8-bit unsigned, Clock	
Output Signals	Hue, Saturation, Intensity 8-bit unsigned	
Throughput	30Mpixles / cycle	
Operating Frequency	30MHz	
	Low-Power Implementation	Direct Implementation
Area	4.19mm^2	3.7mm^2
Number of Pipeline Stages	5	4
Output Signal Deviation	Between -2 and +2 bit (0.78%)	Between -1 and +1 bit (0.39%)
Active Capacitance	187.9pF	298.9pF
Maximum Settling Time	18ns	36ns
Maximum Throughput	50Mpixles / cycle	27.7Mpixles / cycle
Avg. Power Consumption	140mW (@30MHz)	224mW (@30MHz)

Table 2: Performance of the RGB to HSI Converter

Originally the circuit was designed to convert images with a resolution of 1024 by 1024 pixels at 25 frames per second. However, after optimising the power consumption the maximum conversion rate was 50 Mpixels per second. At this peak conversion rate the power consumption will rise by 39%, from 140µW to 224µW. Therefore, the clock frequency should be always kept at the minimum operation speed. This results in a system with the lowest possible power dissipation.

The features of the low power design were compared with a direct implementation of the RGB to HSI algorithm. This direct implementation of the algorithm uses the native VHDL operations to implement the various functions. As shown in Table 2, the power consumption of the direct implementation is 37% higher than that of the power optimised implementation. Additionally, the maximum throughput and the maximum computational image resolution were increased in the low power version by a factor of 1.8. This increased performance is due to the balancing of the paths. In the low power implementation paths which did not meet the timing requirements were shortened and subsequently, the overall delay was decreased. In a voltage scalable circuit, this could be used for even further power reductions. In this case it would be possible to reduce the voltage down to approximately 3.5V in a full custom design which would result in a reduction in power by half. The number of pipeline stages of the power optimised design was increased by one. This had the effect of balancing the paths and reducing the glitching in the divider structures. It also caused a rise in latency by one clock cycle. However, if the latency of 90ns of the power optimised design is compared to that of the direct implementation, it can be seen that the additional pipeline stage did reduce the latency by 54ns. The only features of the low power implementation which has not improved are the deviation and area. However, this result was anticipated as area is the factor most often traded for a reduced power consumption [2]. The area increase of 13% is very reasonable if it is compared to the reduced power consumption of 37% and increased performance of nearly 93%. The larger deviation of ±1bit was traded against a smaller power consumption of the hue path. However, as has been shown in [7] this larger deviation only occurs in less than 2% of all possible output sates and does not contribute to any perceptual error to the image transformation.

To implement this RGB to HSI conversion on a DSP chip such as the TI TMS320C6211, a minimum of 19 operations are required. If a RGB signal with a resolution of 1024 by 1024 pixels is to be transformed at a frame rate of 25, this results in 78,643,200bytes/second. Therefore, implementation of the transformation of this RGB signal using Kender's algorithm would result in 489MOPS which is half of the available processing power of the TMS320C6211 [9] and could consume 1.1W [9] which is 7.9 times that of the proposed image conversion circuit.

3 Conclusions

A comparison of the approaches presented in this paper and a traditional implementation of the RGB to HSI algorithm showed a significant power saving of 37%. Also the computational throughput of the circuit was improved by a factor of 1.8. This was achieved because in the low-power version of the implementation a path balancing approach was used, which resulted in a maximum path length of 18ns. The only drawback to the low power implementation is a 13% increase in area. This was due to the more complex logic required to optimise the switching activity, as well as the additional pipeline stages used to balance the path length. A comparison of the low power circuit with a DSP chip has shown its superiority in terms of power consumption and computational performance.

In conclusion, the task of showing that a thorough investigation at the algorithmic level of the VLSI design cycle can lead to significant power savings was successfully undertaken. As the design was to be mapped to an ASIC, the traditional approach of voltage scaling was not applicable. Therefore, this project has focused on the reduction of the active capacitance in a real-time processing algorithm for the transformation of RGB to HSI. Here various typical design problems such as the division by a constant factor, and the implementation of a pipeline were investigated. Novel approaches have been presented to effectively reduce the active capacitance and therefore the power consumption of the IC. This has resulted in a circuit with a reduced power consumption and increased performance.

References

[1] J. Kender, "Saturation, hue and normalized color," *Carnegie-Mellon University, Computer Science Dept.*, Pittsburgh PA. 1976.

[2] Chandarakasan and S. Sheng, "Low-Power CMOS Digital Design," *IEEE Journal of Solid State Circuits*, Vol. 27, No. 4, April 1992.

[3] A.Th. Schwarzbacher and J.B. Foley, "Low-power design: an image processing chip for RGB to HSI conversion," *Irish Systems and Signals Conference*, Derry, Ireland, pp. 165-172, June 1997.

[4] J.E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Computer*, vol. EC-8, no. 3, pp. 330-334, September 1959.

[5] A.Th. Schwarzbacher, A. Brasching, Th.H. Wahl, and J.B. Foley, "Optimisation and implementation of the arctan function for the power domain," *Electronic Circuits and Systems Conference*, Bratislava, Slovakia, pp. 33-36, September 1999.

[6] A.Th. Schwarzbacher, P.A. Comiskey and J.B. Foley, "Reduction of the power consumption at the algorithmic level of CMOS circuits," *Electronic Systems and Devices Conference*, Bruno, Czech Republic, pp. 5-8, June 1998.

[7] A.Th. Schwarzbacher, J.B. Foley, M. Brutscheck and O. Schwingel, "Optimisation of real-time signal processing algorithms for low-power CMOS implementations," *International Symposium on Communication Systems, Networks and Digital Signal Processing 2000*, Bournemouth, Untied Kingdom, pp. 38-42, July 2000.

[8] A.Th. Schwarzbacher, M. Brutscheck, O. Schwingel and J.B. Foley, "Constant divider structures of the form $2^n \pm 1$ for VLSI implementation," *Irish Signals and Systems Conference*, Dublin, Ireland, pp. 368-375, June 2000.

[9] Texas Instruments, "How to begin development today with the TMS320C6211 DSP," *Application Report*, Texas Instruments, September 1998.