

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

GateOS: A minimalist Windowing Environment and Operating System for FPGAs

A thesis presented in partial fulfilment of the requirements
for the degree of

**Master of Engineering
In
Computer Systems Engineering**

At Massey University, Palmerston North,
New Zealand

Andreas Buhler

2007

ABSTRACT

In order to debug and tune stand-alone FPGA image processing configurations, it is necessary for a developer to also create the required debug tools and to implement them on the FPGA. This process takes both time and effort that could be better spent on improving the image processing algorithms. The Gate Array Terminal Operating System (GateOS) is proposed to relieve the developer of the need to construct many of these debugging tools. In GateOS we separate the image processing algorithms from the rest of the operating system. GateOS is presented to the developer as a Handel-C library, which can be customised at compile-time, to facilitate the creation of windows and widgets. Several types of widgets are described that can manipulate the parameters of image processing algorithms and enable the end-user to dynamically rearrange the position of a window on the VDU. An end user is able to interact with GateOS with both a keyboard and a mouse.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Associate Professor Donald Bailey, for all the guidance given to me over the previous year. Without his sound advice, GateOS would not be what it is today. I thank him for being brutally honest with me whenever I became sidetracked and deviated from dealing with the core issues of GateOS. I am also grateful for the considerable amount of time and patience expended on his behalf proofreading this thesis. Personally, I am astounded at how he was able to do this on his very tight schedule. I am thankful to Donald and Massey University for allowing me to use and take home various FPGA development boards. I have appreciated all the advice given to me by Chris Johnston and Kim Gribbon during the year. Their ideas for GateOS, recorded in a conference paper (Bailey et al, 2006) have proved invaluable in formulating the requirements of GateOS and to some extent, its design. I would like to thank Massey University for allowing me to complete the research for my thesis using their equipment in the computer labs.

I also acknowledge the role of Xilinx in providing me with licences to the ISE suite of software tools, which were required for the implementation of GateOS.

Finally, I would like to thank Roger Gook of Celoxica who has provided me with licences for the DK suite of software, so that I was able to use Handel-C to implement GateOS. He has also been very helpful by providing me with the source code for various board level drivers.

TABLE OF CONTENTS

ABSTRACT.....	i
ACKNOWLEDGEMENTS.....	ii
TABLE OF CONTENTS	iii
LIST OF FIGURES	vi
LIST OF TABLES.....	viii
CHAPTER One INTRODUCTION & LITERATURE REVIEW.....	1
CHAPTER Two REQUIREMENTS ANALYSIS & HIGH LEVEL DESIGN	7
2.1 Introduction.....	7
2.2 Proposal	8
2.3 Hardware Environment	9
2.4 System Overview.....	11
2.5 Non-Volatile Storage Manager.....	14
CHAPTER Three WINDOW MANAGEMENT.....	15
3.1 Introduction.....	15
3.2 Representing a window's size and location	16
3.2.1 Virtual Coordinate Space	17
3.2.2 Window extent	18
3.3 Overlap of windows.....	19
3.4 Window States	20
3.5 Window Regions	21
3.6 Managing and displaying windows.....	23
3.6.1 Parallel Incidence Test.....	23
3.6.2 Window Transition Method.....	26
3.7 Discussion.....	34
CHAPTER Four WIDGET MANAGEMENT.....	36
4.1 Introduction.....	36
4.2 Widget Display Layers	37
4.3 Widget Details.....	38
4.3.1 Label Widget.....	38

4.3.2	Button Widget.....	39
4.3.3	Text Edit Widget.....	40
4.3.4	Slider Widget	43
4.3.5	Imaging Widgets	46
4.4	Data-Structures and Interactions.....	50
4.4.1	Imaging widgets.....	52
4.5	Scheduling and Display of Widgets	53
4.5.1	Image Layer.....	53
4.5.2	Algorithm Control Layer	53
4.5.3	Window Control Layer	56
4.6	Discussion.....	56
CHAPTER Five MANAGEMENT & DISPLAY OF TEXT		58
5.1	Introduction.....	58
5.2	Requirements and Intended Usage.....	59
5.2.1	Window Labels.....	59
5.2.2	Widget Annotations.....	60
5.2.3	Output Text Box	61
5.2.4	Image annotations.....	61
5.2.5	Text editing	61
5.3	Analysis and Design	62
5.3.1	Image-Table Lookup Method.....	63
5.3.2	Font Table Lookup Method	66
5.4	Implementation.....	71
5.5	Discussion.....	73
CHAPTER Six INPUT MANAGEMENT.....		75
6.1	Introduction.....	75
6.2	Cursor Layer.....	76
6.3	Keyboard Input.....	79
6.4	Mouse Input	79
6.5	Discussion.....	81
CHAPTER Seven DISCUSSION & CONCLUSIONS.....		82

REFERENCES87

LIST OF FIGURES

Figure 2-1 Timing Regions within a video frame, active, H-blanking and V-blanking .	10
Figure 2-2 GateOS IP Core and its relationship to the rest of GateOS.....	12
Figure 3-1 Windowing System architectural overview	16
Figure 3-2 Choices of origin within virtual coordinate space.....	17
Figure 3-3 Virtual coordinate space based on the two encoding schemes.....	19
Figure 3-4 Changing the z-index of a window.....	20
Figure 3-5 Final layer with a grip for resizing windows.....	22
Figure 3-6 Detecting whether a window is present at a screen coordinate.....	24
Figure 3-7 Updated window identification that uses less hardware per window	25
Figure 3-8 Selecting the window with the highest z-index.....	25
Figure 3-9 How WLL entries for each scan-line are related to window positions	28
Figure 3-10 Worst case scenario with each windows vertical edges displayed.....	28
Figure 3-11 Edge list lookup-table.....	30
Figure 4-1 Content window layer split into three widget sub-layers	37
Figure 4-2 A simple Label Widget	38
Figure 4-3 The background of a Button Widget can be one of two possible colors.....	40
Figure 4-4 An Edit Widget displaying the string “The brown fox jumped”	41
Figure 4-5 Vertical and horizontal slider widgets.....	43
Figure 4-6 The offset of a horizontal Slider Bar from the left border.....	44
Figure 4-7 Two zoom buttons, one in and the other out	46
Figure 4-8 Slider widget is used to manipulate denominator of zoom fraction.....	47
Figure 4-9 Sample Histogram Content	48
Figure 4-10 B_VALUE and I_VALUE data structures.....	51
Figure 4-11 The data structures necessary for each widget type	51
Figure 4-12 Aligning common properties between widgets	52
Figure 4-13 Valid and invalid widget layouts.....	54
Figure 4-14 Order of widget data structures is determined by row then column	55
Figure 5-1 A simple textual label displayed on a window.....	60
Figure 5-2 String annotations on label and button widgets.....	60
Figure 5-3 The construction of the String LUT at compile-time.....	64
Figure 5-4 The use of an external utility to construct Image Table.....	65

Figure 5-5 Possible implementation strategies	66
Figure 5-6 Using bitmaps for individual characters.....	67
Figure 5-7 The string id and how it relates to the string, offset, and length tables.....	69
Figure 5-8 The Text Manager design.....	70
Figure 5-9 GUI Utility that manipulates the String Table	72

LIST OF TABLES

Table 4-1 Cursor movements	42
Table 4-2 Available numeric string conversions	43
Table 7-1 Slices used on a Xilinx ML402 (Virtex 4) for 8, 16, 32 and 64 windows.....	84
Table 7-2 Breakdown of resources required, as estimated by the Celoxica build tools .	85