

Image Calculator Final Project

Vilius Vysniauskas, Jonny Hurwitz, and Alex Peters

The Pivot

- Original intention was to combine several hardware modules
 - Modules: VMODCam and iRobot
 - Description: moving robot which can identify a color, drive toward it, and “retrieve” it
- Quickly ran into problems
 - Combining projects in the EDK meant manual re-creation of one of the projects
 - Caused errors in the SDK when trying to adapt code from the tutorials
- After 3 weeks of EDK debugging, we decided to try something else
 - Solely based on the VMODCam module

Project Description

A calculator that can see. The calculator uses image processing to identify numbers and operators that the user presents to the camera, and then calculates the result of an expression when the user is finished.

The calculator supports operations such as addition, subtraction, multiplication, and division. Operands are currently configured with a maximum of 4 digits each, with room for expansion depending on memory usage.

Industry Standards

- Visual
 - Consumers expect smooth animation, display synced to refresh rate, consistent aspect ratio
 - Shortcomings: tearing occurs (causes flickering), image is stretched vertically
- Hardware/Software Co-Design
 - Successful integration of FPGA/Xilinx modules and C programming concepts
 - Tightly coupled design
 - Hardware limitations (mainly memory) constrained certain software solutions
- Computer-Human Interaction
 - Edge cases considered and fixed
 - Clear and focused interaction through terminal
 - Response given on each taken action

Hardware

- VMod camera with dual 2MP cameras
 - SoC includes image flow processor for selectable output resolutions
 - We used RGB with 4 bits per color
- Video output to the monitor is over HDMI via the VHDCI connector
- Register writes done via Xilinx library functions with raw target addresses
- Image processing software running on MicroBlaze in C
 - Virtex-5 FPGA
 - Digilent Genesys development board
 - FPGA writes messages to console over serial connection

Software

- Our code also reads the camera values from RAM, each pixel is stored in 2 bytes as RGBx444
- To scan for a character, we divide a 200x200 pixel array into a 5x5 grid and analyze each block of the grid
- White value threshold for each color (RGB) is calculated by sampling data outside of the target box
- If the block is determined to be below the calculated white threshold, the block is marked as a 1, if not 0

	0	1	2		
+	+	+	+	+	+
	0	1	2		0
+	+	+	+	+	+
	3	4	5		1
+	+	+	+	+	+
	6	7	8		2
+	+	+	+	+	+
	9	10	11		3
+	+	+	+	+	+
	12	13	14		4
+	+	+	+	+	+

```
+-----+
| x| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|12|13|14|
+-----+
```

Grid representation in a short

	0	1	2	
0	0	1	2	
1	3	4	5	
2	6	7	8	
3	9	10	11	
4	12	13	14	

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
---	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----

Software cont.

- Once the grid is analyzed, the results are stored in a short, with each bit representing the value of a grid block
- The scan is repeated 4 more times, then the set of 5 shorts are passed into a function (`determineChar`), which averages each bit to determine if it should be a 1 or a 0
- The function then switches on the averaged short to check if it matches the short code of a digit or operator, then outputs the correct char
- The digits and operators are stored in a char array until processed in the function `printAnswer`

Software cont.

- `printAnswer` calls `stupidscanf`, which extracts operands `a` and `b` from the char array
- Then `printAnswer` performs the correct operation on `a` and `b`, and prints the answer to the terminal
- After an answer is calculated, everything is cleared and the program can accept a new expression

Development Costs

- Hardware
 - Camera module, FPGA, FPGA development board
 - Power supplies and cables
 - Debug time from engineers
- Software
 - Development and debug time from engineers
- General
 - Our sanity

0 1 2 3 4

5 6 7 8 9

+ - ■ ▤

8

8-

8-9

Your answer is: -1

Your expression has been cleared.

7

Please re-enter number or operator, unable to scan correctly.

76

765

765/

765/3

Your answer is: 255

Your expression has been cleared.

9

99

Please re-enter number or operator, unable to scan correctly.

999

9999

9999*

9999*9

9999*99

9999*999

9999*9999

Your answer is: 99980001

Your expression has been cleared.

After a result has been calculated and presented, the buffer is cleared and the software is ready for the next expression.

The software was unable to recognize the letter, so we ask the user to retry.

Each input character is presented to the user as part of the expression after the character has been successfully interpreted.

Demo

Thank you!