COMPENG 3DQ5 - Project Report

Group 34

Jashanjyot Randhawa - 400337963 - Keith MacLean - 400255044

Introduction

COMPENG 3DQ5 is a digital design oriented course in which students are tasked with designing the hardware to decompress an image using SystemVerilog code. In the project three milestones are required to be completed and put together to create a final product which takes a bitstream input and decompresses to a 320x240 coloured image.

Design structure

The design structure used in this project was that of creating separate modules for the milestone 1, milestone 2, and dual port ram initialization implementations. These modules were combined with the provided VGA, SRAM, and UART controllers in the top level finite state machine of the project. An alternative that was considered for the design structure of the project was breaking the states of each milestone into separate modules. This design would have facilitated simpler debugging in addition to making the design easier to comprehend. However, given that the group had fallen behind schedule it was necessary to use the most streamlined design structure possible.

Implementation details

Register table

Module	Register name	Bits	Description
Milestone1	SRAM_address	18	Address register used for accessing the external memory.
Milestone1	Y_val, U_val, V_val	32 (each)	Registers to store the values of Y read from the SRAM, and U and V calculated.
Milestone1	U_buf, V_buf	8 (each)	To keep values for another cycle.

Milestone1	U_p5, U_p3, U_p1 V_p5, V_p3, V_p1,U_m5, U_m3, U_m1 V_m5, V_m3, V_m1	8 (each)	Registers that comprise the shift registers for U and V
Milestone1	SRAM_read_data, SRAM_write_data	16 (each)	Registers assigned for holding data read from the SRAM or to be written to the SRAM, respectively.
Milestone1	R_val, G_val, B_val	32 (each)	To store RGB before writing to SRAM
Milestone1	RGB_count, UV_count, Y_count	18 (each)	Counters for addressing the SRAM
Milestone1	op1, op2, op3, op4, op5, op6	32 (each)	Operands for each of the multipliers
Milestone1	result1, result2, result3	32 (each)	Results from the multiplications, clipped from a 64 bit long
Milestone1	line_count	9	To count the current row
Milestone2	RAM0_address_a, RAM0_address_b, RAM0_wren_a, RAM0_wren_b, RAM0_read_data_a, RAM0_read_data_b, RAM0_write_data_a, RAM0_write_data_b RAM1_address_a, RAM1_address_b, RAM1_wren_a, RAM1_wren_b, RAM1_read_data_a, RAM1_read_data_b, RAM1_write_data_a, RAM1_write_data_b	Address: 7 (each), wren: 1 (each), read/write data: 32 (each)	Dual port ram addresses, enables and data registers. Used for writing and reading to and from the SRAM.

Milestone2	S_prime_count	8	For indexing the S' values to read and write from the dual port RAM
Milestone2	C_count, S_count, SC_count	7 (each)	Same usage as S' count but for other values.
Milestone2	S_offset	7	Offset for calculating transpose matrix of S'C
Milestone2	row_count, col_count	Row: 3 Col: 7	For indexing blocks
Milestone2	curr_max_row	18	Also for indexing blocks
Milestone2	row_offset, col_offset	18 (each)	Also for indexing blocks
Milestone2	fetch_Y	1	To determine if working on Y blocks or U and V blocks
Milestone2	S_prime_C	32	To store S'C values before writing to dual port RAM
Milestone2	S_result	32	To store S before writing to dual port RAM
Milestone2	S_prime_buf, SC_buf, S_buf	32 (each)	To store values for a cycle to write two at a time ot the dual port RAM.
Milestone2	write_count, block_count	Write: 19 Block: 12	To track write address for the SRAM and current block number
Milestone2	col, row	18	Also for indexing blocks
Milestone2	op1, op2, op3, op4	32	Operands for the two multipliers

Milestone2	result1, result2	32	Results of multipliers
Milestone2	done_read, done_write, write_ready, read_flag, even_loop, first_loop, done_curr_state	1 (each)	Flags to determine when writing/reading are done and when certain actions are to be performed and at which point a state should be left.

Timeline

Week	Keith MacLean	Jashan Randhawa
1	Reading through the project description, understanding the project timeline, and reviewing necessary lab components.	Started to read the project description. Understood the connection between the project and labs.
2	Worked on milestone 1 state table, used lecture and project description resources to generate ideas for how the finite state machine could be implemented	Started creating a state table for milestone 1. Went through many resources to understand the specific states.
3	Began Verilog implementation of the created state table for milestone 1.	Finished the state table for milestone 1. Starting implementing it into the milestone 1 code.
4	Worked on a rough milestone 2 state table whilst finalizing milestone 1 implementation and beginning to assist with the debugging process.	Continued writing code for milestone 1. This started the heavy debugging process. Also took a look at milestone 2 to try to understand it theoretically.
5	Began rapid implementation of milestone 2 in Verilog using the skeletal state table developed, no time remained for code to be debugged,	Continued with debugging the milestone 1 code. This ended with the completion of milestone 1. Helped with the understanding of milestone 2.

	worked on report.	

Milestone 1 Implementation

The hardware of the first milestone for the most part consists of shift registers, multipliers, adders and multiplexers. In the first milestone values used in the interpolation are stored and cycled through a shift register. These values have a constant subtracted using an adder then are used in the three multipliers used in the milestone to calculate the upsampled values of U and V. During this portion of the design the symmetry of the interpolation is abused to reduce the total number of clock cycles required to complete the decoding. Following the multiplication calculations, the results are stored in registers. These results once again have a constant removed in an adder then are used in the multipliers then shifted to determine the final RGB values which are also stored within registers. There are three groups of states within the finite state machine for this milestone. The first group is the lead in states. The lead in states are used to pad the left side of the screen with 0 values to prevent image distortion on the side. The second group of states is the common case, which is used to perform the common decoding of the RGB values, and loops until the end of the row is reached. Upon reaching the end of the row the third and final group of states, the lead out, are used to pad the right side of the screen with values equal to the border value to prevent distortion. These groups of states are then repeated for all rows. Given that there are a total of 16 common case states that are looped 80 times per row, and there are 240 rows, the total latency will be (16*80*240) + (240*29(LO states) + 240*12(LI states) = 317 040 clock cycles. The approach to this milestone was simply to follow the hardware structure described above implemented into the three groupings of states. The milestone passes the testbench v1 for the motorcycle and panda images, but strangely has 4 mismatches for the cat image. The multiplier efficiency in this milestone is approximately 83%. This was calculated as multiplier 1 has 50% usage and multipliers 2 and 3 have 100% usage

Milestone 2

The hardware of the second milestone consists of dual port rams, multipliers, adders and multiplexers. The dual port rams store the value of the transposed C matrix, the fetched S’ matrix, the result of the S’C matrix multiplication, and the final result S. These values are obtained by reading the S’ values from the SRAM into the dual port RAMs, then feeding the results of this read along with the predefined C matrix into the multipliers to perform the matrix multiplication before writing the results into a second dual port RAM. Following this the C transpose is a matrix multiplied with the result from the previous multiplication. It should be noted that these results are shifted as necessary to achieve the divisions required. This milestone is also broken into 3 groups of states, which each have 2 subgroups. The lead in case has a group for fetching S' and a group for calculating S'C. The common case has a group for calculating C^T(S'C) and reading another S', and a group for writing S and calculating S'C. The lead out case has a group for calculating one last C^T(S'C) and writing the last S. These states are repeated twice, once for the Y blocks, and a second time for the U and V blocks. The latency can be calculated as follows (2*11*64) + (2*6*64) + 3*2 + (4*64*2*1200) + 2 + (2*4*64*1200) + 2 + 4*64*2 + 2 + 2*2*32 = 1 231 628 clock cycles. The multiplier usage will be equal to 512/516 as the multipliers are used in all of the looped states in the common case, and the three states prior to the fetching state are unused and the 1 state prior to the writing state is unused. This is approximately 99.2% usage.

This milestone is completely incomplete as it stands. There is currently an error which causes the common case to infinitely loop. The rest of this paragraph will be dedicated to explaining the approach used to design the milestone since the debugging process could not be completed. The first important piece of information is that only the transposed C matrix is initialized in the dual port RAM, this is to allow reading the memory linearly as the matrices can be multiplied row by row as the rows of the transpose are equal to the columns of the base matrix. After each row has been read, a check is performed to determine if the end of the block has been reached, if it has, reading and writing will be halted and the multiplications will continue. The result of the multiplication S'C is written as its transpose for the same purpose as storing the transpose C, reading the memory linearly allows for a much simpler indexing method. After all S' values have been fetched and all C^T(S'C) values are calculated in the common case, the next grouping of states for the common case occurs. Similarly, when the block has finished being written to the SRAM, writing is halted and multiplications of S'C continue, which once again are written as their transpose in the dual port RAM. Following this a check is performed to determine whether all of the Y blocks have been dealt with, if so the lead out states begin, performing the final S calculation and writing it to the SRAM before starting from the lead in case once again with the conditions for the U and V blocks enabled. Throughout the design flags are used constantly to determine when to enter and exit states as achieving the required multiplier usage needed the states to be changed at awkward points. The blocks were indexed as follows: the column count would be offset by 8 every time the last row of a block was reached, upon reaching the last block in a row of blocks, the column offset was reset and the maximum row was increased by 8.

Timing Analysis

Inspection of the critical path shows that the common case of milestone 2 is by far the greatest delay in any of the hardware design. Similarly, when omitting the second milestone, milestone 1.

Conclusion

This project allowed the group to create learning experiences in many different skills both technical and practical. On the technical side of the learning experience, the members of the group developed much greater skills in SystemVerilog programming in addition to debugging hardware through simulation. Further, the project demanded extreme critical thinking to reach a solution, although this proved to be incredibly challenging it was also incredibly rewarding and yielded a great improvement in skills oriented around digital systems design. Many practical skills were also bolstered in completing this project. The group developed much greater teamwork skills in addition to problem solving and time management skills. To conclude, despite the incredible challenge posed by this project, the rewards far outweighed any difficulties that which the group faced.

References

a. Karim Mahmoud, “3dq5-2022-project-description”, McMaster Universirty,https://avenue.cllmcmaster.ca/d2l/le/content/479686/viewContent/3913580/View, [accessed: 11, 03, 2022]
b. Karim Mahmoud, “Lab 5”, McMaster Universirty,https://avenue.cllmcmaster.ca/d2l/le/content/479686/viewContent/3913580/View, [accessed: 11, 09, 2022]

c. Karim Mahmoud, “Lecture notes”, McMaster Universirty,https://avenue.cllmcmaster.ca/d2l/le/content/479686/viewContent/3913580/View, [accessed: 11, 16, 2022]