Verilog Sudoku Solver

Magson Gao, Shreeyam Kacker

8								
		3	6					
	7			9		2		
	5				7			
				4	5	7		
			1				3	
		1					6	8
		8	5				1	
	9					4		

Contents

1 Abstract

The aim of this project is to design and implement a Sudoku solver using digital logic written in Verilog HDL. In order to implement this solver it is necessary that the digital logic is written to follow the rules of Sudoku, use several key techniques for solving harder Sudoku and when it is necessary, how to guess values for cells in the Sudoku grid.

The techniques that have been implemented include the candidate line, naked group and hidden group methods. Whilst these methods are sufficient for solving most novice to intermediate level Sudoku, the most difficult Sudoku require backtracking. The previous guesses will be stored on a stack based architecture to allow for previous states of the grid to be restored. The purpose of implementing the aforementioned techniques is to significantly reduce the search space required for guessing, allowing the solution to be found in fewer clock cycles and requiring less memory to store previous guesses for the typical Sudoku. The FPGA that will be used is that on the Digilent Nexys 4 development board.

2 Nomenclature

In this section, the nomenclature used in subsequent sections is defined. The squares containing a single digit in Figure 2.1 is defined as a cell and the 3 by 3 block of cells highlighted by the thickened border is a square. Each 9 cell row and 9 cell column are referred to as a row and column respectively.

	2		5		1		9	
8			2		3			6
	3			6			7	
		1				6		
5	4						1	9
		2				7		
	9			3			8	
2			8		4			7
	1		9		7		6	

Unsolved Sudoku

4	2	*6*	5	*7*	1	*3*	9	*8*
8	*5*	*7*	2	*9*	3	*1*	*4*	6
1	3	*9*	*4*	6	*8*	*2*	7	*5*
9	*7*	1	*3*	*8*	*5*	6	*2*	*4*
5	4	*3*	*7*	*2*	*6*	*8*	1	9
6	*8*	2	*1*	*4*	*9*	7	*5*	*3*
7	9	*4*	*6*	3	*2*	*5*	8	*1*
2	*6*	*5*	8	*1*	4	*9*	*3*	7
3	1	*8*	9	*5*	7	*4*	6	*2*

Solved Sudoku

Figure 2.1: Example unsolved and solved Sudoku puzzles.

3 Goals

Baseline

1. Character recognition from image uploaded into ROM (Shreeyam)
2. Single position solver (Magson)
3. Candidate line solver (Magson)

Expected

1. Naked pair and triple solvers (Magson)
2. Hidden pair and triple solvers (Magson)
3. Video display for solved sudoku puzzle (Shreeyam)
4. Puzzle input using camera and character recognition (Shreeyam)

Stretch

1. Full state space search and backtracking, ability to solve world's hardest sudoku (Magson)
2. Tutorial mode that shows if a guess is correct or not (Shreeyam)
3. Tutorial mode that shows required techniques to solve each square (Shreeyam)

4 Block Diagrams

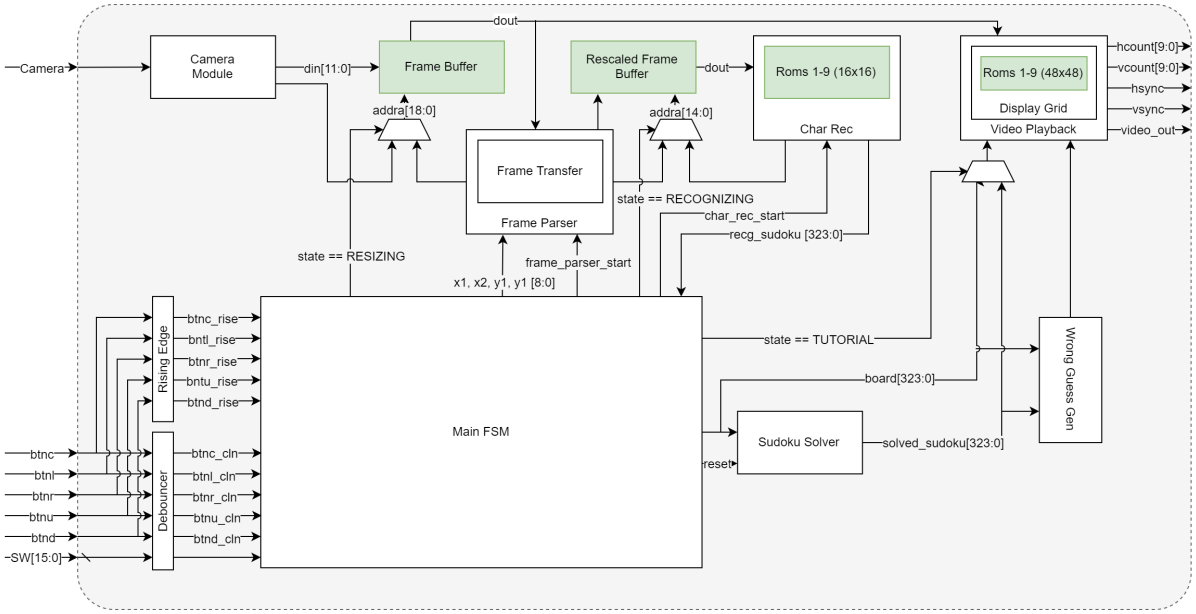

Figure 4.1: Complete diagram of sudoku solver system.

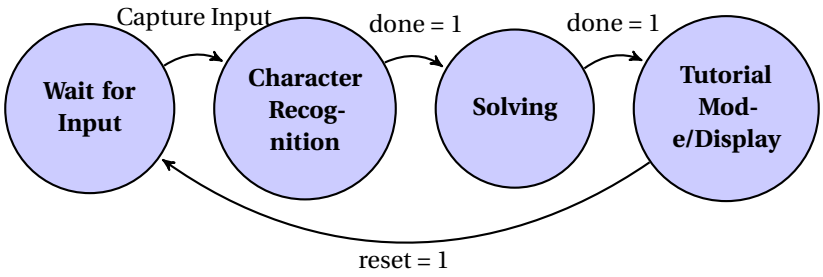

Figure 4.2: Overall functional diagram showing flow between overall states, with tutorial mode if such a stretch goal is implemented, otherwise displaying the solution on screen,

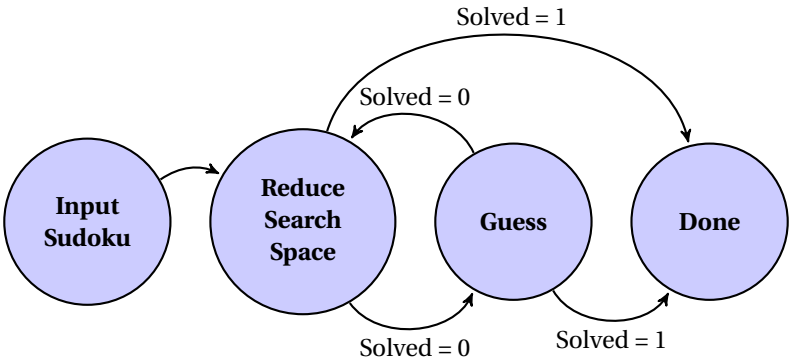

Figure 4.3: The high level functional diagram for the Sudoku solver, showing high level states.

5 Design and Overview

The first choice that we had to make was whether to use the Labkit or the Nexys 4 development board. It became clear that although the labkit had more BRAM, the additional LUTs available on the Nexys 4 were more desireable if we wanted to make a fast solver. In the end, it turned out the number of LUTs we ended up using was almost 50% more than the amount available on the labkit, so this ended up being a wise decision.

Beyond that, the only other major design decision that made us decide between the labkit and Nexys 4 was the 12-bit color on the Nexys 4 and the 24-bit color on the labkit. In the end we decided that the 12-bit color on the Nexys 4 provided sufficient granularity to threshold numbers accurately.

6 Subsystem Overview

Camera (`camera.v`)

The code to interface to the camera was provided by Weston Braun. This was simply reorganized into a module `camera.v` in order to compartmentalize code and keep it independent of everything else by reducing the number of interconnects in the top level module.

The camera continually writes to a high resolution frame buffer in BRAM and stores data as 12-bit RGB values for each pixel, until it is told to stop. After that, the data in the frame buffer remains static, and additional processing can be done on it.

Main FSM (`top.v`, Shreeyam)

The main FSM for the entire system was kept in the top level file for the project, since it interacted with many of the inputs. This simplified the architecture of the system compared to having a separate module with a large amount of inputs.

Number	Name	Description
0	IDLE	Camera input is written to high resolution frame buffer, which is then directly output to the screen.
1	CHOOSE_XY1	Camera stops writing to frame buffer to effectively capture a picture, and first pair of crosshairs is displayed on the screen to identify top-left corner of puzzle.
2	CHOOSE_XY2	Second pair of crosshairs is displayed to identify bottom-right corner of sudoku puzzle.
3	RESIZING	Control of memory address is relinquished from video output module and passed to frame parser module. Image is then rescaled and written to low resolution frame buffer.
4	RECOGNIZING	Control of memory address of low resolution frame buffer is relinquished from frame parser module and given to character recognition module. Character recognition sequence then begins over the expected position of each cell.
5	CONFIRMING	Unused state. Originally used to nudge selected coordinates of puzzle one pixel in each direction to see how it would change character recognition output. Taken out due to difficutly of automatically starting modules in the FSM.
6	FIXING	Output is shown as recognized sudoku puzzle. User can now correct mistakes over the board with the selected cell being highlighted.
7	SOLVING	Input is passed to sudoku solver to start solving board.
8	OUTPUT	Change video output from recognized sudoku puzzle to solved sudoku puzzle, without any green selection box.
9	TUTORIAL	Change video output back from solved board to input board. User can input numbers into this board and visual feedback is displayed if the input number is different from that in the solved board.

Table 6.1: List of all states in FSM.

The entire FSM is linear in that one state very simply flows from one to another on the same key, with each state triggering a particular action.

Frame Parser (`frame_parser.v`, Shreeyam)

The frame parser was responsible for rescaling the image to a fixed size and transferring that data to a second, lower resolution frame buffer. The advantage of doing this is that now the image is of a known size, rather than having to dynamically calculate where each cell would be given that it occupies some width and height in the high resolution frame buffer.

The frame parser has its own finite state machine that controls its flow. Initially the distances in the x and y directions are divided by the desired width and height of the image, and the quotient and remainder are stored. To simplify things, the desired width and height were set to be the same value, since the grid is square. This was value was deemed to be 144 pixels square, since this represents a 9 by 9 sudoku grid, with each cell occupying a 16 by 16 pixel region.

After doing this, the address of the high resolution frame buffer is scanned across. Each time, the quotient is added to the coordinate and the remainder is added to an accumulator. If the accumulator becomes larger than the desired size, the skip size is incremented by an extra value, and the accumulator has the desired size subtracted from it. This is analogous to continually adding the remainder as fraction and adding an extra pixel once the sum overflows a value of unity, in order to correct the drift that comes from simply adding the rounded-down quotient.

While scanning across each address, values are passed from the high resolution frame buffer and then written to the low resolution frame buffer, obeying any timing requirements that they might have. In this case, data appeared at the output of the high resolution frame buffer two clock cycles delayed, and writing to the low resolution frame buffer had the data appear delayed by a single clock cycle.

Character Recognition (`char_rec.v`, Shreeyam)

Character recognition works by convolving each 16 by 16 cell in the low resolution frame buffer with a ROM of each of the numbers from 1 to 9. From there, a probability is assigned given how many of the pixels matched with the ROM, and the highest probability number is selected.

In order to make full use of the zero bus turnaround memory, it is advantageous to continually read or write to it whenever it is operational. For this reason we make sure to continually scan lines across the low resolution frame buffer and avoid incorporating clock cycle delays anywhere.

Modern character recognition techniques are much more advanced, employing neural networks or hidden markov models, but this simple template matching technique was deemed sufficient due to the fact that the inputs are of a very closed form, since we controlled those completely. However, this came with its own difficulties and challenges that will be discussed later.

The first thing we to do was choose a font that we would design our character recognition around. Through preliminary research we found that the fonts Consolas, OCR-A, and OCR-B are all commonly used to be friendly to optical character recognition based systems. OCR-A is also commonly known as the font used on checkbooks.

Originally we experimented with using OCR-A and Consolas exclusively, because OCR-B required payment to obtain. Between these two options, Consolas proved to be more applicable to our constraints, since OCR-A was so thin that it was hard to scale to 16 pixels wide and still be easily distinguishable from other characters after dilation, and this made the alignment much more difficult as well.

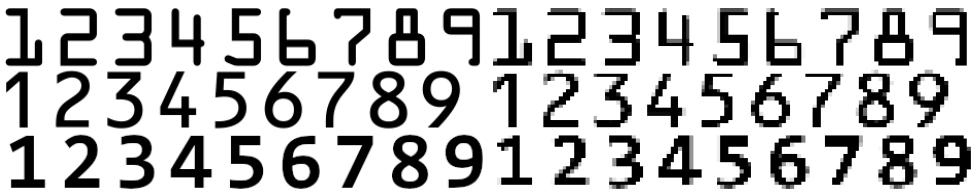

Figure 6.1: Comparison of original size and nearest-neighbour rescaled fonts (from top to bottom): OCR-A, OCR-B, Consolas

Video Playback (`video_playback.v`, Shreeyam)

The video playback module handles all the interfacing between the internal state and logic of the solver with the only output device, the video display.

Initially this module started off as Weston Braun's code to read back data from the camera frame buffer, but we expanded it to also be able to display the selection crosshairs, sudoku grid, selected cell, and tutorial mode.

The display of the characters was done using the same techniques as in Lab 3 enumerated for all 81 cells, with the output depending on the value of the cells. The characters can also be switched to images of the staff of 6.111, which we had to do since there were nine in total. To save BRAM and additional complexity, these images were rendered in high definition one-bit dithered black and white.

Sudoku Solver (Magson)

An initial implementation which was considered used a naive binary representation of each number on the Sudoku grid. The hardware requirements to determine which numbers were missing a row, column or square would involve 9 comparators for each cell in the Sudoku grid as each cell can take a value from 1 to 9. For each of the 27 sets of cells of cardinality 9 there would be 81 comparators producing 9 single bit results for each of the 9 cells in the set. In order to determine if a set consists of a solved set, 9×9 input OR operations are required. This requirement can be reduced significantly be realizing that the output of the comparators are in fact the one hot encoding of the values in the cell. Thus the solver can instead store each number of each cell as a one hot code, thus removing the need for comparators entirely.

For the chosen implementation the number of bits required to represent a Sudoku grid is 81×9 bits. In order to determine which numbers are missing in the 27 sets, the solver requires $27 \times 9 \times 8$ OR operations (each containing 2 single bit inputs). This is because 8 OR gates are required to compute the OR operation between 9 single bit numbers representing the presence of 1 of 9 possible numbers. This is implementable using $27 \times 9 \times 2$ 6-input look-up tables in the worst case.

To implement the advanced techniques which apply to the possible values of each cell, each cell requires a mask of size 9 bits, formed from AND-ing the 3 masks generated from the row, column and square sets. This results in 9×3 input AND operations, which is implementable using (in the worst case) 9×6-input look-up tables. Each mask is modified further using the advanced techniques.

The candidate line technique can be implemented by storing 6 masks, 3 for each column and 3 for each row associated with each of the 9 squares of a Sudoku grid. Each mask contains the possible values which can exist in the row or column. This is computed by OR operations over each of the 3×9 bit masks. By computing the AND operation of each of the 6 masks which every other mask in the row or column (requiring $6 \times 2 \times 9$ bit AND operations per square), a candidate line can be identified by observing all zeros as a result of the 2 AND operations.

The mask modifications from the advanced techniques require significant computation and thus parts of the computation have been done sequentially. In terms of digital logic, the hidden group problem is in fact a super-set of the naked group problem and can be solved using the following algorithm:

1. Assume each cell has a 9 bit mask containing 1s for values which it can take and 0s for values which it cannot take.
2. Using a 4 bit output adder, compute the number of 1s in each mask.
3. For each set in the 27, AND each bit of the mask with the bits in each of the other masks. For each set store the 8 resulting masks.
4. If the number of stored masks which are 0 are equal to $(9 - number\ of\ ones\ in\ initial\ mask)$ and not all the stored masks are 0 then a hidden group is detected.
5. If this is the case, replace the masks contained in the hidden group with the stored masks.
6. If the number of resulting masks containing the same number of 1s as the initial mask is 1 less than the number of 1s in the initial mask a naked group is inferred.
7. If this is the case AND the masks outside the hidden group with the NOT of the initial mask.

The required hardware for each of the steps is as follows:

1. 81×9 bit masks.
2. The number of 1s in each mask can range from 1 to 9. A simple implementation will use 4×6 input look-up tables. Thus for all cells in a set, 72 look-up tables are required.

3. For each set in the 27, this step requires 72 × 9 × 2-bit AND operations.

4. This step requires approximately 27 × 2 look-up tables. This is because there are 27 masks per set each of size 9 bits. Thus if 2 × 6 input look-up tables are used per mask (this is an over-estimate), 54 look-up tables are required in total. Counting the number of 0 masks requires approximately 5 × 6 input look-up tables. The comparisons (with 0 and with $9 - number\ of\ ones\ in\ initial\ mask$) requires an additional 2 look-up tables.

5. The replacement operation requires a write enable signal on all the masks.

6. This requires an additional 72 look-up tables to determine the number of 1s in each of the stored masks and an additional 81 × 9 bit masks. In addition equality must be checked with 9 × 8 pairs of values. The outputs are summed up using at most 9 × 4 look-up tables and an additional equality operation is required to compare the number of matches with the number of 1s in the initial mask.

It is worth noting that whilst the above technique can detect all naked groups and all hidden pairs it cannot in general detect all hidden groups. This is because a hidden group can be formed from a set of numbers which is not present (in its entirety) in any of the members of the hidden group. One possible method of solving this issue is to compare each mask with all possible combinations of pairs, triples and quadruples. Used in tandem with the above technique allows all hidden groups up to cardinality 5 to be detected. Beyond cardinality 5 the algorithm has diminishing returns as the number of bits in the mask which can be set to 0 is reduced.

The state space search uses a stack based approach. There is first of all a 3D array called the previous values register (`pvr`) which contains a mask of all possible values for each cell. There is also a 4D array called `pvr_prevs` which acts like a stack, storing the possible values for each backtracking step.

During normal operation, the possible values register is iterated through, and the cell with the least number of possibilities is kept. Each cell has a countdown register which tracks when changes happen. When the countdown register for each cell has exceeded its maximum value of 81 (i.e. the cell has not changed while cycling over the entire grid), the guessing algorithm begins operation. The top of the `pvr_prevs` stack instantiates a guess and removes the least significant high bit of that cell (e.g. if the `pvr` cell has value `011000010`, `pvr_prevs` will store `011000000`) by setting it to zero. The row, column, and values being guessed are stored in three different stacks in a similar configuration.

The possible values register is then changed so that the guessed cell has a single value. By selecting the cell with the smallest number of possible values, the likelihood of an incorrect guess is reduced, as well as the initial branching factor. The least significant high bit is extracted in a single clock cycle by abusing two's complement, with the operation $M\&-M$. The done counter is then reset to 81 and normal combinatorial operation resumes.

If can error is detected, i.e. when there are no possible values for given cell, or a row or column is full of numbers but not solved, the possible values register is replaced by the top of `pvr_prevs`, effectively backtracking to a previous state. The stack pointer of `pvr_prevs` is also decremented, which changes where the possible values register will be stored next.

7 Testing and Debugging

The solver module was evaluated by a testbench containing a set of multiple randomly generated Sudoku for each difficulty level. Each Sudoku was generated using the QQWing Sudoku generator [3] as it provides not only the Sudoku itself but a list of the required algorithms used to solve the Sudoku puzzle and the number of required guesses. The testbench also included the "World's Hardest Sudoku" [1] which is one of the approximately, 50,000 minimum hint Sudoku each of which contain only 17

of the 81 cells pre-filled [2]. The testbench contains 5 modes: simple, easy, intermediate, hard, and world's hardest sudoku.

The simple mode tests the basic solver functionality to represent the grid as a one hot code and compute row, column and square solves when only single possible value can be inserted. The easy mode tests this with real world Sudoku generated from the QQWing generator where multiple possible values are initially possible for each square but advanced techniques are not required. The intermediate Sudoku require these advanced techniques but do not require backtracking. The hard Sudoku require all the aforementioned techniques as well as backtracking. The solver was tested in order of increasing difficulty as it will be difficult to determine which technique is used to solve the Sudoku (any Sudoku can be solved given a long enough time and sufficient memory using backtracking).

Due to the way the architecture was split between the main FSM and group FSM, an additional test bench was created with puzzles designed to invoke the group FSM.

Every other module besides the character recognition and camera module were evaluated by their own test bench. This was especially useful when trying to get the frame rescaling working, we debugged by matching the simulation waveform with values that we'd expect from MATLAB. Beyond that, the hex display was useful for determining state transitiong, as well as piping outputs to physical pins on the development board and looking at their signals on an oscilloscope as sanity checks.

8 Challenges and Improvements

The largest challenge with the sudoku solver was dealing with available area on the FPGA. As taught in lectures, throwing more hardware at a problem generally makes it faster, but there is a hard limit on the Nexys 4 on the amount of hardware that is available. This became especially more difficult when incorporating guessing, as the register which stores the previous states is actually a 4-dimensional array, and could have easily blown up in size if a larger depth was desired. The size of this buffer was determined by seeing how many backtracking steps the world's hardest sudoku required, which is six, then doubling it and rounding it to the next power of two for good measure, in order to make sure no possible puzzle would be able to overflow the previous guess register. One humorous side effect of this is that when given an empty board, the solver will deterministically make up a sudoku puzzle and then solve it automatically.

Resource	Utilization	Available	Percent Utilization
LUT	42726	63400	67.39117
LUTRAM	3	19000	0.015789473
FF	16691	126800	13.163249
BRAM	125	135	92.59259
DSP	6	240	2.5
IO	58	210	27.61905
BUFG	3	32	9.375
MMCM	1	6	16.666668

Table 8.1: Total device utilization

Area also limited the hidden group algorithm significantly. There was not sufficient area available on the FPGA to incorporate the hidden group algorithm into the main finite state machine that each cell has, so this had to be split up into a separate state machine that scanned across each cell sequentially. This then required interfacing with the state machine that operates across all cells, via the group mask register.

Area also meant that the character recognition had to switch from being one long generate loop to a sequential implementation, however this happened so fast anyways compared to a human's perception, this was deemed non-essential to the goal of this project.

If a guess is incorrect, since each cell is operating in parallel, it's possible that the same two numbers can be placed simultaneously. Essentially a race condition can occur between guesses, which yielded and invalid board. In previous versions of the code before the guessing algorithm was implemented, this was mitigated by checking whether a number could be placed in that cell. However with the case where an invalid guess is made and two cells take the same value leading to an incorrect solution, both cells can take those values despite being invalid, and so an extra error detection condition where if a row is full but not solved was added.

Interfacing two-dimensional arrays between the solver and all other modules was also a very big challenge. Verilog does not let you pass two-dimensional arrays between modules, hence a lot of extra code had to be written to flatten and unflatten arrays. There were also issues with endianness, which caused the sudoku puzzle to be output as the transpose of what it actually was. This required a last minute code change over a phone call to resolve, but could have completely been avoided in SystemVerilog.

The hidden group algorithm can also only solve for cells where at least one cell contains the complete list of possibilities. However, we found that despite this, the algorithm found a large enough amount of cases that its implementation was worthwhile. The larger the group size, the less likely it can be solved exclusively using these group techniques and they will become less and less common. Our technique works for all pairs and the majority of triples, which are the most common group sizes.

It also turned out to be a big challenge trying to keep track of which techniques solved a particular cell. This is because a cell can often be solved using multiple techniques, and this would require interfacing between the 81 finite state machines used to solve the board. This would have required a massive overhaul of the solver architecture, potentially making it much slower as a result. Hence, we chose to foregoe this stretch goal in lieu of additional speed.

The frame parser was difficult to debug since conceptually it was hard to deal with the extra delay incurred by the fixed two clock cycle delay on the BRAM, especially given that this happened twice since data was transferred from one block of BRAM to another. However, when data was not transferred properly from one block of BRAM to another, it was never due to the delay being improperly implemented. Rather, it was due to other errors, such as registers not being reset correctly on new scan lines that caused a lot of the errors in this module.

The character recognition system also found distinct troubles when dealing with real world inputs compared to images loaded into BRAM. Unfortunately, the real world is much more uneven than nice test patterns generated using MATLAB. It was incredibly difficult to get an evenly lit image of the sudoku puzzle first of all, which made thresholding very difficult to do. Camera noise also contributed this causing specks around characters, but some erosion and dilation could have fixed this relatively easily.

Secondly, barrel distortion added an unexpectedly large amount of curvature to the image that could not be corrected for very easily. Combined with the very alignment-sensitive character recognition technique, this meant that correctly matching more than the first few cells of a puzzle were almost impossible, as cells would be guaranteed to be out of alignment by the end anyways, even if the image loaded into BRAM could be matched exactly.

Misalignment by the human operator could also very easily cause cells to be just out of alignment. In our own testing we found that the tolerance of alignment was just 3 pixels or so, and this would apply to both crosshairs. If one crosshair was misaligned, this would be similar to how tuning a guitar string by string would cause error to build up and eventually cause incorrect results. We wanted to

add a feature that would allow the selection to be 'nudged' by one pixel in each direction or so but due to issues with memory and timing, this was deemed out of scope and not essential to the core functionality of the project.

9 Conclusion

Overall, this project was a success. Without any additional optimizations we were able to solve the world's hardest sudoku in just under 10,000 clock cycles, which is approximately 300 µs. This makes this the world's fastest sudoku solver.

For many simple puzzles, it was possible to solve the vast majority of them in under 10,000 clock cycles.

For future students of 6.111, it cannot be reiterated enough, but it's completely worth saving the stress and detriment to your wellbeing by starting as early as possible. It's easy to get comfortable with the reduced lab hours, but this just leads to sleepless nights later on. Especially given our project for which the solver either works it doesn't, being able to start early and get everything integrated as soon as possible was ideal, as this allowed to truly refine our project to become the world's fastest sudoku solver. One of the largest benefits of starting early too is that you have the liberty of being able to literally sleep on problems and then approach them later with a fresh mind, which generally allows you to solve whatever issue that you're having much faster overall.

It's also ideal to do as little integration as possible on the project. Often times integration can lead to bugs that require a lot of tracing back to code that you may have written weeks ago. Magson gave an incredibly simple interface to his Sudoku solver that allowed the integration to be done over a single day, while many projects had to deal with integration issues and faced the asymmetry of the fact that writing code is much easier than reading code.

10 Acknowledgements

We'd like to thank all the staff of 6.111 who taught us Verilog so well and helped debug our code and save us when we were truly in massive trouble.

We'd also like to thank the MIT Global Education office and our coordinators at Imperial College for having us here. Without being given such a great opportunity like this we would not have been able to create something that we're so proud of.

Shreeyam

6.111 has been the best course I've ever taken in my life without question and has been exactly what I've wanted to get out of my education at MIT. Thanks again!

11 Bibliography

[1] Artika Inkala. In: (2010). URL: http://www.aisudoku.com/index_en.html.

[2] Gary McGuire, Bastian Tugemann, and Gilles Civario. “There is no 16-Clue Sudoku: Solving the Sudoku Minimum Number of Clues Problem”. In: *CoRR* abs/1201.0749 (2012). arXiv: 1201.0749. URL: http://arxiv.org/abs/1201.0749.

[3] Stephen Ostermiller. In: (). URL: https://qqwing.com/generate.html.

A Verilog

```
module camera(
    input video_clk,
    input camera_start,
    input sioc,
    input siod,
    input capture_frame,
    input camera_vsync,
    input camera_hsync,
    input[7:0] camera_dout,
    input camera_clk,
    output[15:0] camera_pixel,
    output camera_pixel_valid,
    output camera_frame_done,
    output[11:0] memory_write_data,
    output[18:0] memory_write_addr,
    output memory_write_enable
);


//camera configuration module
camera_configure camera_configure_1 (
    .clk(video_clk),
    .start(camera_start),
    .sioc(sioc),
    .siod(siod),
    .done()
    );

//camera interface
camera_read camera_read_1 (
    .p_clock(camera_clk),
    .vsync(camera_vsync),
    .href(camera_hsync),
    .p_data(camera_dout),
    .pixel_data(camera_pixel),
    .pixel_valid(camera_pixel_valid),
    .frame_done(camera_frame_done)
    );


//write camera data to frame buffer
camera_address_gen camera_address_gen_1 (
    .camera_clk(camera_clk),
    .camera_pixel_valid(camera_pixel_valid),
    .camera_frame_done(camera_frame_done),
    .capture_frame(capture_frame),
    .camera_pixel(camera_pixel),
    .memory_data(memory_write_data),
    .memory_addr(memory_write_addr),
    .memory_we(memory_write_enable)
    );

```

```
endmodule

`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/07/2015 10:20:20 AM
// Design Name:
// Module Name: camera_configure
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module camera_configure
    #(
    parameter CLK_FREQ=25000000
    )
    (
    input wire clk,
    input wire start,
    output wire sioc,
    output wire siod,
    output wire done
    );

    wire [7:0] rom_addr;
    wire [15:0] rom_dout;
    wire [7:0] SCCB_addr;
    wire [7:0] SCCB_data;
    wire SCCB_start;
    wire SCCB_ready;
    wire SCCB_SIOC_oe;
    wire SCCB_SIOD_oe;

    assign sioc = SCCB_SIOC_oe ? 1'b0 : 1'bZ;
    assign siod = SCCB_SIOD_oe ? 1'b0 : 1'bZ;

    OV7670_config_rom rom1(
        .clk(clk),
        .addr(rom_addr),
        .dout(rom_dout)
        );

```

```
    OV7670_config #(.CLK_FREQ(CLK_FREQ)) config_1(
        .clk(clk),
        .SCCB_interface_ready(SCCB_ready),
        .rom_data(rom_dout),
        .start(start),
        .rom_addr(rom_addr),
        .done(done),
        .SCCB_interface_addr(SCCB_addr),
        .SCCB_interface_data(SCCB_data),
        .SCCB_interface_start(SCCB_start)
        );

    SCCB_interface #( .CLK_FREQ(CLK_FREQ)) SCCB1(
        .clk(clk),
        .start(SCCB_start),
        .address(SCCB_addr),
        .data(SCCB_data),
        .ready(SCCB_ready),
        .SIOC_oe(SCCB_SIOC_oe),
        .SIOD_oe(SCCB_SIOD_oe)
        );

endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date:    21:12:24 12/03/2014
// Design Name:
// Module Name:    camera_read
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////

module camera_read(
        input wire p_clock,
        input wire vsync,
        input wire href,
        input wire [7:0] p_data,
        output reg [15:0] pixel_data =0,
        output reg pixel_valid = 0,
        output reg frame_done = 0
    );

```

```

        reg [1:0] FSM_state = 0;
    reg pixel_half = 0;

        localparam WAIT_FRAME_START = 0;
        localparam ROW_CAPTURE = 1;


        always@(posedge p_clock)
        begin

        case(FSM_state)

        WAIT_FRAME_START: begin //wait for VSYNC
           FSM_state <= (!vsync) ? ROW_CAPTURE : WAIT_FRAME_START;
           frame_done <= 0;
           pixel_half <= 0;
        end

        ROW_CAPTURE: begin
           FSM_state <= vsync ? WAIT_FRAME_START : ROW_CAPTURE;
           frame_done <= vsync ? 1 : 0;
           pixel_valid <= (href && pixel_half) ? 1 : 0;
           if (href) begin
               pixel_half <= ~ pixel_half;
               if (pixel_half) pixel_data[7:0] <= p_data;
               else pixel_data[15:8] <= p_data;
           end
        end


        endcase
        end

endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/26/2018 08:32:13 PM
// Design Name:
// Module Name: char_rec
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
```

```
    //////////////////////////////////////////////////////////////////////////////////
21
22
23  module char_rec(
24          input clk,
25          input start,
26          output done,
27
28          output reg[14:0] img_ram_addr,
29          input[11:0] img_ram_data,
30
31          output reg[323:0] recg_sudoku
32
33      );
34
35  parameter IMG_WIDTH = 144;
36  parameter DELAY_CYCLES = 2;
37
38  // States
39  parameter IDLE = 0;
40  parameter RECG = 1;
41  parameter RECONF = 2;
42
43  reg[3:0] state = 0;
44
45  reg[11:0] cycle_counter = 0;
46  reg[8:0] mem_addr_counter = 0;
47  reg[8:0] x0 = 0;
48  reg[8:0] y0 = 0;
49  reg[4:0] hcount = 0;
50  reg[4:0] vcount = 0;
51
52  wire[8:0] x = hcount + x0;
53  wire[8:0] y = vcount + y0;
54
55  // Probabilities
56
57  reg[8:0] none_score = 0;
58  reg[8:0] one_score = 0;
59  reg[8:0] two_score = 0;
60  reg[8:0] three_score = 0;
61  reg[8:0] four_score = 0;
62  reg[8:0] five_score = 0;
63  reg[8:0] six_score = 0;
64  reg[8:0] seven_score = 0;
65  reg[8:0] eight_score = 0;
66  reg[8:0] nine_score = 0;
67
68  //assign img_ram_addr = x + (y * IMG_WIDTH);
69
70  parameter THRESHOLD = 20;
71  parameter CELL_LR_WIDTH = 16;
72
73  wire[5:0] combined_pixel_data = img_ram_data[3:0] + img_ram_data[7:4] +
        img_ram_data[11:8];
74  wire pix_logical = combined_pixel_data > THRESHOLD;
75
```

```
76  wire[3:0] max_score;
77
78  max_score max_score_1 (
79      .none_score(none_score),
80      .one_score(one_score),
81      .two_score(two_score),
82      .three_score(three_score),
83      .four_score(four_score),
84      .five_score(five_score),
85      .six_score(six_score),
86      .seven_score(seven_score),
87      .eight_score(eight_score),
88      .nine_score(nine_score),
89      .max_score_out(max_score));
90
91  // ROMs
92
93  wire one_d;
94  wire two_d;
95  wire three_d;
96  wire four_d;
97  wire five_d;
98  wire six_d;
99  wire seven_d;
100 wire eight_d;
101 wire nine_d;
102
103 reg[7:0] rom_addr = 0;
104
105 one_16_d one_16_d_1 (
106     .clka(clk),
107     .addra(rom_addr),
108     .douta(one_d));
109
110 two_16_d two_16_d_1 (
111     .clka(clk),
112     .addra(rom_addr),
113     .douta(two_d));
114
115 three_16_d three_16_d_1 (
116     .clka(clk),
117     .addra(rom_addr),
118     .douta(three_d));
119
120 four_16_d four_16_d_1 (
121     .clka(clk),
122     .addra(rom_addr),
123     .douta(four_d));
124
125 five_16_d five_16_d_1 (
126     .clka(clk),
127     .addra(rom_addr),
128     .douta(five_d));
129
130 six_16_d six_16_d_1 (
131     .clka(clk),
132     .addra(rom_addr),
133     .douta(six_d));
```

```
134
135 seven_16_d seven_16_d_1 (
136     .clka(clk),
137     .addra(rom_addr),
138     .douta(seven_d));
139
140 eight_16_d eight_16_d_1 (
141     .clka(clk),
142     .addra(rom_addr),
143     .douta(eight_d));
144
145 nine_16_d nine_16_d_1 (
146     .clka(clk),
147     .addra(rom_addr),
148     .douta(nine_d));
149
150 always @(posedge clk) begin
151     img_ram_addr <= x + y * IMG_WIDTH;
152     rom_addr <= ((hcount < 16) && (vcount < 16)) ? hcount + (vcount *
        CELL_LR_WIDTH) : 255;
153     case(state)
154         IDLE: begin
155             if(start) begin
156                 state <= RECG;
157                 cycle_counter <= 0;
158                 mem_addr_counter <= 0;
159                 recg_sudoku <= 0;
160                 x0 <= 0;
161                 y0 <= 0;
162                 hcount <= 0;
163                 vcount <= 0;
164
165                 none_score  <= 0;
166                 one_score   <= 0;
167                 two_score   <= 0;
168                 three_score <= 0;
169                 four_score  <= 0;
170                 five_score  <= 0;
171                 six_score   <= 0;
172                 seven_score <= 0;
173                 eight_score <= 0;
174                 nine_score  <= 0;
175             end
176         end
177         RECG: begin
178             mem_addr_counter <= mem_addr_counter + 1;
179
180             if (hcount < CELL_LR_WIDTH - 1) begin
181                 hcount <= hcount + 1;
182             end else begin
183                 hcount <= 0;
184                 vcount <= vcount + 1;
185             end
186
187             // Main recognition loop
188             if(mem_addr_counter == 256 + DELAY_CYCLES) begin
189                 state <= RECONF;
190             end else
```

```
191
192             if(mem_addr_counter >= DELAY_CYCLES) begin
193                 none_score  <= none_score   + (pix_logical == 1);
194                 one_score   <= one_score    + ~(pix_logical == one_d);
195                 two_score   <= two_score    + ~(pix_logical == two_d);
196                 three_score <= three_score  + ~(pix_logical == three_d)
                        ;
197                 four_score  <= four_score   + ~(pix_logical == four_d);
198                 five_score  <= five_score   + ~(pix_logical == five_d);
199                 six_score   <= six_score    + ~(pix_logical == six_d);
200                 seven_score <= seven_score  + ~(pix_logical == seven_d)
                        ;
201                 eight_score <= eight_score  + ~(pix_logical == eight_d)
                        ;
202                 nine_score  <= nine_score   + ~(pix_logical == nine_d);
203             end
204         end
205         RECONF: begin
206             // Shift
207             recg_sudoku <= {max_score, recg_sudoku[323:4]};
208             mem_addr_counter <= 0;
209             hcount <= 0;
210             vcount <= 0;
211
212             none_score  <= 0;
213             one_score   <= 0;
214             two_score   <= 0;
215             three_score <= 0;
216             four_score  <= 0;
217             five_score  <= 0;
218             six_score   <= 0;
219             seven_score <= 0;
220             eight_score <= 0;
221             nine_score  <= 0;
222
223             if(x0 <= 144 - CELL_LR_WIDTH - 1) begin
224                 x0 <= x0 + CELL_LR_WIDTH;
225
226                 state <= RECG;
227             end else if(y0 <= 144 - CELL_LR_WIDTH - 1) begin
228                 y0 <= y0 + CELL_LR_WIDTH;
229                 x0 <= 0;
230
231                 state <= RECG;
232             end else begin
233                 state <= IDLE;
234             end
235         end
236     endcase
237 end
238
239 assign done = ~start && (state == IDLE);
240
241 endmodule
```

```
1 `timescale 1ns / 1ps
2 //
    //////////////////////////////////////////////////////////////////////////////////
```

```
// Company:
// Engineer:
//
// Create Date: 11/20/2018 10:57:07 PM
// Design Name:
// Module Name: clk_prescale
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module clk_prescale(
        input clk,
        output clk_ps
    );

reg[16:0] counter = 0;

always @(posedge clk) begin
    if(counter == 120000) begin
        counter <= 0;
    end else begin
        counter <= counter + 1;
    end
end

assign clk_ps = (counter == 120000);
endmodule
```

```
localparam   GRID_SIZE = 9;
localparam   MAX_GUESSES = 16;

`define GET_LSB(M)\
-M & M

`define MAX(N1, N2)\
((N1 > N2) ? N1 : N2)

`define MAX_OUT_OF_NINE(N1, N2, N3, N4, N5, N6, N7, N8, N9)\
`MAX(N1, `MAX(N2, `MAX(N3, `MAX(N4, `MAX(N5, `MAX(N6, `MAX(N7, `MAX(N8,
    N9) ) ) ) ) ) ) )

`define INST_ROM(TYPE, NAME, CLK, ADDR, OUT)\
TYPE NAME (\
        .clka(CLK),\
        .addra(ADDR),\
```

```
        .douta(OUT)\
)

function automatic [3:0] bcd;
input [8:0] one_hot_in;
begin
        case(one_hot_in)
                9'b0_0000_0001  : bcd = 4'd1;
                9'b0_0000_0010  : bcd = 4'd2;
                9'b0_0000_0100  : bcd = 4'd3;
                9'b0_0000_1000  : bcd = 4'd4;
                9'b0_0001_0000  : bcd = 4'd5;
                9'b0_0010_0000  : bcd = 4'd6;
                9'b0_0100_0000  : bcd = 4'd7;
                9'b0_1000_0000  : bcd = 4'd8;
                9'b1_0000_0000  : bcd = 4'd9;
                default                 : bcd = 4'd0;
        endcase
end
endfunction
```

```
// Switch Debounce Module
// use your system clock for the clock input
// to produce a synchronous, debounced output
module debounce #(parameter DELAY=250000)   // .01 sec with a 100Mhz
    clock
                (input reset, clk, noisy,
                 output reg clean);

   reg [19:0] count = 0;
   reg new;

   always @(posedge clk)
     if (reset)
       begin
          count <= 0;
          new <= noisy;
          clean <= noisy;
       end
     else if (noisy != new)
       begin
          new <= noisy;
          count <= 0;
       end
     else if (count == DELAY)
       clean <= new;
     else
       count <= count+1;

endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:   g.p.hom
// Engineer:
//
```

```
// Create Date:    18:18:59 04/21/2013
// Module Name:    display_8hex

// Description:  Display 8 hex numbers on 7 segment display
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module display_8hex(
    input clk,                  // system clock
    input [31:0] data,          // 8 hex numbers, msb first
    output reg [6:0] seg,       // seven segment display output
    output reg [7:0] strobe     // digit strobe
    );

    localparam bits = 13;

    reg [bits:0] counter = 0;  // clear on power up

    wire [6:0] segments[15:0]; // 16 7 bit memorys
    assign segments[0]  = 7'b100_0000;
    assign segments[1]  = 7'b111_1001;
    assign segments[2]  = 7'b010_0100;
    assign segments[3]  = 7'b011_0000;
    assign segments[4]  = 7'b001_1001;
    assign segments[5]  = 7'b001_0010;
    assign segments[6]  = 7'b000_0010;
    assign segments[7]  = 7'b111_1000;
    assign segments[8]  = 7'b000_0000;
    assign segments[9]  = 7'b001_1000;
    assign segments[10] = 7'b000_1000;
    assign segments[11] = 7'b000_0011;
    assign segments[12] = 7'b010_0111;
    assign segments[13] = 7'b010_0001;
    assign segments[14] = 7'b000_0110;
    assign segments[15] = 7'b000_1110;

    always @(posedge clk) begin
      counter <= counter + 1;
      case (counter[bits:bits-2])
          3'b000: begin
                  seg <= segments[data[31:28]];
                  strobe <= 8'b0111_1111 ;
                 end

          3'b001: begin
                  seg <= segments[data[27:24]];
                  strobe <= 8'b1011_1111 ;
                 end

          3'b010: begin
                   seg <= segments[data[23:20]];
                   strobe <= 8'b1101_1111 ;
                  end
```

```
62            3'b011: begin
63                    seg <= segments[data[19:16]];
64                    strobe <= 8'b1110_1111;
65                   end
66            3'b100: begin
67                    seg <= segments[data[15:12]];
68                    strobe <= 8'b1111_0111;
69                   end
70
71            3'b101: begin
72                    seg <= segments[data[11:8]];
73                    strobe <= 8'b1111_1011;
74                   end
75
76            3'b110: begin
77                     seg <= segments[data[7:4]];
78                     strobe <= 8'b1111_1101;
79                    end
80            3'b111: begin
81                    seg <= segments[data[3:0]];
82                    strobe <= 8'b1111_1110;
83                   end
84
85
86
87
88          endcase
89        end
90
91  endmodule
```

```
1   // The divisor module divides one number by another. It
2   // produces a signal named "ready" when the quotient output
3   // is ready, and takes a signal named "start" to indicate
4   // the the input dividend and divisor is ready.
5   // sign -- 0 for unsigned, 1 for twos complement
6
7   // It uses a simple restoring divide algorithm.
8   // http://en.wikipedia.org/wiki/Division_(digital)#Restoring_division
9
10  module divider #(parameter WIDTH = 8)
11    (input clk, sign, start,
12     input [WIDTH-1:0] dividend,
13     input [WIDTH-1:0] divisor,
14     output reg [WIDTH-1:0] quotient,
15     output [WIDTH-1:0] remainder,
16     output ready);
17
18     reg [WIDTH-1:0]  quotient_temp;
19     reg [WIDTH*2-1:0] dividend_copy, divisor_copy, diff;
20     reg negative_output;
21
22     assign remainder = (!negative_output) ?
23               dividend_copy[WIDTH-1:0] : ~dividend_copy[WIDTH-1:0] + 1'
                    b1;
24
25     reg [5:0] bit;
26     reg del_ready = 1;
```

```
    assign ready = (!bit) & ~del_ready;

    wire [WIDTH-2:0] zeros = 0;
    initial bit = 0;
    initial negative_output = 0;
    always @( posedge clk ) begin
       del_ready <= !bit;
       if( start ) begin

          bit = WIDTH;
          quotient = 0;
          quotient_temp = 0;
          dividend_copy = (!sign || !dividend[WIDTH-1]) ?
                          {1'b0,zeros,dividend} :
                          {1'b0,zeros,~dividend + 1'b1};
          divisor_copy = (!sign || !divisor[WIDTH-1]) ?
                          {1'b0,divisor,zeros} :
                          {1'b0,~divisor + 1'b1,zeros};

          negative_output = sign &&
                            ((divisor[WIDTH-1] && !dividend[WIDTH-1])
                             ||(!divisor[WIDTH-1] && dividend[WIDTH-1]))
                                ;
        end
       else if ( bit > 0 ) begin
          diff = dividend_copy - divisor_copy;
          quotient_temp = quotient_temp << 1;
          if( !diff[WIDTH*2-1] ) begin
             dividend_copy = diff;
             quotient_temp[0] = 1'd1;
          end
          quotient = (!negative_output) ?
                     quotient_temp :
                     ~quotient_temp + 1'b1;
          divisor_copy = divisor_copy >> 1;
          bit = bit - 1'b1;
       end
    end
endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/15/2018 02:28:24 PM
// Design Name:
// Module Name: frame_transfer
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
```

```
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module frame_transfer(
        input clk,
        input[18:0] read_addr,
        input[11:0] read_data,

        // Properly timed outputs
        output[14:0] write_addr_out,
        output[11:0] write_data_out,

        output we_out,

        input start,
        output done
    );

parameter TARGET = 144;
parameter MAX_CYCLES = TARGET ** 2 + 2;

// States
parameter IDLE = 0;
parameter TRANSFERRING = 1;

reg state = 0;

reg[15:0] cycle_count = 0;

// Shift registers for data, addresses

// Read data shift register (only 2 needed)
reg[11:0] read_data_sr[1:0];

// Write address SR (only 1 needed)
// I guess not really a shift register but whatever
//reg[14:0] write_addr_sr;

always @(posedge clk) begin
    case(state)
        IDLE: begin
            if(start) begin
                state <= TRANSFERRING;
                read_data_sr[1] <= read_data_sr[0];
                read_data_sr[0] <= read_data;
                cycle_count <= 0;
            end
        end
        TRANSFERRING: begin
            cycle_count <= cycle_count + 1;

            // Shift data
            read_data_sr[1] <= read_data_sr[0];
```

```
            read_data_sr[0] <= read_data;
//            write_addr_sr <= write_addr;

            if(cycle_count == MAX_CYCLES - 1) begin
                state <= IDLE;
            end
        end
    endcase

end

//assign write_addr_out = write_addr_sr;
assign write_addr_out = cycle_count;
assign write_data_out = read_data_sr[1];
assign we_out = (state == TRANSFERRING);
assign done = !start && (cycle_count == MAX_CYCLES);
endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/27/2018 05:20:23 PM
// Design Name:
// Module Name: max_score
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module max_score(
        input[8:0] none_score,
        input[8:0] one_score,
        input[8:0] two_score,
        input[8:0] three_score,
        input[8:0] four_score,
        input[8:0] five_score,
        input[8:0] six_score,
        input[8:0] seven_score,
        input[8:0] eight_score,
        input[8:0] nine_score,
        output[3:0] max_score_out
    );

```

```
reg[3:0] max_score = 0;
assign max_score_out = max_score;
always @* begin
    // Warning: Stupidity ahead
    if(none_score - 24 >= one_score &&
       none_score - 24 >= two_score &&
       none_score - 24 >= three_score &&
       none_score - 24 >= four_score &&
       none_score - 24 >= five_score &&
       none_score - 24 >= six_score &&
       none_score - 24 >= seven_score &&
       none_score - 24 >= eight_score &&
       none_score - 24 >= nine_score) begin

       max_score = 0;
    end else
    if(one_score >= one_score &&
       one_score >= two_score &&
       one_score >= three_score &&
       one_score >= four_score &&
       one_score >= five_score &&
       one_score >= six_score &&
       one_score >= seven_score &&
       one_score >= eight_score &&
       one_score >= nine_score) begin

       max_score = 1;
    end else
    if(two_score >= one_score &&
       two_score >= two_score &&
       two_score >= three_score &&
       two_score >= four_score &&
       two_score >= five_score &&
       two_score >= six_score &&
       two_score >= seven_score &&
       two_score >= eight_score &&
       two_score >= nine_score) begin

       max_score = 2;
    end else
    if(three_score >= one_score &&
       three_score >= two_score &&
       three_score >= three_score &&
       three_score >= four_score &&
       three_score >= five_score &&
       three_score >= six_score &&
       three_score >= seven_score &&
       three_score >= eight_score &&
       three_score >= nine_score) begin

       max_score = 3;
    end else
    if(four_score - 5 >= one_score &&
       four_score - 5 >= two_score &&
       four_score - 5 >= three_score &&
       four_score - 5 >= four_score &&
       four_score - 5 >= five_score &&
       four_score - 5 >= six_score &&
```

```
95          four_score - 5 >= seven_score &&
96          four_score - 5 >= eight_score &&
97          four_score - 5 >= nine_score) begin
98
99          max_score = 4;
100     end else
101     if(five_score - 2 >= one_score &&
102         five_score - 2 >= two_score &&
103         five_score - 2 >= three_score &&
104         five_score - 2 >= four_score &&
105         five_score - 2 >= five_score &&
106         five_score - 2 >= six_score &&
107         five_score - 2 >= seven_score &&
108         five_score - 2 >= eight_score &&
109         five_score - 2 >= nine_score) begin
110
111         max_score = 5;
112     end else
113     if(six_score >= one_score &&
114         six_score >= two_score &&
115         six_score >= three_score &&
116         six_score >= four_score &&
117         six_score >= five_score &&
118         six_score >= six_score &&
119         six_score >= seven_score &&
120         six_score >= eight_score &&
121         six_score >= nine_score) begin
122
123         max_score = 6;
124     end else
125     if(seven_score >= one_score &&
126         seven_score >= two_score &&
127         seven_score >= three_score &&
128         seven_score >= four_score &&
129         seven_score >= five_score &&
130         seven_score >= six_score &&
131         seven_score >= seven_score &&
132         seven_score >= eight_score &&
133         seven_score >= nine_score) begin
134
135         max_score = 7;
136     end else
137     if(eight_score >= one_score &&
138         eight_score >= two_score &&
139         eight_score >= three_score &&
140         eight_score >= four_score &&
141         eight_score >= five_score &&
142         eight_score >= six_score &&
143         eight_score >= seven_score &&
144         eight_score >= eight_score &&
145         eight_score >= nine_score) begin
146
147         max_score = 8;
148     end else
149     if(nine_score - 20 >= one_score &&
150         nine_score - 20 >= two_score &&
151         nine_score - 20 >= three_score &&
152         nine_score - 20 >= four_score &&
```

```
        nine_score - 20 >= five_score &&
        nine_score - 20 >= six_score &&
        nine_score - 20 >= seven_score &&
        nine_score - 20 >= eight_score &&
        nine_score - 20 >= nine_score) begin

        max_score = 9;
    end
end
endmodule
```

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/06/2015 02:09:28 PM
// Design Name:
// Module Name: OV7670_config
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module OV7670_config
#(
    parameter CLK_FREQ = 25000000
)
(
    input wire clk,
    input wire SCCB_interface_ready,
    input wire [15:0] rom_data,
    input wire start,
    output reg [7:0] rom_addr,
    output reg done,
    output reg [7:0] SCCB_interface_addr,
    output reg [7:0] SCCB_interface_data,
    output reg SCCB_interface_start
    );

    initial begin
        rom_addr = 0;
        done = 0;
        SCCB_interface_addr = 0;
        SCCB_interface_data = 0;
```

```
44          SCCB_interface_start = 0;
45      end
46
47      localparam FSM_IDLE = 0;
48      localparam FSM_SEND_CMD = 1;
49      localparam FSM_DONE = 2;
50      localparam FSM_TIMER = 3;
51
52      reg [2:0] FSM_state = FSM_IDLE;
53      reg [2:0] FSM_return_state;
54      reg [31:0] timer = 0;
55
56      always@(posedge clk) begin
57
58          case(FSM_state)
59
60              FSM_IDLE: begin
61                  FSM_state <= start ? FSM_SEND_CMD : FSM_IDLE;
62                  rom_addr <= 0;
63                  done <= start ? 0 : done;
64              end
65
66              FSM_SEND_CMD: begin
67                  case(rom_data)
68                      16'hFFFF: begin //end of ROM
69                          FSM_state <= FSM_DONE;
70                      end
71
72                      16'hFFF0: begin //delay state
73                          timer <= (CLK_FREQ/100); //10 ms delay
74                          FSM_state <= FSM_TIMER;
75                          FSM_return_state <= FSM_SEND_CMD;
76                          rom_addr <= rom_addr + 1;
77                      end
78
79                      default: begin //normal rom commands
80                          if (SCCB_interface_ready) begin
81                              FSM_state <= FSM_TIMER;
82                              FSM_return_state <= FSM_SEND_CMD;
83                              timer <= 0; //one cycle delay gives ready
                                    chance to deassert
84                              rom_addr <= rom_addr + 1;
85                              SCCB_interface_addr <= rom_data[15:8];
86                              SCCB_interface_data <= rom_data[7:0];
87                              SCCB_interface_start <= 1;
88                          end
89                      end
90                  endcase
91              end
92
93              FSM_DONE: begin //signal done
94                  FSM_state <= FSM_IDLE;
95                  done <= 1;
96              end
97
98
99              FSM_TIMER: begin //count down and jump to next state
100                 FSM_state <= (timer == 0) ? FSM_return_state :
```

```
                    FSM_TIMER;
                timer <= (timer==0) ? 0 : timer - 1;
                SCCB_interface_start <= 0;
            end
        endcase
    end
endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 11/13/2018 09:04:55 PM
// Design Name:
// Module Name: reset
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module pwr_reset(
        input clk,
        input reset_input,
        output reset
    );

reg[23:0] sr = 24'hFFFF;

always @(posedge clk) begin
    sr <= {sr[22:0], 1'b0};
end

assign reset = reset_input || (sr[15] == 0);
endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 12/06/2018 04:31:30 PM
// Design Name:
```

```
// Module Name: rise
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


module rise(
        input clk,
        input in,
        output out
    );

reg last = 0;

always @(posedge clk) begin
    last <= in;
end

assign out = (in && ~last);

endmodule
```

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 11/06/2015 11:48:34 AM
// Design Name:
// Module Name: SCCB_interface
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
    //////////////////////////////////////////////////////////////////////////////////


```

```
22
23  module SCCB_interface
24  #(
25      parameter CLK_FREQ = 25000000,
26      parameter SCCB_FREQ = 100000
27  )
28  (
29      input wire clk,
30      input wire start,
31      input wire [7:0] address,
32      input wire [7:0] data,
33      output reg ready,
34      output reg SIOC_oe,
35      output reg SIOD_oe
36      );
37
38      localparam CAMERA_ADDR = 8'h42;
39      localparam FSM_IDLE = 0;
40      localparam FSM_START_SIGNAL = 1;
41      localparam FSM_LOAD_BYTE = 2;
42      localparam FSM_TX_BYTE_1 = 3;
43      localparam FSM_TX_BYTE_2 = 4;
44      localparam FSM_TX_BYTE_3 = 5;
45      localparam FSM_TX_BYTE_4 = 6;
46      localparam FSM_END_SIGNAL_1 = 7;
47      localparam FSM_END_SIGNAL_2 = 8;
48      localparam FSM_END_SIGNAL_3 = 9;
49      localparam FSM_END_SIGNAL_4 = 10;
50      localparam FSM_DONE = 11;
51      localparam FSM_TIMER = 12;
52
53
54      initial begin
55          SIOC_oe = 0;
56          SIOD_oe = 0;
57          ready = 1;
58      end
59
60      reg [3:0] FSM_state = 0;
61      reg [3:0] FSM_return_state = 0;
62      reg [31:0] timer = 0;
63      reg [7:0] latched_address;
64      reg [7:0] latched_data;
65      reg [1:0] byte_counter = 0;
66      reg [7:0] tx_byte = 0;
67      reg [3:0] byte_index = 0;
68
69
70      always@(posedge clk) begin
71
72          case(FSM_state)
73
74              FSM_IDLE: begin
75                  byte_index <= 0;
76                  byte_counter <= 0;
77                  if (start) begin
78                      FSM_state <= FSM_START_SIGNAL;
79                      latched_address <= address;
```

```
80                  latched_data <= data;
81                  ready <= 0;
82              end
83              else begin
84                  ready <= 1;
85              end
86          end
87
88          FSM_START_SIGNAL: begin //comunication interface start
                signal, bring SIOD low
89              FSM_state <= FSM_TIMER;
90              FSM_return_state <= FSM_LOAD_BYTE;
91              timer <= (CLK_FREQ/(4*SCCB_FREQ));
92              SIOC_oe <= 0;
93              SIOD_oe <= 1;
94          end
95
96          FSM_LOAD_BYTE: begin //load next byte to be transmitted
97              FSM_state <= (byte_counter == 3) ? FSM_END_SIGNAL_1 :
                    FSM_TX_BYTE_1;
98              byte_counter <= byte_counter + 1;
99              byte_index <= 0; //clear byte index
100             case(byte_counter)
101                 0: tx_byte <= CAMERA_ADDR;
102                 1: tx_byte <= latched_address;
103                 2: tx_byte <= latched_data;
104                 default: tx_byte <= latched_data;
105             endcase
106         end
107
108         FSM_TX_BYTE_1: begin //bring SIOC low and and delay for
                next state
109             FSM_state <= FSM_TIMER;
110             FSM_return_state <= FSM_TX_BYTE_2;
111             timer <= (CLK_FREQ/(4*SCCB_FREQ));
112             SIOC_oe <= 1;
113         end
114
115         FSM_TX_BYTE_2: begin //assign output data,
116             FSM_state <= FSM_TIMER;
117             FSM_return_state <= FSM_TX_BYTE_3;
118             timer <= (CLK_FREQ/(4*SCCB_FREQ)); //delay for SIOD to
                    stabilize
119             SIOD_oe <= (byte_index == 8) ? 0 : ~tx_byte[7]; //allow
                    for 9 cycle ack, output enable signal is inverting
120         end
121
122         FSM_TX_BYTE_3: begin // bring SIOC high
123             FSM_state <= FSM_TIMER;
124             FSM_return_state <= FSM_TX_BYTE_4;
125             timer <= (CLK_FREQ/(2*SCCB_FREQ));
126             SIOC_oe <= 0; //output enable is an inverting pulldown
127         end
128
129         FSM_TX_BYTE_4: begin //check for end of byte, incriment
                counter
130             FSM_state <= (byte_index == 8) ? FSM_LOAD_BYTE :
                    FSM_TX_BYTE_1;
```

```
131                 tx_byte <= tx_byte<<1; //shift in next data bit
132                 byte_index <= byte_index + 1;
133             end
134
135             FSM_END_SIGNAL_1: begin //state is entered with SIOC high,
                    SIOD high. Start by bringing SIOC low
136                 FSM_state <= FSM_TIMER;
137                 FSM_return_state <= FSM_END_SIGNAL_2;
138                 timer <= (CLK_FREQ/(4*SCCB_FREQ));
139                 SIOC_oe <= 1;
140             end
141
142             FSM_END_SIGNAL_2: begin // while SIOC is low, bring SIOD
                    low
143                 FSM_state <= FSM_TIMER;
144                 FSM_return_state <= FSM_END_SIGNAL_3;
145                 timer <= (CLK_FREQ/(4*SCCB_FREQ));
146                 SIOD_oe <= 1;
147             end
148
149             FSM_END_SIGNAL_3: begin // bring SIOC high
150                 FSM_state <= FSM_TIMER;
151                 FSM_return_state <= FSM_END_SIGNAL_4;
152                 timer <= (CLK_FREQ/(4*SCCB_FREQ));
153                 SIOC_oe <= 0;
154             end
155
156             FSM_END_SIGNAL_4: begin // bring SIOD high when SIOC is
                    high
157                 FSM_state <= FSM_TIMER;
158                 FSM_return_state <= FSM_DONE;
159                 timer <= (CLK_FREQ/(4*SCCB_FREQ));
160                 SIOD_oe <= 0;
161             end
162
163             FSM_DONE: begin //add delay between transactions
164                 FSM_state <= FSM_TIMER;
165                 FSM_return_state <= FSM_IDLE;
166                 timer <= (2*CLK_FREQ/(SCCB_FREQ));
167                 byte_counter <= 0;
168             end
169
170             FSM_TIMER: begin //count down and jump to next state
171                 FSM_state <= (timer == 0) ? FSM_return_state :
                        FSM_TIMER;
172                 timer <= (timer==0) ? 0 : timer - 1;
173             end
174         endcase
175     end
176
177
178 endmodule
```

```
1  //
2  // DEFINITIONS
3  //
4  `define RESET_GRID(GRID, VALUE) \
5  GRID[0][0] = VALUE;\
```

```
GRID[0][1] = VALUE;\
GRID[0][2] = VALUE;\
GRID[0][3] = VALUE;\
GRID[0][4] = VALUE;\
GRID[0][5] = VALUE;\
GRID[0][6] = VALUE;\
GRID[0][7] = VALUE;\
GRID[0][8] = VALUE;\
GRID[1][0] = VALUE;\
GRID[1][1] = VALUE;\
GRID[1][2] = VALUE;\
GRID[1][3] = VALUE;\
GRID[1][4] = VALUE;\
GRID[1][5] = VALUE;\
GRID[1][6] = VALUE;\
GRID[1][7] = VALUE;\
GRID[1][8] = VALUE;\
GRID[2][0] = VALUE;\
GRID[2][1] = VALUE;\
GRID[2][2] = VALUE;\
GRID[2][3] = VALUE;\
GRID[2][4] = VALUE;\
GRID[2][5] = VALUE;\
GRID[2][6] = VALUE;\
GRID[2][7] = VALUE;\
GRID[2][8] = VALUE;\
GRID[3][0] = VALUE;\
GRID[3][1] = VALUE;\
GRID[3][2] = VALUE;\
GRID[3][3] = VALUE;\
GRID[3][4] = VALUE;\
GRID[3][5] = VALUE;\
GRID[3][6] = VALUE;\
GRID[3][7] = VALUE;\
GRID[3][8] = VALUE;\
GRID[4][0] = VALUE;\
GRID[4][1] = VALUE;\
GRID[4][2] = VALUE;\
GRID[4][3] = VALUE;\
GRID[4][4] = VALUE;\
GRID[4][5] = VALUE;\
GRID[4][6] = VALUE;\
GRID[4][7] = VALUE;\
GRID[4][8] = VALUE;\
GRID[5][0] = VALUE;\
GRID[5][1] = VALUE;\
GRID[5][2] = VALUE;\
GRID[5][3] = VALUE;\
GRID[5][4] = VALUE;\
GRID[5][5] = VALUE;\
GRID[5][6] = VALUE;\
GRID[5][7] = VALUE;\
GRID[5][8] = VALUE;\
GRID[6][0] = VALUE;\
GRID[6][1] = VALUE;\
GRID[6][2] = VALUE;\
GRID[6][3] = VALUE;\
GRID[6][4] = VALUE;\
```

```
GRID[6][5] = VALUE;\
GRID[6][6] = VALUE;\
GRID[6][7] = VALUE;\
GRID[6][8] = VALUE;\
GRID[7][0] = VALUE;\
GRID[7][1] = VALUE;\
GRID[7][2] = VALUE;\
GRID[7][3] = VALUE;\
GRID[7][4] = VALUE;\
GRID[7][5] = VALUE;\
GRID[7][6] = VALUE;\
GRID[7][7] = VALUE;\
GRID[7][8] = VALUE;\
GRID[8][0] = VALUE;\
GRID[8][1] = VALUE;\
GRID[8][2] = VALUE;\
GRID[8][3] = VALUE;\
GRID[8][4] = VALUE;\
GRID[8][5] = VALUE;\
GRID[8][6] = VALUE;\
GRID[8][7] = VALUE;\
GRID[8][8] = VALUE

`define OR_GRID(GRID) \
GRID[0][0] |\
GRID[0][1] |\
GRID[0][2] |\
GRID[0][3] |\
GRID[0][4] |\
GRID[0][5] |\
GRID[0][6] |\
GRID[0][7] |\
GRID[0][8] |\
GRID[1][0] |\
GRID[1][1] |\
GRID[1][2] |\
GRID[1][3] |\
GRID[1][4] |\
GRID[1][5] |\
GRID[1][6] |\
GRID[1][7] |\
GRID[1][8] |\
GRID[2][0] |\
GRID[2][1] |\
GRID[2][2] |\
GRID[2][3] |\
GRID[2][4] |\
GRID[2][5] |\
GRID[2][6] |\
GRID[2][7] |\
GRID[2][8] |\
GRID[3][0] |\
GRID[3][1] |\
GRID[3][2] |\
GRID[3][3] |\
GRID[3][4] |\
GRID[3][5] |\
GRID[3][6] |\
```

```
GRID[3][7] |\
GRID[3][8] |\
GRID[4][0] |\
GRID[4][1] |\
GRID[4][2] |\
GRID[4][3] |\
GRID[4][4] |\
GRID[4][5] |\
GRID[4][6] |\
GRID[4][7] |\
GRID[4][8] |\
GRID[5][0] |\
GRID[5][1] |\
GRID[5][2] |\
GRID[5][3] |\
GRID[5][4] |\
GRID[5][5] |\
GRID[5][6] |\
GRID[5][7] |\
GRID[5][8] |\
GRID[6][0] |\
GRID[6][1] |\
GRID[6][2] |\
GRID[6][3] |\
GRID[6][4] |\
GRID[6][5] |\
GRID[6][6] |\
GRID[6][7] |\
GRID[6][8] |\
GRID[7][0] |\
GRID[7][1] |\
GRID[7][2] |\
GRID[7][3] |\
GRID[7][4] |\
GRID[7][5] |\
GRID[7][6] |\
GRID[7][7] |\
GRID[7][8] |\
GRID[8][0] |\
GRID[8][1] |\
GRID[8][2] |\
GRID[8][3] |\
GRID[8][4] |\
GRID[8][5] |\
GRID[8][6] |\
GRID[8][7] |\
GRID[8][8]

`define RESET_PREVS(ARR, VAL) \
ARR[1] <= VAL;\
ARR[2] <= VAL;\
ARR[3] <= VAL;\
ARR[4] <= VAL;\
ARR[5] <= VAL;\
ARR[6] <= VAL;\
ARR[7] <= VAL;\
ARR[8] <= VAL;\
ARR[9] <= VAL;\
```

```
ARR[10] <= VAL;\
ARR[11] <= VAL;\
ARR[12] <= VAL;\
ARR[13] <= VAL;\
ARR[14] <= VAL;\
ARR[15] <= VAL;\
ARR[16] <= VAL;\
ARR[17] <= VAL;\
ARR[18] <= VAL;\
ARR[19] <= VAL;\
ARR[20] <= VAL;\
ARR[21] <= VAL;\
ARR[22] <= VAL;\
ARR[23] <= VAL;\
ARR[24] <= VAL;\
ARR[25] <= VAL;\
ARR[26] <= VAL;\
ARR[27] <= VAL;\
ARR[28] <= VAL;\
ARR[29] <= VAL;\
ARR[30] <= VAL;\
ARR[31] <= VAL;\
ARR[32] <= VAL;\
ARR[33] <= VAL;\
ARR[34] <= VAL;\
ARR[35] <= VAL;\
ARR[36] <= VAL;\
ARR[37] <= VAL;\
ARR[38] <= VAL;\
ARR[39] <= VAL;\
ARR[40] <= VAL;\
ARR[41] <= VAL;\
ARR[42] <= VAL;\
ARR[43] <= VAL;\
ARR[44] <= VAL;\
ARR[45] <= VAL;\
ARR[46] <= VAL;\
ARR[47] <= VAL;\
ARR[48] <= VAL;\
ARR[49] <= VAL;\
ARR[50] <= VAL;\
ARR[51] <= VAL;\
ARR[52] <= VAL;\
ARR[53] <= VAL;\
ARR[54] <= VAL;\
ARR[55] <= VAL;\
ARR[56] <= VAL;\
ARR[57] <= VAL;\
ARR[58] <= VAL;\
ARR[59] <= VAL;\
ARR[60] <= VAL;\
ARR[61] <= VAL;\
ARR[62] <= VAL;\
ARR[63] <= VAL;\
ARR[64] <= VAL;\
ARR[65] <= VAL;\
ARR[66] <= VAL;\
ARR[67] <= VAL;\
```

```
ARR[68] <= VAL;\
ARR[69] <= VAL;\
ARR[70] <= VAL;\
ARR[71] <= VAL;\
ARR[72] <= VAL;\
ARR[73] <= VAL;\
ARR[74] <= VAL;\
ARR[75] <= VAL;\
ARR[76] <= VAL;\
ARR[77] <= VAL;\
ARR[78] <= VAL;\
ARR[79] <= VAL;\
ARR[80] <= VAL;\
ARR[81] <= VAL;\
ARR[82] <= VAL;\
ARR[83] <= VAL;\
ARR[84] <= VAL;\
ARR[85] <= VAL;\
ARR[86] <= VAL;\
ARR[87] <= VAL;\
ARR[88] <= VAL;\
ARR[89] <= VAL;\
ARR[90] <= VAL;\
ARR[91] <= VAL;\
ARR[92] <= VAL;\
ARR[93] <= VAL;\
ARR[94] <= VAL;\
ARR[95] <= VAL;\
ARR[96] <= VAL;\
ARR[97] <= VAL;\
ARR[98] <= VAL;\
ARR[99] <= VAL;\
ARR[100] <= VAL;\
ARR[101] <= VAL;\
ARR[102] <= VAL;\
ARR[103] <= VAL;\
ARR[104] <= VAL;\
ARR[105] <= VAL;\
ARR[106] <= VAL;\
ARR[107] <= VAL;\
ARR[108] <= VAL;\
ARR[109] <= VAL;\
ARR[110] <= VAL;\
ARR[111] <= VAL;\
ARR[112] <= VAL;\
ARR[113] <= VAL;\
ARR[114] <= VAL;\
ARR[115] <= VAL;\
ARR[116] <= VAL;\
ARR[117] <= VAL;\
ARR[118] <= VAL;\
ARR[119] <= VAL;\
ARR[120] <= VAL;\
ARR[121] <= VAL;\
ARR[122] <= VAL;\
ARR[123] <= VAL;\
ARR[124] <= VAL;\
ARR[125] <= VAL;\
```

```
296 ARR[126] <= VAL;\
297 ARR[127] <= VAL
298
299
300 `define RESET_3_BY_9(GRID) \
301 GRID[0][0] <= 0;\
302 GRID[1][0] <= 0;\
303 GRID[2][0] <= 0;\
304 GRID[3][0] <= 0;\
305 GRID[4][0] <= 0;\
306 GRID[5][0] <= 0;\
307 GRID[6][0] <= 0;\
308 GRID[7][0] <= 0;\
309 GRID[8][0] <= 0;\
310 GRID[0][1] <= 0;\
311 GRID[1][1] <= 0;\
312 GRID[2][1] <= 0;\
313 GRID[3][1] <= 0;\
314 GRID[4][1] <= 0;\
315 GRID[5][1] <= 0;\
316 GRID[6][1] <= 0;\
317 GRID[7][1] <= 0;\
318 GRID[8][1] <= 0;\
319 GRID[0][2] <= 0;\
320 GRID[1][2] <= 0;\
321 GRID[2][2] <= 0;\
322 GRID[3][2] <= 0;\
323 GRID[4][2] <= 0;\
324 GRID[5][2] <= 0;\
325 GRID[6][2] <= 0;\
326 GRID[7][2] <= 0;\
327 GRID[8][2] <= 0
328
329 `define SUM_L9_MASK(M) \
330 M[0] + M[1] + M[2] + M[3] + M[4] + M[5] + M[6] + M[7] + M[8]
331
332 `define HIDDEN_GROUP_MASK(MARR) \
333 {(|MARR[8]), (|MARR[7]), (|MARR[6]), (|MARR[5]), (|MARR[4]), (|MARR[3])
      , (|MARR[2]), (|MARR[1]), (|MARR[0])}
334
335 `define NAKED_GROUP_MASK(MARR, N)    \
336 ((((MARR[8]) == N) ? 1'b1 : 1'b0), (((MARR[7]) == N) ? 1'b1 : 1'b0),
      (((MARR[6]) == N) ? 1'b1 : 1'b0),  \
337  (((MARR[5]) == N) ? 1'b1 : 1'b0), (((MARR[4]) == N) ? 1'b1 : 1'b0),
       (((MARR[3]) == N) ? 1'b1 : 1'b0),  \
338  (((MARR[2]) == N) ? 1'b1 : 1'b0), (((MARR[1]) == N) ? 1'b1 : 1'b0),
       (((MARR[0]) == N) ? 1'b1 : 1'b0)}
339
340 `define SET_ARR_TO_SUM(MARR, SARR) \
341 assign MARR[8] = `SUM_L9_MASK(SARR[8]);\
342 assign MARR[7] = `SUM_L9_MASK(SARR[7]);\
343 assign MARR[6] = `SUM_L9_MASK(SARR[6]);\
344 assign MARR[5] = `SUM_L9_MASK(SARR[5]);\
345 assign MARR[4] = `SUM_L9_MASK(SARR[4]);\
346 assign MARR[3] = `SUM_L9_MASK(SARR[3]);\
347 assign MARR[2] = `SUM_L9_MASK(SARR[2]);\
348 assign MARR[1] = `SUM_L9_MASK(SARR[1]);\
349 assign MARR[0] = `SUM_L9_MASK(SARR[0])
```

```
350
351 `define SET_ARR_TO_SUM_2D(MARR, SARR, IND) \
352 assign MARR[8] = `SUM_L9_MASK(SARR[8][IND]);\
353 assign MARR[7] = `SUM_L9_MASK(SARR[7][IND]);\
354 assign MARR[6] = `SUM_L9_MASK(SARR[6][IND]);\
355 assign MARR[5] = `SUM_L9_MASK(SARR[5][IND]);\
356 assign MARR[4] = `SUM_L9_MASK(SARR[4][IND]);\
357 assign MARR[3] = `SUM_L9_MASK(SARR[3][IND]);\
358 assign MARR[2] = `SUM_L9_MASK(SARR[2][IND]);\
359 assign MARR[1] = `SUM_L9_MASK(SARR[1][IND]);\
360 assign MARR[0] = `SUM_L9_MASK(SARR[0][IND])
361
362 `define ASSIGN_ARR_9(ARR1, ARR2, MASK) \
363 if (MASK[0]) ARR1[0] <= ARR2[0];\
364 if (MASK[1]) ARR1[1] <= ARR2[1];\
365 if (MASK[2]) ARR1[2] <= ARR2[2];\
366 if (MASK[3]) ARR1[3] <= ARR2[3];\
367 if (MASK[4]) ARR1[4] <= ARR2[4];\
368 if (MASK[5]) ARR1[5] <= ARR2[5];\
369 if (MASK[6]) ARR1[6] <= ARR2[6];\
370 if (MASK[7]) ARR1[7] <= ARR2[7];\
371 if (MASK[8]) ARR1[8] <= ARR2[8]
372
373 `define ASSIGN_ARR_TO_DIM_2_9(ARR1, ARR2, IND_D1, MASK) \
374 if (MASK[0]) ARR1[0][IND_D1] <= ARR2[0];\
375 if (MASK[1]) ARR1[1][IND_D1] <= ARR2[1];\
376 if (MASK[2]) ARR1[2][IND_D1] <= ARR2[2];\
377 if (MASK[3]) ARR1[3][IND_D1] <= ARR2[3];\
378 if (MASK[4]) ARR1[4][IND_D1] <= ARR2[4];\
379 if (MASK[5]) ARR1[5][IND_D1] <= ARR2[5];\
380 if (MASK[6]) ARR1[6][IND_D1] <= ARR2[6];\
381 if (MASK[7]) ARR1[7][IND_D1] <= ARR2[7];\
382 if (MASK[8]) ARR1[8][IND_D1] <= ARR2[8]
383
384 `define ASSIGN_ARR_9_CONST(ARR1, CONST, MASK) \
385 if (MASK[0]) ARR1[0] <= CONST;\
386 if (MASK[1]) ARR1[1] <= CONST;\
387 if (MASK[2]) ARR1[2] <= CONST;\
388 if (MASK[3]) ARR1[3] <= CONST;\
389 if (MASK[4]) ARR1[4] <= CONST;\
390 if (MASK[5]) ARR1[5] <= CONST;\
391 if (MASK[6]) ARR1[6] <= CONST;\
392 if (MASK[7]) ARR1[7] <= CONST;\
393 if (MASK[8]) ARR1[8] <= CONST
394
395 `define ASSIGN_ARR_TO_DIM_2_9_CONST(ARR1, CONST, IND_D1, MASK) \
396 if (MASK[0]) ARR1[0][IND_D1] <= CONST;\
397 if (MASK[1]) ARR1[1][IND_D1] <= CONST;\
398 if (MASK[2]) ARR1[2][IND_D1] <= CONST;\
399 if (MASK[3]) ARR1[3][IND_D1] <= CONST;\
400 if (MASK[4]) ARR1[4][IND_D1] <= CONST;\
401 if (MASK[5]) ARR1[5][IND_D1] <= CONST;\
402 if (MASK[6]) ARR1[6][IND_D1] <= CONST;\
403 if (MASK[7]) ARR1[7][IND_D1] <= CONST;\
404 if (MASK[8]) ARR1[8][IND_D1] <= CONST
405
406 //
407 // FUNCTIONS
```

```
//
function automatic [8:0] one_hot;
input [3:0] raw_in;
begin
        case(raw_in)
                4'd1    : one_hot = 9'b0_0000_0001;
                4'd2    : one_hot = 9'b0_0000_0010;
                4'd3    : one_hot = 9'b0_0000_0100;
                4'd4    : one_hot = 9'b0_0000_1000;
                4'd5    : one_hot = 9'b0_0001_0000;
                4'd6    : one_hot = 9'b0_0010_0000;
                4'd7    : one_hot = 9'b0_0100_0000;
                4'd8    : one_hot = 9'b0_1000_0000;
                4'd9    : one_hot = 9'b1_0000_0000;
                default : one_hot = 9'b0_0000_0000;
        endcase
end
endfunction

function automatic valid_one_hot;
input [8:0] one_hot_in;
begin
        case(one_hot_in)
                9'b0_0000_0001  : valid_one_hot = 1'b1;
                9'b0_0000_0010  : valid_one_hot = 1'b1;
                9'b0_0000_0100  : valid_one_hot = 1'b1;
                9'b0_0000_1000  : valid_one_hot = 1'b1;
                9'b0_0001_0000  : valid_one_hot = 1'b1;
                9'b0_0010_0000  : valid_one_hot = 1'b1;
                9'b0_0100_0000  : valid_one_hot = 1'b1;
                9'b0_1000_0000  : valid_one_hot = 1'b1;
                9'b1_0000_0000  : valid_one_hot = 1'b1;
                default                 : valid_one_hot = 1'b0;
        endcase
end
endfunction

function automatic [3:0] get_square;
input [3:0] r_count, c_count;
begin
        if (r_count >= 6)
        begin
                if (c_count >= 6)
                begin
                        get_square = 8;
                end
                else if (c_count >= 3)
                begin
                        get_square = 7;
                end
                else
                begin
                        get_square = 6;
                end
        end
        else if (r_count >= 3)
        begin
                if (c_count >= 6)
```

```
                begin
                        get_square = 5;
                end
                else if (c_count >= 3)
                begin
                        get_square = 4;
                end
                else
                begin
                        get_square = 3;
                end
        end
        else
        begin
                if (c_count >= 6)
                begin
                        get_square = 2;
                end
                else if (c_count >= 3)
                begin
                        get_square = 1;
                end
                else
                begin
                        get_square = 0;
                end
        end
end
endfunction

function automatic [8:0] get_exclusive_line_possibilities;
input [8:0] p1, p2, p3;
begin
        get_exclusive_line_possibilities = p1 & (~p2) & (~p3);
end
endfunction

function automatic [8:0] mux9mask;
input [3:0] sel;
input [8:0] i0, i1, i2, i3, i4, i5, i6, i7, i8;
begin
        case(sel)
                0: mux9mask = i0;
                1: mux9mask = i1;
                2: mux9mask = i2;
                3: mux9mask = i3;
                4: mux9mask = i4;
                5: mux9mask = i5;
                6: mux9mask = i6;
                7: mux9mask = i7;
                8: mux9mask = i8;
                default: ;
        endcase
end
endfunction
```

```
module soduku_solver(
                                                clk_in          ,
```

```
                                                    reset_in       ,
                                                    board_in       ,
                                                    board_out      ,
                                                    done_out       ,
                          invalid_out
                                                );
// INCLUDES
//
        `include "common_lib.v"
        `include "soduku_lib.v"
        // `include "soduku_tb_lib.v"

//
// PARAMETERS
//
        localparam   DONE_COUNTDOWN_START = 81;
//
// PORT DECLARATIONS
//
        // Input clock and reset signal
        input        clk_in, reset_in;
        input  [4*(GRID_SIZE)*(GRID_SIZE)-1:0] board_in;
        output [4*(GRID_SIZE)*(GRID_SIZE)-1:0] board_out;
        output       done_out;
    output               invalid_out;

        // 2D array of 4 bit BCD values
        // Contains all the numbers in unsolved board
        // Number = 0 implies contains value
        wire  [3:0] unsolved [0:(GRID_SIZE-1)] [0:(GRID_SIZE-1)];

        // 2D array of 4 bit BCD values
        // Contains all the numbers in solved board
        wire  [3:0] solved [0:(GRID_SIZE-1)] [0:(GRID_SIZE-1)];

        // This monstrosity is because of the use of Verilog instead of
            SystemVerilog
    assign board_out = {solved[8][8],
        solved[8][7],
        solved[8][6],
        solved[8][5],
        solved[8][4],
        solved[8][3],
        solved[8][2],
        solved[8][1],
        solved[8][0],
        solved[7][8],
        solved[7][7],
        solved[7][6],
        solved[7][5],
        solved[7][4],
        solved[7][3],
        solved[7][2],
        solved[7][1],
        solved[7][0],
        solved[6][8],
        solved[6][7],
        solved[6][6],
```

```
        solved[6][5],
        solved[6][4],
        solved[6][3],
        solved[6][2],
        solved[6][1],
        solved[6][0],
        solved[5][8],
        solved[5][7],
        solved[5][6],
        solved[5][5],
        solved[5][4],
        solved[5][3],
        solved[5][2],
        solved[5][1],
        solved[5][0],
        solved[4][8],
        solved[4][7],
        solved[4][6],
        solved[4][5],
        solved[4][4],
        solved[4][3],
        solved[4][2],
        solved[4][1],
        solved[4][0],
        solved[3][8],
        solved[3][7],
        solved[3][6],
        solved[3][5],
        solved[3][4],
        solved[3][3],
        solved[3][2],
        solved[3][1],
        solved[3][0],
        solved[2][8],
        solved[2][7],
        solved[2][6],
        solved[2][5],
        solved[2][4],
        solved[2][3],
        solved[2][2],
        solved[2][1],
        solved[2][0],
        solved[1][8],
        solved[1][7],
        solved[1][6],
        solved[1][5],
        solved[1][4],
        solved[1][3],
        solved[1][2],
        solved[1][1],
        solved[1][0],
        solved[0][8],
        solved[0][7],
        solved[0][6],
        solved[0][5],
        solved[0][4],
        solved[0][3],
        solved[0][2],
```

```
118         solved[0][1],
119         solved[0][0]};
120
121
122     // the depth in the pvr_prevs we need to restore from
123     reg      [15:0]                          guess_number;
124     reg      [(GRID_SIZE*GRID_SIZE-1):0] error_detected;
125
126         assign invalid_out = (|error_detected) & (~|guess_number);
127
128
129 // REG/WIRE DEFINITIONS
130 //
131         // Contains the current progress of the board
132         // Uses one hot encoding
133         reg     [(GRID_SIZE-1):0] one_hot_board_reg   [0:(GRID_SIZE-1)]
               [0:(GRID_SIZE-1)];
134
135         // Contains 1s where the possible values are
136         reg     [(GRID_SIZE-1):0] pvr [0:(GRID_SIZE-1)] [0:(GRID_SIZE-1)
               ];
137
138         // Contains the past 16 pre-guess pvrs
139         reg     [(GRID_SIZE-1):0]   pvr_prevs [0:(GRID_SIZE-1)] [0:(
               GRID_SIZE-1)] [0:(MAX_GUESSES-1)];
140
141         // Masks of the pvr due to the hidden and naked groups
142         reg     [(GRID_SIZE-1):0] gmr [0:(GRID_SIZE-1)] [0:(GRID_SIZE-1)
               ];
143
144         // A grid containing masks for guesses
145         reg     [(GRID_SIZE-1):0] guesses   [0:(MAX_GUESSES-1)];
146         // the row we will guess next
147         reg      [3:0]                      guess_row [0:(MAX_GUESSES-1)];
148         // the column we will guess next
149         reg      [3:0]                      guess_col [0:(MAX_GUESSES-1)];
150
151         // We define the squares, rows, columns as follows:
152         // 0 ..................... 8
153         // ________________________
154         // |       |       |       | 0
155         // |   0   |   1   |   2   | .
156         // |       |       |       | .
157         // |-------|-------|-------|
158         // |       |       |       | .
159         // |   3   |   4   |   5   | .
160         // |       |       |       | .
161         // |-------|-------|-------|
162         // |       |       |       | .
163         // |   6   |   7   |   8   | .
164         // |_______|_______|_______| 8
165
166         // Contains ones where the rows are solved
167         wire    [(GRID_SIZE-1):0] rows_solved;
168         // Contains ones where the columns are solved
169         wire    [(GRID_SIZE-1):0] cols_solved;
170         // Contains ones where the squares are solved
171         wire    [(GRID_SIZE-1):0] squares_solved;
```

```
172
173         // Contains ones where the numbers are contained
174         wire    [(GRID_SIZE-1):0] rows_contains         [0:(GRID_SIZE
                -1)];
175
176         wire    [(GRID_SIZE-1):0] cols_contains         [0:(GRID_SIZE
                -1)];
177
178         wire    [(GRID_SIZE-1):0] squares_contains      [0:(GRID_SIZE
                -1)];
179
180         // candidate line [i][j] is the mask of values which exist on
                the ith square on the jth line in the square.
181         // candidate lines rows
182         reg     [(GRID_SIZE-1):0] candidate_line_rows_reg   [(GRID_SIZE
                -1):0] [2:0];
183         // candidate lines columns
184         reg     [(GRID_SIZE-1):0] candidate_line_cols_reg   [(GRID_SIZE
                -1):0] [2:0];
185         // groups
186         wire    [(GRID_SIZE-1):0] row_groups [(GRID_SIZE-1):0] [(
                GRID_SIZE-1):0];
187         wire    [(GRID_SIZE-1):0] col_groups [(GRID_SIZE-1):0] [(
                GRID_SIZE-1):0];
188         wire    [(GRID_SIZE-1):0] squ_groups [(GRID_SIZE-1):0] [(
                GRID_SIZE-1):0];
189
190         reg     [6:0]                                       done_countdown
                [(GRID_SIZE-1):0] [(GRID_SIZE-1):0];
191         wire    [6:0]
                done_countdown_orred;
192         wire                                                timeout;
193
194
195 //
196 // GENERATORS
197 //
198         // !!DISGUSTING-CODE WARNING!!
199         // GET YOUR SICK BAG READY.
200         generate
201                 genvar row_genvar;
202                 genvar col_genvar;
203
204                 for (row_genvar = 0; row_genvar < (GRID_SIZE);
                        row_genvar = row_genvar + 1)
205                 begin: row_generator
206                         for (col_genvar = 0; col_genvar < (GRID_SIZE);
                                col_genvar = col_genvar + 1)
207                         begin: col_generator
208                                 wire    [(GRID_SIZE-1):0]
                                        squares_contains_single;
209                                 wire    [(GRID_SIZE-1):0]
                                        single_candidate_mask;
210                                 wire    [(GRID_SIZE-1):0]
                                        single_position_mask_temp;
211                                 wire    [(GRID_SIZE-1):0]
                                        single_position_mask;
212                                 // contains 0s where there is space
```

```
213                             wire    [(GRID_SIZE-1):0]
                                    single_pos_row;
214                             wire    [(GRID_SIZE-1):0]
                                    single_pos_col;
215                             wire    [(GRID_SIZE-1):0] single_pos_sq
                                    ;
216
217                             wire    [(GRID_SIZE-1):0]
                                    possibilities_mask;
218                             wire    [(GRID_SIZE-1):0]
                                    candidate_line_r;
219                             wire    [(GRID_SIZE-1):0]
                                    candidate_line_c;
220
221                             wire
                                                guess_this_cell;
222                             wire
                                                full_row;
223                             wire
                                                full_col;
224
225                             assign full_row =       valid_one_hot(
                                    pvr[row_genvar][0]) &
226                                                                     valid_one_ho
                                                                        (
                                                                        pvr
                                                                        [
                                                                        row_genva
                                                                        ]
                                                                        [
                                                                        1
                                                                        ]
                                                                        )

                                                                        &
227                                                                     valid_one_ho
                                                                        (
                                                                        pvr
                                                                        [
                                                                        row_genva
                                                                        ]
                                                                        [
                                                                        2
                                                                        ]
                                                                        )

                                                                        &
228                                                                     valid_one_ho
                                                                        (
                                                                        pvr
                                                                        [
                                                                        row_genva
                                                                        ]
                                                                        [
                                                                        3
                                                                        ]
```

```
                )

                &

229             valid_one_ho
                (
                pvr
                [
                row_genva
                ]
                [
                4
                ]
                )

                &

230             valid_one_ho
                (
                pvr
                [
                row_genva
                ]
                [
                5
                ]
                )

                &

231             valid_one_ho
                (
                pvr
                [
                row_genva
                ]
                [
                6
                ]
                )

                &

232             valid_one_ho
                (
                pvr
                [
                row_genva
                ]
                [
                7
                ]
                )

                &

233             valid_one_ho
                (
```

```
pvr[row_genva][8]);
                    assign full_col =      valid_one_hot(
                        pvr[0][col_genvar]) &
                                                valid_one_ho(pvr[1][col_genva]) &
                                                valid_one_ho(pvr[2][col_genva]) &
                                                valid_one_ho(pvr[3][col_genva]) &
                                                valid_one_ho(pvr[4][
```

```
                col_genva
                ]
                )

                &

239             valid_one_ho
                (
                pvr
                [
                5
                ]
                [
                col_genva
                ]
                )

                &

240             valid_one_ho
                (
                pvr
                [
                6
                ]
                [
                col_genva
                ]
                )

                &

241             valid_one_ho
                (
                pvr
                [
                7
                ]
                [
                col_genva
                ]
                )

                &

242             valid_one_ho
                (
                pvr
                [
                8
                ]
                [
                col_genva
                ]
                )
                ;

243
```

```
244                 assign single_pos_row = {
245                         valid_one_hot({pvr[row_genvar][
                               8][8], pvr[row_genvar][7][8]
                               , pvr[row_genvar][6][8], pvr
                               [row_genvar][5][8], pvr[
                               row_genvar][4][8], pvr[
                               row_genvar][3][8], pvr[
                               row_genvar][2][8], pvr[
                               row_genvar][1][8], pvr[
                               row_genvar][0][8]}),
246                         valid_one_hot({pvr[row_genvar][
                               8][7], pvr[row_genvar][7][7]
                               , pvr[row_genvar][6][7], pvr
                               [row_genvar][5][7], pvr[
                               row_genvar][4][7], pvr[
                               row_genvar][3][7], pvr[
                               row_genvar][2][7], pvr[
                               row_genvar][1][7], pvr[
                               row_genvar][0][7]}),
247                         valid_one_hot({pvr[row_genvar][
                               8][6], pvr[row_genvar][7][6]
                               , pvr[row_genvar][6][6], pvr
                               [row_genvar][5][6], pvr[
                               row_genvar][4][6], pvr[
                               row_genvar][3][6], pvr[
                               row_genvar][2][6], pvr[
                               row_genvar][1][6], pvr[
                               row_genvar][0][6]}),
248                         valid_one_hot({pvr[row_genvar][
                               8][5], pvr[row_genvar][7][5]
                               , pvr[row_genvar][6][5], pvr
                               [row_genvar][5][5], pvr[
                               row_genvar][4][5], pvr[
                               row_genvar][3][5], pvr[
                               row_genvar][2][5], pvr[
                               row_genvar][1][5], pvr[
                               row_genvar][0][5]}),
249                         valid_one_hot({pvr[row_genvar][
                               8][4], pvr[row_genvar][7][4]
                               , pvr[row_genvar][6][4], pvr
                               [row_genvar][5][4], pvr[
                               row_genvar][4][4], pvr[
                               row_genvar][3][4], pvr[
                               row_genvar][2][4], pvr[
                               row_genvar][1][4], pvr[
                               row_genvar][0][4]}),
250                         valid_one_hot({pvr[row_genvar][
                               8][3], pvr[row_genvar][7][3]
                               , pvr[row_genvar][6][3], pvr
                               [row_genvar][5][3], pvr[
                               row_genvar][4][3], pvr[
                               row_genvar][3][3], pvr[
                               row_genvar][2][3], pvr[
                               row_genvar][1][3], pvr[
                               row_genvar][0][3]}),
251                         valid_one_hot({pvr[row_genvar][
                               8][2], pvr[row_genvar][7][2]
                               , pvr[row_genvar][6][2], pvr
```

```
                                    [row_genvar][5][2], pvr[
                                    row_genvar][4][2], pvr[
                                    row_genvar][3][2], pvr[
                                    row_genvar][2][2], pvr[
                                    row_genvar][1][2], pvr[
                                    row_genvar][0][2]}),
252                                 valid_one_hot({pvr[row_genvar][
                                    8][1], pvr[row_genvar][7][1]
                                    , pvr[row_genvar][6][1], pvr
                                    [row_genvar][5][1], pvr[
                                    row_genvar][4][1], pvr[
                                    row_genvar][3][1], pvr[
                                    row_genvar][2][1], pvr[
                                    row_genvar][1][1], pvr[
                                    row_genvar][0][1]}),
253                                 valid_one_hot({pvr[row_genvar][
                                    8][0], pvr[row_genvar][7][0]
                                    , pvr[row_genvar][6][0], pvr
                                    [row_genvar][5][0], pvr[
                                    row_genvar][4][0], pvr[
                                    row_genvar][3][0], pvr[
                                    row_genvar][2][0], pvr[
                                    row_genvar][1][0], pvr[
                                    row_genvar][0][0]})
254                             };
255
256                             assign single_pos_col = {
257                                 valid_one_hot({pvr[8][
                                    col_genvar][8], pvr[7][
                                    col_genvar][8], pvr[6][
                                    col_genvar][8], pvr[5][
                                    col_genvar][8], pvr[4][
                                    col_genvar][8], pvr[3][
                                    col_genvar][8], pvr[2][
                                    col_genvar][8], pvr[1][
                                    col_genvar][8], pvr[0][
                                    col_genvar][8]}),
258                                 valid_one_hot({pvr[8][
                                    col_genvar][7], pvr[7][
                                    col_genvar][7], pvr[6][
                                    col_genvar][7], pvr[5][
                                    col_genvar][7], pvr[4][
                                    col_genvar][7], pvr[3][
                                    col_genvar][7], pvr[2][
                                    col_genvar][7], pvr[1][
                                    col_genvar][7], pvr[0][
                                    col_genvar][7]}),
259                                 valid_one_hot({pvr[8][
                                    col_genvar][6], pvr[7][
                                    col_genvar][6], pvr[6][
                                    col_genvar][6], pvr[5][
                                    col_genvar][6], pvr[4][
                                    col_genvar][6], pvr[3][
                                    col_genvar][6], pvr[2][
                                    col_genvar][6], pvr[1][
                                    col_genvar][6], pvr[0][
                                    col_genvar][6]}),
260                                 valid_one_hot({pvr[8][
```

```
                col_genvar][5], pvr[7][
                col_genvar][5], pvr[6][
                col_genvar][5], pvr[5][
                col_genvar][5], pvr[4][
                col_genvar][5], pvr[3][
                col_genvar][5], pvr[2][
                col_genvar][5], pvr[1][
                col_genvar][5], pvr[0][
                col_genvar][5]}),
261         valid_one_hot({pvr[8][
                col_genvar][4], pvr[7][
                col_genvar][4], pvr[6][
                col_genvar][4], pvr[5][
                col_genvar][4], pvr[4][
                col_genvar][4], pvr[3][
                col_genvar][4], pvr[2][
                col_genvar][4], pvr[1][
                col_genvar][4], pvr[0][
                col_genvar][4]}),
262         valid_one_hot({pvr[8][
                col_genvar][3], pvr[7][
                col_genvar][3], pvr[6][
                col_genvar][3], pvr[5][
                col_genvar][3], pvr[4][
                col_genvar][3], pvr[3][
                col_genvar][3], pvr[2][
                col_genvar][3], pvr[1][
                col_genvar][3], pvr[0][
                col_genvar][3]}),
263         valid_one_hot({pvr[8][
                col_genvar][2], pvr[7][
                col_genvar][2], pvr[6][
                col_genvar][2], pvr[5][
                col_genvar][2], pvr[4][
                col_genvar][2], pvr[3][
                col_genvar][2], pvr[2][
                col_genvar][2], pvr[1][
                col_genvar][2], pvr[0][
                col_genvar][2]}),
264         valid_one_hot({pvr[8][
                col_genvar][1], pvr[7][
                col_genvar][1], pvr[6][
                col_genvar][1], pvr[5][
                col_genvar][1], pvr[4][
                col_genvar][1], pvr[3][
                col_genvar][1], pvr[2][
                col_genvar][1], pvr[1][
                col_genvar][1], pvr[0][
                col_genvar][1]}),
265         valid_one_hot({pvr[8][
                col_genvar][0], pvr[7][
                col_genvar][0], pvr[6][
                col_genvar][0], pvr[5][
                col_genvar][0], pvr[4][
                col_genvar][0], pvr[3][
                col_genvar][0], pvr[2][
                col_genvar][0], pvr[1][
                col_genvar][0], pvr[0][
```

```
                                        col_genvar][0]})
266                                 };
267
268                                 if (row_genvar >= 6)
269                                 begin
270                                         if (col_genvar >= 6)
271                                         begin
272                                                 assign
                                                        squares_contains_single
                                                         = squares_contains[
                                                        8];
273                                                 assign candidate_line_c
                                                         =
                                                        candidate_line_cols_reg
                                                        [2][col_genvar-6] &
                                                        candidate_line_cols_reg
                                                        [5][col_genvar-6];
274                                                 assign candidate_line_r
                                                         =
                                                        candidate_line_rows_reg
                                                        [6][row_genvar-6] &
                                                        candidate_line_rows_reg
                                                        [7][row_genvar-6];
275                                                 assign single_pos_sq =
                                                        {
276                                                             valid_one_hot({
                                                                pvr[6][6][8]
                                                                , pvr[6][7][
                                                                8], pvr[6][8
                                                                ][8], pvr[7]
                                                                [6][8], pvr[
                                                                7][7][8],
                                                                pvr[7][8][8]
                                                                , pvr[8][6][
                                                                8], pvr[8][7
                                                                ][8], pvr[8]
                                                                [8][8]}),
277                                                             valid_one_hot({
                                                                pvr[6][6][7]
                                                                , pvr[6][7][
                                                                7], pvr[6][8
                                                                ][7], pvr[7]
                                                                [6][7], pvr[
                                                                7][7][7],
                                                                pvr[7][8][7]
                                                                , pvr[8][6][
                                                                7], pvr[8][7
                                                                ][7], pvr[8]
                                                                [8][7]}),
278                                                             valid_one_hot({
                                                                pvr[6][6][6]
                                                                , pvr[6][7][
                                                                6], pvr[6][8
                                                                ][6], pvr[7]
                                                                [6][6], pvr[
                                                                7][7][6],
                                                                pvr[7][8][6]
                                                                , pvr[8][6][
```

```
                                                    6], pvr[8][7
                                                    ][6], pvr[8]
                                                    [8][6]}),
279                                                 valid_one_hot({
                                                    pvr[6][6][5]
                                                    , pvr[6][7][
                                                    5], pvr[6][8
                                                    ][5], pvr[7]
                                                    [6][5], pvr[
                                                    7][7][5],
                                                    pvr[7][8][5]
                                                    , pvr[8][6][
                                                    5], pvr[8][7
                                                    ][5], pvr[8]
                                                    [8][5]}),
280                                                 valid_one_hot({
                                                    pvr[6][6][4]
                                                    , pvr[6][7][
                                                    4], pvr[6][8
                                                    ][4], pvr[7]
                                                    [6][4], pvr[
                                                    7][7][4],
                                                    pvr[7][8][4]
                                                    , pvr[8][6][
                                                    4], pvr[8][7
                                                    ][4], pvr[8]
                                                    [8][4]}),
281                                                 valid_one_hot({
                                                    pvr[6][6][3]
                                                    , pvr[6][7][
                                                    3], pvr[6][8
                                                    ][3], pvr[7]
                                                    [6][3], pvr[
                                                    7][7][3],
                                                    pvr[7][8][3]
                                                    , pvr[8][6][
                                                    3], pvr[8][7
                                                    ][3], pvr[8]
                                                    [8][3]}),
282                                                 valid_one_hot({
                                                    pvr[6][6][2]
                                                    , pvr[6][7][
                                                    2], pvr[6][8
                                                    ][2], pvr[7]
                                                    [6][2], pvr[
                                                    7][7][2],
                                                    pvr[7][8][2]
                                                    , pvr[8][6][
                                                    2], pvr[8][7
                                                    ][2], pvr[8]
                                                    [8][2]}),
283                                                 valid_one_hot({
                                                    pvr[6][6][1]
                                                    , pvr[6][7][
                                                    1], pvr[6][8
                                                    ][1], pvr[7]
                                                    [6][1], pvr[
                                                    7][7][1],
```

```
                                                pvr[7][8][1]
                                                , pvr[8][6][
                                                1], pvr[8][7
                                                ][1], pvr[8]
                                                [8][1]}),
284                                         valid_one_hot({
                                                pvr[6][6][0]
                                                , pvr[6][7][
                                                0], pvr[6][8
                                                ][0], pvr[7]
                                                [6][0], pvr[
                                                7][7][0],
                                                pvr[7][8][0]
                                                , pvr[8][6][
                                                0], pvr[8][7
                                                ][0], pvr[8]
                                                [8][0]})
285                                 };
286                         end
287                         else if (col_genvar >= 3)
288                         begin
289                                 assign
                                        squares_contains_single
                                         = squares_contains[
                                        7];
290                                 assign candidate_line_c
                                         =
                                        candidate_line_cols_reg
                                        [1][col_genvar-3] &
                                        candidate_line_cols_reg
                                        [4][col_genvar-3];
291                                 assign candidate_line_r
                                         =
                                        candidate_line_rows_reg
                                        [6][row_genvar-6] &
                                        candidate_line_rows_reg
                                        [8][row_genvar-6];
292                                 assign single_pos_sq =
                                        {
293                                             valid_one_hot({
                                                pvr[6][3][8]
                                                , pvr[6][4][
                                                8], pvr[6][5
                                                ][8], pvr[7]
                                                [3][8], pvr[
                                                7][4][8],
                                                pvr[7][5][8]
                                                , pvr[8][3][
                                                8], pvr[8][4
                                                ][8], pvr[8]
                                                [5][8]}),
294                                             valid_one_hot({
                                                pvr[6][3][7]
                                                , pvr[6][4][
                                                7], pvr[6][5
                                                ][7], pvr[7]
                                                [3][7], pvr[
                                                7][4][7],
```

```
                                                    pvr[7][5][7]
                                                    , pvr[8][3][
                                                    7], pvr[8][4
                                                    ][7], pvr[8]
                                                    [5][7]}),
295                                                 valid_one_hot({
                                                    pvr[6][3][6]
                                                    , pvr[6][4][
                                                    6], pvr[6][5
                                                    ][6], pvr[7]
                                                    [3][6], pvr[
                                                    7][4][6],
                                                    pvr[7][5][6]
                                                    , pvr[8][3][
                                                    6], pvr[8][4
                                                    ][6], pvr[8]
                                                    [5][6]}),
296                                                 valid_one_hot({
                                                    pvr[6][3][5]
                                                    , pvr[6][4][
                                                    5], pvr[6][5
                                                    ][5], pvr[7]
                                                    [3][5], pvr[
                                                    7][4][5],
                                                    pvr[7][5][5]
                                                    , pvr[8][3][
                                                    5], pvr[8][4
                                                    ][5], pvr[8]
                                                    [5][5]}),
297                                                 valid_one_hot({
                                                    pvr[6][3][4]
                                                    , pvr[6][4][
                                                    4], pvr[6][5
                                                    ][4], pvr[7]
                                                    [3][4], pvr[
                                                    7][4][4],
                                                    pvr[7][5][4]
                                                    , pvr[8][3][
                                                    4], pvr[8][4
                                                    ][4], pvr[8]
                                                    [5][4]}),
298                                                 valid_one_hot({
                                                    pvr[6][3][3]
                                                    , pvr[6][4][
                                                    3], pvr[6][5
                                                    ][3], pvr[7]
                                                    [3][3], pvr[
                                                    7][4][3],
                                                    pvr[7][5][3]
                                                    , pvr[8][3][
                                                    3], pvr[8][4
                                                    ][3], pvr[8]
                                                    [5][3]}),
299                                                 valid_one_hot({
                                                    pvr[6][3][2]
                                                    , pvr[6][4][
                                                    2], pvr[6][5
                                                    ][2], pvr[7]
```

```
                                                    [3][2], pvr[
                                                    7][4][2],
                                                    pvr[7][5][2]
                                                    , pvr[8][3][
                                                    2], pvr[8][4
                                                    ][2], pvr[8]
                                                    [5][2]}),
300                                             valid_one_hot({
                                                    pvr[6][3][1]
                                                    , pvr[6][4][
                                                    1], pvr[6][5
                                                    ][1], pvr[7]
                                                    [3][1], pvr[
                                                    7][4][1],
                                                    pvr[7][5][1]
                                                    , pvr[8][3][
                                                    1], pvr[8][4
                                                    ][1], pvr[8]
                                                    [5][1]}),
301                                             valid_one_hot({
                                                    pvr[6][3][0]
                                                    , pvr[6][4][
                                                    0], pvr[6][5
                                                    ][0], pvr[7]
                                                    [3][0], pvr[
                                                    7][4][0],
                                                    pvr[7][5][0]
                                                    , pvr[8][3][
                                                    0], pvr[8][4
                                                    ][0], pvr[8]
                                                    [5][0]})
302                                         };
303                                 end
304                                 else
305                                 begin
306                                     assign
                                        squares_contains_single
                                         = squares_contains[
                                        6];
307                                     assign candidate_line_c
                                         =
                                        candidate_line_cols_reg
                                        [0][col_genvar] &
                                        candidate_line_cols_reg
                                        [3][col_genvar];
308                                     assign candidate_line_r
                                         =
                                        candidate_line_rows_reg
                                        [7][row_genvar-6] &
                                        candidate_line_rows_reg
                                        [8][row_genvar-6];
309                                     assign single_pos_sq =
                                        {
310                                         valid_one_hot({
                                                    pvr[6][0][8]
                                                    , pvr[6][1][
                                                    8], pvr[6][2
                                                    ][8], pvr[7]
```

```
                                                    [0][8], pvr[
                                                    7][1][8],
                                                    pvr[7][2][8]
                                                    , pvr[8][0][
                                                    8], pvr[8][1
                                                    ][8], pvr[8]
                                                    [2][8]}),
311                                                 valid_one_hot({
                                                    pvr[6][0][7]
                                                    , pvr[6][1][
                                                    7], pvr[6][2
                                                    ][7], pvr[7]
                                                    [0][7], pvr[
                                                    7][1][7],
                                                    pvr[7][2][7]
                                                    , pvr[8][0][
                                                    7], pvr[8][1
                                                    ][7], pvr[8]
                                                    [2][7]}),
312                                                 valid_one_hot({
                                                    pvr[6][0][6]
                                                    , pvr[6][1][
                                                    6], pvr[6][2
                                                    ][6], pvr[7]
                                                    [0][6], pvr[
                                                    7][1][6],
                                                    pvr[7][2][6]
                                                    , pvr[8][0][
                                                    6], pvr[8][1
                                                    ][6], pvr[8]
                                                    [2][6]}),
313                                                 valid_one_hot({
                                                    pvr[6][0][5]
                                                    , pvr[6][1][
                                                    5], pvr[6][2
                                                    ][5], pvr[7]
                                                    [0][5], pvr[
                                                    7][1][5],
                                                    pvr[7][2][5]
                                                    , pvr[8][0][
                                                    5], pvr[8][1
                                                    ][5], pvr[8]
                                                    [2][5]}),
314                                                 valid_one_hot({
                                                    pvr[6][0][4]
                                                    , pvr[6][1][
                                                    4], pvr[6][2
                                                    ][4], pvr[7]
                                                    [0][4], pvr[
                                                    7][1][4],
                                                    pvr[7][2][4]
                                                    , pvr[8][0][
                                                    4], pvr[8][1
                                                    ][4], pvr[8]
                                                    [2][4]}),
315                                                 valid_one_hot({
                                                    pvr[6][0][3]
                                                    , pvr[6][1][
```

```
                                                    3], pvr[6][2
                                                    ][3], pvr[7]
                                                    [0][3], pvr[
                                                    7][1][3],
                                                    pvr[7][2][3]
                                                    , pvr[8][0][
                                                    3], pvr[8][1
                                                    ][3], pvr[8]
                                                    [2][3]}),
316                                                 valid_one_hot({
                                                    pvr[6][0][2]
                                                    , pvr[6][1][
                                                    2], pvr[6][2
                                                    ][2], pvr[7]
                                                    [0][2], pvr[
                                                    7][1][2],
                                                    pvr[7][2][2]
                                                    , pvr[8][0][
                                                    2], pvr[8][1
                                                    ][2], pvr[8]
                                                    [2][2]}),
317                                                 valid_one_hot({
                                                    pvr[6][0][1]
                                                    , pvr[6][1][
                                                    1], pvr[6][2
                                                    ][1], pvr[7]
                                                    [0][1], pvr[
                                                    7][1][1],
                                                    pvr[7][2][1]
                                                    , pvr[8][0][
                                                    1], pvr[8][1
                                                    ][1], pvr[8]
                                                    [2][1]}),
318                                                 valid_one_hot({
                                                    pvr[6][0][0]
                                                    , pvr[6][1][
                                                    0], pvr[6][2
                                                    ][0], pvr[7]
                                                    [0][0], pvr[
                                                    7][1][0],
                                                    pvr[7][2][0]
                                                    , pvr[8][0][
                                                    0], pvr[8][1
                                                    ][0], pvr[8]
                                                    [2][0]})
319                                             };
320                                     end
321                             end
322                             else if (row_genvar >= 3)
323                             begin
324                                     if (col_genvar >= 6)
325                                     begin
326                                             assign
                                                    squares_contains_single
                                                     = squares_contains[
                                                    5];
327                                             assign candidate_line_c
                                                     =
```

```
                                    candidate_line_cols_reg
                                    [2][col_genvar-6] &
                                    candidate_line_cols_reg
                                    [8][col_genvar-6];
328                                 assign candidate_line_r
                                     =
                                    candidate_line_rows_reg
                                    [3][row_genvar-3] &
                                    candidate_line_rows_reg
                                    [4][row_genvar-3];
329                                 assign single_pos_sq =
                                    {
330                                      valid_one_hot({
                                         pvr[3][6][8]
                                         , pvr[3][7][
                                         8], pvr[3][8
                                         ][8], pvr[4]
                                         [6][8], pvr[
                                         4][7][8],
                                         pvr[4][8][8]
                                         , pvr[5][6][
                                         8], pvr[5][7
                                         ][8], pvr[5]
                                         [8][8]}),
331                                      valid_one_hot({
                                         pvr[3][6][7]
                                         , pvr[3][7][
                                         7], pvr[3][8
                                         ][7], pvr[4]
                                         [6][7], pvr[
                                         4][7][7],
                                         pvr[4][8][7]
                                         , pvr[5][6][
                                         7], pvr[5][7
                                         ][7], pvr[5]
                                         [8][7]}),
332                                      valid_one_hot({
                                         pvr[3][6][6]
                                         , pvr[3][7][
                                         6], pvr[3][8
                                         ][6], pvr[4]
                                         [6][6], pvr[
                                         4][7][6],
                                         pvr[4][8][6]
                                         , pvr[5][6][
                                         6], pvr[5][7
                                         ][6], pvr[5]
                                         [8][6]}),
333                                      valid_one_hot({
                                         pvr[3][6][5]
                                         , pvr[3][7][
                                         5], pvr[3][8
                                         ][5], pvr[4]
                                         [6][5], pvr[
                                         4][7][5],
                                         pvr[4][8][5]
                                         , pvr[5][6][
                                         5], pvr[5][7
```

```
                                                                    ][5], pvr[5]
                                                                    [8][5]}),
334                                                                 valid_one_hot({
                                                                    pvr[3][6][4]
                                                                    , pvr[3][7][
                                                                    4], pvr[3][8
                                                                    ][4], pvr[4]
                                                                    [6][4], pvr[
                                                                    4][7][4],
                                                                    pvr[4][8][4]
                                                                    , pvr[5][6][
                                                                    4], pvr[5][7
                                                                    ][4], pvr[5]
                                                                    [8][4]}),
335                                                                 valid_one_hot({
                                                                    pvr[3][6][3]
                                                                    , pvr[3][7][
                                                                    3], pvr[3][8
                                                                    ][3], pvr[4]
                                                                    [6][3], pvr[
                                                                    4][7][3],
                                                                    pvr[4][8][3]
                                                                    , pvr[5][6][
                                                                    3], pvr[5][7
                                                                    ][3], pvr[5]
                                                                    [8][3]}),
336                                                                 valid_one_hot({
                                                                    pvr[3][6][2]
                                                                    , pvr[3][7][
                                                                    2], pvr[3][8
                                                                    ][2], pvr[4]
                                                                    [6][2], pvr[
                                                                    4][7][2],
                                                                    pvr[4][8][2]
                                                                    , pvr[5][6][
                                                                    2], pvr[5][7
                                                                    ][2], pvr[5]
                                                                    [8][2]}),
337                                                                 valid_one_hot({
                                                                    pvr[3][6][1]
                                                                    , pvr[3][7][
                                                                    1], pvr[3][8
                                                                    ][1], pvr[4]
                                                                    [6][1], pvr[
                                                                    4][7][1],
                                                                    pvr[4][8][1]
                                                                    , pvr[5][6][
                                                                    1], pvr[5][7
                                                                    ][1], pvr[5]
                                                                    [8][1]}),
338                                                                 valid_one_hot({
                                                                    pvr[3][6][0]
                                                                    , pvr[3][7][
                                                                    0], pvr[3][8
                                                                    ][0], pvr[4]
                                                                    [6][0], pvr[
                                                                    4][7][0],
                                                                    pvr[4][8][0]
```

```
                                                    , pvr[5][6][
                                                    0], pvr[5][7
                                                    ][0], pvr[5]
                                                    [8][0]})
339                                         };
340                                 end
341                                 else if (col_genvar >= 3)
342                                 begin
343                                         assign
                                                squares_contains_single
                                                 = squares_contains[
                                                4];
344                                         assign candidate_line_c
                                                 =
                                                candidate_line_cols_reg
                                                [1][col_genvar-3] &
                                                candidate_line_cols_reg
                                                [7][col_genvar-3];
345                                         assign candidate_line_r
                                                 =
                                                candidate_line_rows_reg
                                                [3][row_genvar-3] &
                                                candidate_line_rows_reg
                                                [5][row_genvar-3];
346                                         assign single_pos_sq =
                                                {
347                                                     valid_one_hot({
                                                        pvr[3][3][8]
                                                        , pvr[3][4][
                                                        8], pvr[3][5
                                                        ][8], pvr[4]
                                                        [3][8], pvr[
                                                        4][4][8],
                                                        pvr[4][5][8]
                                                        , pvr[5][3][
                                                        8], pvr[5][4
                                                        ][8], pvr[5]
                                                        [5][8]}),
348                                                     valid_one_hot({
                                                        pvr[3][3][7]
                                                        , pvr[3][4][
                                                        7], pvr[3][5
                                                        ][7], pvr[4]
                                                        [3][7], pvr[
                                                        4][4][7],
                                                        pvr[4][5][7]
                                                        , pvr[5][3][
                                                        7], pvr[5][4
                                                        ][7], pvr[5]
                                                        [5][7]}),
349                                                     valid_one_hot({
                                                        pvr[3][3][6]
                                                        , pvr[3][4][
                                                        6], pvr[3][5
                                                        ][6], pvr[4]
                                                        [3][6], pvr[
                                                        4][4][6],
                                                        pvr[4][5][6]
```

```
                                                    , pvr[5][3][
                                                    6], pvr[5][4
                                                    ][6], pvr[5]
                                                    [5][6]}),
350                                                 valid_one_hot({
                                                    pvr[3][3][5]
                                                    , pvr[3][4][
                                                    5], pvr[3][5
                                                    ][5], pvr[4]
                                                    [3][5], pvr[
                                                    4][4][5],
                                                    pvr[4][5][5]
                                                    , pvr[5][3][
                                                    5], pvr[5][4
                                                    ][5], pvr[5]
                                                    [5][5]}),
351                                                 valid_one_hot({
                                                    pvr[3][3][4]
                                                    , pvr[3][4][
                                                    4], pvr[3][5
                                                    ][4], pvr[4]
                                                    [3][4], pvr[
                                                    4][4][4],
                                                    pvr[4][5][4]
                                                    , pvr[5][3][
                                                    4], pvr[5][4
                                                    ][4], pvr[5]
                                                    [5][4]}),
352                                                 valid_one_hot({
                                                    pvr[3][3][3]
                                                    , pvr[3][4][
                                                    3], pvr[3][5
                                                    ][3], pvr[4]
                                                    [3][3], pvr[
                                                    4][4][3],
                                                    pvr[4][5][3]
                                                    , pvr[5][3][
                                                    3], pvr[5][4
                                                    ][3], pvr[5]
                                                    [5][3]}),
353                                                 valid_one_hot({
                                                    pvr[3][3][2]
                                                    , pvr[3][4][
                                                    2], pvr[3][5
                                                    ][2], pvr[4]
                                                    [3][2], pvr[
                                                    4][4][2],
                                                    pvr[4][5][2]
                                                    , pvr[5][3][
                                                    2], pvr[5][4
                                                    ][2], pvr[5]
                                                    [5][2]}),
354                                                 valid_one_hot({
                                                    pvr[3][3][1]
                                                    , pvr[3][4][
                                                    1], pvr[3][5
                                                    ][1], pvr[4]
                                                    [3][1], pvr[
```

```
                                                    4][4][1],
                                                    pvr[4][5][1]
                                                    , pvr[5][3][
                                                    1], pvr[5][4
                                                    ][1], pvr[5]
                                                    [5][1]}),
355                                             valid_one_hot({
                                                    pvr[3][3][0]
                                                    , pvr[3][4][
                                                    0], pvr[3][5
                                                    ][0], pvr[4]
                                                    [3][0], pvr[
                                                    4][4][0],
                                                    pvr[4][5][0]
                                                    , pvr[5][3][
                                                    0], pvr[5][4
                                                    ][0], pvr[5]
                                                    [5][0]})
356                                     };
357                             end
358                             else
359                             begin
360                                     assign
                                            squares_contains_single
                                             = squares_contains[
                                            3];
361                                     assign candidate_line_c
                                             =
                                            candidate_line_cols_reg
                                            [0][col_genvar] &
                                            candidate_line_cols_reg
                                            [6][col_genvar];
362                                     assign candidate_line_r
                                             =
                                            candidate_line_rows_reg
                                            [4][row_genvar-3] &
                                            candidate_line_rows_reg
                                            [5][row_genvar-3];
363                                     assign single_pos_sq =
                                            {
364                                             valid_one_hot({
                                                    pvr[3][0][8]
                                                    , pvr[3][1][
                                                    8], pvr[3][2
                                                    ][8], pvr[4]
                                                    [0][8], pvr[
                                                    4][1][8],
                                                    pvr[4][2][8]
                                                    , pvr[5][0][
                                                    8], pvr[5][1
                                                    ][8], pvr[5]
                                                    [2][8]}),
365                                             valid_one_hot({
                                                    pvr[3][0][7]
                                                    , pvr[3][1][
                                                    7], pvr[3][2
                                                    ][7], pvr[4]
                                                    [0][7], pvr[
```

```
                                                    4][1][7],
                                                    pvr[4][2][7]
                                                    , pvr[5][0][
                                                    7], pvr[5][1
                                                    ][7], pvr[5]
                                                    [2][7]}),
366                                                 valid_one_hot({
                                                    pvr[3][0][6]
                                                    , pvr[3][1][
                                                    6], pvr[3][2
                                                    ][6], pvr[4]
                                                    [0][6], pvr[
                                                    4][1][6],
                                                    pvr[4][2][6]
                                                    , pvr[5][0][
                                                    6], pvr[5][1
                                                    ][6], pvr[5]
                                                    [2][6]}),
367                                                 valid_one_hot({
                                                    pvr[3][0][5]
                                                    , pvr[3][1][
                                                    5], pvr[3][2
                                                    ][5], pvr[4]
                                                    [0][5], pvr[
                                                    4][1][5],
                                                    pvr[4][2][5]
                                                    , pvr[5][0][
                                                    5], pvr[5][1
                                                    ][5], pvr[5]
                                                    [2][5]}),
368                                                 valid_one_hot({
                                                    pvr[3][0][4]
                                                    , pvr[3][1][
                                                    4], pvr[3][2
                                                    ][4], pvr[4]
                                                    [0][4], pvr[
                                                    4][1][4],
                                                    pvr[4][2][4]
                                                    , pvr[5][0][
                                                    4], pvr[5][1
                                                    ][4], pvr[5]
                                                    [2][4]}),
369                                                 valid_one_hot({
                                                    pvr[3][0][3]
                                                    , pvr[3][1][
                                                    3], pvr[3][2
                                                    ][3], pvr[4]
                                                    [0][3], pvr[
                                                    4][1][3],
                                                    pvr[4][2][3]
                                                    , pvr[5][0][
                                                    3], pvr[5][1
                                                    ][3], pvr[5]
                                                    [2][3]}),
370                                                 valid_one_hot({
                                                    pvr[3][0][2]
                                                    , pvr[3][1][
                                                    2], pvr[3][2
```

```
                                                                    ][2], pvr[4]
                                                                    [0][2], pvr[
                                                                    4][1][2],
                                                                    pvr[4][2][2]
                                                                    , pvr[5][0][
                                                                    2], pvr[5][1
                                                                    ][2], pvr[5]
                                                                    [2][2]}),
371                                                             valid_one_hot({
                                                                    pvr[3][0][1]
                                                                    , pvr[3][1][
                                                                    1], pvr[3][2
                                                                    ][1], pvr[4]
                                                                    [0][1], pvr[
                                                                    4][1][1],
                                                                    pvr[4][2][1]
                                                                    , pvr[5][0][
                                                                    1], pvr[5][1
                                                                    ][1], pvr[5]
                                                                    [2][1]}),
372                                                             valid_one_hot({
                                                                    pvr[3][0][0]
                                                                    , pvr[3][1][
                                                                    0], pvr[3][2
                                                                    ][0], pvr[4]
                                                                    [0][0], pvr[
                                                                    4][1][0],
                                                                    pvr[4][2][0]
                                                                    , pvr[5][0][
                                                                    0], pvr[5][1
                                                                    ][0], pvr[5]
                                                                    [2][0]})
373                                                     };
374                                             end
375                                     end
376                                     else
377                                     begin
378                                             if (col_genvar >= 6)
379                                             begin
380                                                     assign
                                                            squares_contains_single
                                                             = squares_contains[
                                                            2];
381                                                     assign candidate_line_c
                                                             =
                                                            candidate_line_cols_reg
                                                            [5][col_genvar-6] &
                                                            candidate_line_cols_reg
                                                            [8][col_genvar-6];
382                                                     assign candidate_line_r
                                                             =
                                                            candidate_line_rows_reg
                                                            [0][row_genvar] &
                                                            candidate_line_rows_reg
                                                            [1][row_genvar];
383                                                     assign single_pos_sq =
                                                            {
384                                                             valid_one_hot({
```

```
                                                    pvr[0][6][8]
                                                    , pvr[0][7][
                                                    8], pvr[0][8
                                                    ][8], pvr[1]
                                                    [6][8], pvr[
                                                    1][7][8],
                                                    pvr[1][8][8]
                                                    , pvr[2][6][
                                                    8], pvr[2][7
                                                    ][8], pvr[2]
                                                    [8][8]}),
385                                                 valid_one_hot({
                                                    pvr[0][6][7]
                                                    , pvr[0][7][
                                                    7], pvr[0][8
                                                    ][7], pvr[1]
                                                    [6][7], pvr[
                                                    1][7][7],
                                                    pvr[1][8][7]
                                                    , pvr[2][6][
                                                    7], pvr[2][7
                                                    ][7], pvr[2]
                                                    [8][7]}),
386                                                 valid_one_hot({
                                                    pvr[0][6][6]
                                                    , pvr[0][7][
                                                    6], pvr[0][8
                                                    ][6], pvr[1]
                                                    [6][6], pvr[
                                                    1][7][6],
                                                    pvr[1][8][6]
                                                    , pvr[2][6][
                                                    6], pvr[2][7
                                                    ][6], pvr[2]
                                                    [8][6]}),
387                                                 valid_one_hot({
                                                    pvr[0][6][5]
                                                    , pvr[0][7][
                                                    5], pvr[0][8
                                                    ][5], pvr[1]
                                                    [6][5], pvr[
                                                    1][7][5],
                                                    pvr[1][8][5]
                                                    , pvr[2][6][
                                                    5], pvr[2][7
                                                    ][5], pvr[2]
                                                    [8][5]}),
388                                                 valid_one_hot({
                                                    pvr[0][6][4]
                                                    , pvr[0][7][
                                                    4], pvr[0][8
                                                    ][4], pvr[1]
                                                    [6][4], pvr[
                                                    1][7][4],
                                                    pvr[1][8][4]
                                                    , pvr[2][6][
                                                    4], pvr[2][7
                                                    ][4], pvr[2]
```

```
                                                    [8][4]}),
389                                                 valid_one_hot({
                                                    pvr[0][6][3]
                                                    , pvr[0][7][
                                                    3], pvr[0][8
                                                    ][3], pvr[1]
                                                    [6][3], pvr[
                                                    1][7][3],
                                                    pvr[1][8][3]
                                                    , pvr[2][6][
                                                    3], pvr[2][7
                                                    ][3], pvr[2]
                                                    [8][3]}),
390                                                 valid_one_hot({
                                                    pvr[0][6][2]
                                                    , pvr[0][7][
                                                    2], pvr[0][8
                                                    ][2], pvr[1]
                                                    [6][2], pvr[
                                                    1][7][2],
                                                    pvr[1][8][2]
                                                    , pvr[2][6][
                                                    2], pvr[2][7
                                                    ][2], pvr[2]
                                                    [8][2]}),
391                                                 valid_one_hot({
                                                    pvr[0][6][1]
                                                    , pvr[0][7][
                                                    1], pvr[0][8
                                                    ][1], pvr[1]
                                                    [6][1], pvr[
                                                    1][7][1],
                                                    pvr[1][8][1]
                                                    , pvr[2][6][
                                                    1], pvr[2][7
                                                    ][1], pvr[2]
                                                    [8][1]}),
392                                                 valid_one_hot({
                                                    pvr[0][6][0]
                                                    , pvr[0][7][
                                                    0], pvr[0][8
                                                    ][0], pvr[1]
                                                    [6][0], pvr[
                                                    1][7][0],
                                                    pvr[1][8][0]
                                                    , pvr[2][6][
                                                    0], pvr[2][7
                                                    ][0], pvr[2]
                                                    [8][0]})
393                                         };
394                                 end
395                                 else if (col_genvar >= 3)
396                                 begin
397                                         assign
                                                squares_contains_single
                                                 = squares_contains[
                                                1];
398                                         assign candidate_line_c
```

```
                                    =
                                   candidate_line_cols_reg
                                   [4][col_genvar-3] &
                                   candidate_line_cols_reg
                                   [7][col_genvar-3];
399                            assign candidate_line_r
                                    =
                                   candidate_line_rows_reg
                                   [0][row_genvar] &
                                   candidate_line_rows_reg
                                   [2][row_genvar];
400                            assign single_pos_sq =
                                   {
401                                     valid_one_hot({
                                           pvr[0][3][8]
                                           , pvr[0][4][
                                           8], pvr[0][5
                                           ][8], pvr[1]
                                           [3][8], pvr[
                                           1][4][8],
                                           pvr[1][5][8]
                                           , pvr[2][3][
                                           8], pvr[2][4
                                           ][8], pvr[2]
                                           [5][8]}),
402                                     valid_one_hot({
                                           pvr[0][3][7]
                                           , pvr[0][4][
                                           7], pvr[0][5
                                           ][7], pvr[1]
                                           [3][7], pvr[
                                           1][4][7],
                                           pvr[1][5][7]
                                           , pvr[2][3][
                                           7], pvr[2][4
                                           ][7], pvr[2]
                                           [5][7]}),
403                                     valid_one_hot({
                                           pvr[0][3][6]
                                           , pvr[0][4][
                                           6], pvr[0][5
                                           ][6], pvr[1]
                                           [3][6], pvr[
                                           1][4][6],
                                           pvr[1][5][6]
                                           , pvr[2][3][
                                           6], pvr[2][4
                                           ][6], pvr[2]
                                           [5][6]}),
404                                     valid_one_hot({
                                           pvr[0][3][5]
                                           , pvr[0][4][
                                           5], pvr[0][5
                                           ][5], pvr[1]
                                           [3][5], pvr[
                                           1][4][5],
                                           pvr[1][5][5]
                                           , pvr[2][3][
```

```
                                                                    5], pvr[2][4
                                                                    ][5], pvr[2]
                                                                    [5][5]}),
405                                                                 valid_one_hot({
                                                                    pvr[0][3][4]
                                                                    , pvr[0][4][
                                                                    4], pvr[0][5
                                                                    ][4], pvr[1]
                                                                    [3][4], pvr[
                                                                    1][4][4],
                                                                    pvr[1][5][4]
                                                                    , pvr[2][3][
                                                                    4], pvr[2][4
                                                                    ][4], pvr[2]
                                                                    [5][4]}),
406                                                                 valid_one_hot({
                                                                    pvr[0][3][3]
                                                                    , pvr[0][4][
                                                                    3], pvr[0][5
                                                                    ][3], pvr[1]
                                                                    [3][3], pvr[
                                                                    1][4][3],
                                                                    pvr[1][5][3]
                                                                    , pvr[2][3][
                                                                    3], pvr[2][4
                                                                    ][3], pvr[2]
                                                                    [5][3]}),
407                                                                 valid_one_hot({
                                                                    pvr[0][3][2]
                                                                    , pvr[0][4][
                                                                    2], pvr[0][5
                                                                    ][2], pvr[1]
                                                                    [3][2], pvr[
                                                                    1][4][2],
                                                                    pvr[1][5][2]
                                                                    , pvr[2][3][
                                                                    2], pvr[2][4
                                                                    ][2], pvr[2]
                                                                    [5][2]}),
408                                                                 valid_one_hot({
                                                                    pvr[0][3][1]
                                                                    , pvr[0][4][
                                                                    1], pvr[0][5
                                                                    ][1], pvr[1]
                                                                    [3][1], pvr[
                                                                    1][4][1],
                                                                    pvr[1][5][1]
                                                                    , pvr[2][3][
                                                                    1], pvr[2][4
                                                                    ][1], pvr[2]
                                                                    [5][1]}),
409                                                                 valid_one_hot({
                                                                    pvr[0][3][0]
                                                                    , pvr[0][4][
                                                                    0], pvr[0][5
                                                                    ][0], pvr[1]
                                                                    [3][0], pvr[
                                                                    1][4][0],
```

```
                                                pvr[1][5][0]
                                                , pvr[2][3][
                                                0], pvr[2][4
                                                ][0], pvr[2]
                                                [5][0]})
410                                         };
411                                 end
412                                 else
413                                 begin
414                                         assign
                                                squares_contains_single
                                                 = squares_contains[
                                                0];
415                                         assign candidate_line_c
                                                 =
                                                candidate_line_cols_reg
                                                [3][col_genvar] &
                                                candidate_line_cols_reg
                                                [6][col_genvar];
416                                         assign candidate_line_r
                                                 =
                                                candidate_line_rows_reg
                                                [1][row_genvar] &
                                                candidate_line_rows_reg
                                                [2][row_genvar];
417                                         assign single_pos_sq =
                                                {
418                                                     valid_one_hot({
                                                        pvr[0][0][8]
                                                        , pvr[0][1][
                                                        8], pvr[0][2
                                                        ][8], pvr[1]
                                                        [0][8], pvr[
                                                        1][1][8],
                                                        pvr[1][2][8]
                                                        , pvr[2][0][
                                                        8], pvr[2][1
                                                        ][8], pvr[2]
                                                        [2][8]}),
419                                                     valid_one_hot({
                                                        pvr[0][0][7]
                                                        , pvr[0][1][
                                                        7], pvr[0][2
                                                        ][7], pvr[1]
                                                        [0][7], pvr[
                                                        1][1][7],
                                                        pvr[1][2][7]
                                                        , pvr[2][0][
                                                        7], pvr[2][1
                                                        ][7], pvr[2]
                                                        [2][7]}),
420                                                     valid_one_hot({
                                                        pvr[0][0][6]
                                                        , pvr[0][1][
                                                        6], pvr[0][2
                                                        ][6], pvr[1]
                                                        [0][6], pvr[
                                                        1][1][6],
```

```
                                                    pvr[1][2][6]
                                                    , pvr[2][0][
                                                    6], pvr[2][1
                                                    ][6], pvr[2]
                                                    [2][6]}),
421                                                 valid_one_hot({
                                                    pvr[0][0][5]
                                                    , pvr[0][1][
                                                    5], pvr[0][2
                                                    ][5], pvr[1]
                                                    [0][5], pvr[
                                                    1][1][5],
                                                    pvr[1][2][5]
                                                    , pvr[2][0][
                                                    5], pvr[2][1
                                                    ][5], pvr[2]
                                                    [2][5]}),
422                                                 valid_one_hot({
                                                    pvr[0][0][4]
                                                    , pvr[0][1][
                                                    4], pvr[0][2
                                                    ][4], pvr[1]
                                                    [0][4], pvr[
                                                    1][1][4],
                                                    pvr[1][2][4]
                                                    , pvr[2][0][
                                                    4], pvr[2][1
                                                    ][4], pvr[2]
                                                    [2][4]}),
423                                                 valid_one_hot({
                                                    pvr[0][0][3]
                                                    , pvr[0][1][
                                                    3], pvr[0][2
                                                    ][3], pvr[1]
                                                    [0][3], pvr[
                                                    1][1][3],
                                                    pvr[1][2][3]
                                                    , pvr[2][0][
                                                    3], pvr[2][1
                                                    ][3], pvr[2]
                                                    [2][3]}),
424                                                 valid_one_hot({
                                                    pvr[0][0][2]
                                                    , pvr[0][1][
                                                    2], pvr[0][2
                                                    ][2], pvr[1]
                                                    [0][2], pvr[
                                                    1][1][2],
                                                    pvr[1][2][2]
                                                    , pvr[2][0][
                                                    2], pvr[2][1
                                                    ][2], pvr[2]
                                                    [2][2]}),
425                                                 valid_one_hot({
                                                    pvr[0][0][1]
                                                    , pvr[0][1][
                                                    1], pvr[0][2
                                                    ][1], pvr[1]
```

```
                                                    [0][1], pvr[
                                                    1][1][1],
                                                    pvr[1][2][1]
                                                    , pvr[2][0][
                                                    1], pvr[2][1
                                                    ][1], pvr[2]
                                                    [2][1]}),
426                                               valid_one_hot({
                                                    pvr[0][0][0]
                                                    , pvr[0][1][
                                                    0], pvr[0][2
                                                    ][0], pvr[1]
                                                    [0][0], pvr[
                                                    1][1][0],
                                                    pvr[1][2][0]
                                                    , pvr[2][0][
                                                    0], pvr[2][1
                                                    ][0], pvr[2]
                                                    [2][0]})
427                                         };
428                                 end
429                         end
430
431                         assign single_candidate_mask      = (~
                               rows_contains[row_genvar]) & (~
                               cols_contains[col_genvar]) & (~
                               squares_contains_single);
432                         assign single_position_mask_temp  = (
                               single_pos_row | single_pos_col |
                               single_pos_sq) & pvr[row_genvar][
                               col_genvar];
433                         assign single_position_mask       = (|
                               single_position_mask_temp) ?
                               single_position_mask_temp : 9'
                               b111_111_111;
434                         assign guess_this_cell
                                        = (timeout == 1) && (
                               row_genvar == guess_row[guess_number
                               ]) && (col_genvar == guess_col[
                               guess_number]);
435                         assign possibilities_mask         = pvr
                               [row_genvar][col_genvar]
436
437
438
439
```

```
440
441
442
443                             always @(posedge clk_in)
444                             begin
445                                     if (reset_in)
446                                     begin
447                                             done_countdown[
                                                row_genvar][
                                                col_genvar]
                                                           <=
                                                DONE_COUNTDOWN_START
                                                ;
448                                             error_detected[
                                                row_genvar*GRID_SIZE
                                                 + col_genvar] <= 1'
                                                b0;
449                                             one_hot_board_reg[
                                                row_genvar][
                                                col_genvar]
                                                <= one_hot(unsolved[
                                                row_genvar][
                                                col_genvar]);
450                                             // If the board has a
```

```
                                    set value, it can
                                    only be that value
451                                 // if not, the cell
                                    could be anything
452                                 pvr[row_genvar][
                                    col_genvar] <= |(
                                    one_hot(unsolved[
                                    row_genvar][
                                    col_genvar])) ?
453
454
455                                 `RESET_PREVS(pvr_prevs[
                                    row_genvar][
                                    col_genvar], |(
                                    one_hot(unsolved[
                                    row_genvar][
                                    col_genvar])) ?
456
457
458                             end
459                             else if ((|error_detected) & (|
                                guess_number))
460                             begin
461                                     error_detected[
                                        row_genvar*GRID_SIZE
                                         + col_genvar] <= 1'
                                        b0;
```

```
462                     done_countdown[
                            row_genvar][
                            col_genvar] <=
                            DONE_COUNTDOWN_START
                            ;
463                     pvr[row_genvar][
                            col_genvar] <=
                            pvr_prevs[row_genvar
                            ][col_genvar][
                            guess_number-1];
464
465                     if (valid_one_hot(
                            pvr_prevs[row_genvar
                            ][col_genvar][
                            guess_number-1]))
466                     begin
467                             one_hot_board_reg
                                [row_genvar]
                                [col_genvar]
                                 <=
                                pvr_prevs[
                                row_genvar][
                                col_genvar][
                                guess_number
                                -1];
468                     end
469                     else
470                     begin
471                             one_hot_board_reg
                                [row_genvar]
                                [col_genvar]
                                 <= 0;
472                     end
473             end
474             else if ((full_row & ~(
                    rows_solved[row_genvar])) |
                    (full_col & ~(cols_solved[
                    col_genvar])))
475             begin
476                     // $display("ROW: %d,
                            COL: %d", row_genvar
                            , col_genvar);
477                     // $display("ERROR
                            DETECTED HERE");
478                     // $display("FULL_COL:
                            %d", full_col);
479                     // $display("FULL_ROW:
                            %d", full_row);
480
481                     // $display("Rows
                            Solved: ");
482                     // $display("%b %b %b %
                            b %b %b %b %b %b",
                            rows_solved[0],
483                     //
```

```
                                    rows_solved[1],
484                                 //

                                    rows_solved[2],
485                                 //

                                    rows_solved[3],
486                                 //

                                    rows_solved[4],
487                                 //

                                    rows_solved[5],
488                                 //

                                    rows_solved[6],
489                                 //

                                    rows_solved[7],
490                                 //

                                    rows_solved[8] );
491
492                                 // $display("Columns
                                        Solved: ");
493                                 // $display("%b %b %b %
                                        b %b %b %b %b %b",
                                        cols_solved[0],
494                                 //

                                    cols_solved[1],
495                                 //
```

```
                                    cols_solved[2],
496                                 //

                                    cols_solved[3],
497                                 //

                                    cols_solved[4],
498                                 //

                                    cols_solved[5],
499                                 //

                                    cols_solved[6],
500                                 //

                                    cols_solved[7],
501                                 //

                                    cols_solved[8] );
502
503                                 error_detected[
                                        row_genvar*GRID_SIZE
                                         + col_genvar] <= 1;
504                                 // $display("Output: ")
                                        ;
505                                 // `PRINTGRID(solved);
506                             end
507                             else if (one_hot_board_reg[
                                    row_genvar][col_genvar] ==
                                    0)
508                             begin
509                                 if (possibilities_mask
                                        == 0)
510                                 begin
511                                     error_detected[
                                            row_genvar*
                                            GRID_SIZE +
```

```
                                        col_genvar]
                                        <= 1;
512                                 end
513                                 if (valid_one_hot(
                                        possibilities_mask))
514                                 begin
515                                         one_hot_board_reg
                                            [row_genvar]
                                            [col_genvar]
                                             <=
                                            possibilities_mask
                                            ;
516                                 end
517                                 pvr[row_genvar][
                                        col_genvar] <=
                                        possibilities_mask;
518
519                                 if (timeout)
520                                 begin
521                                         done_countdown[
                                            row_genvar][
                                            col_genvar]
                                            <=
                                            DONE_COUNTDOWN_START
                                            ;
522                                 end
523                                 else if ((
                                        possibilities_mask
                                        == pvr[row_genvar][
                                        col_genvar]) && (
                                        done_countdown[
                                        row_genvar][
                                        col_genvar] > 0))
524                                 begin
525                                         done_countdown[
                                            row_genvar][
                                            col_genvar]
                                            <=
                                            done_countdown
                                            [row_genvar]
                                            [col_genvar]
                                             - 1;
526                                 end
527                                 else if (done_countdown
                                        [row_genvar][
                                        col_genvar] > 0)
528                                 begin // something
                                        changed
529                                         done_countdown[
                                            row_genvar][
                                            col_genvar]
                                            <=
                                            DONE_COUNTDOWN_START
                                            ;
530                                 end
531                         end
532                         else if (done_countdown[
                                row_genvar][col_genvar] > 0)
```

```
533                                     begin // already solved
534                                             done_countdown[
                                                    row_genvar][
                                                    col_genvar] <= 0;
535                                     end
536
537                                     if ((timeout == 0) && (
                                            row_genvar == guess_row[
                                            guess_number]) && (
                                            col_genvar == guess_col[
                                            guess_number]))
538                                     begin
539                                             pvr_prevs[row_genvar][
                                                    col_genvar][
                                                    guess_number] <= (~
                                                    guesses[guess_number
                                                    ]) & pvr[row_genvar]
                                                    [col_genvar];
540                                     end
541                                     else if (~timeout)
542                                     begin
543                                             pvr_prevs[row_genvar][
                                                    col_genvar][
                                                    guess_number] <= pvr
                                                    [row_genvar][
                                                    col_genvar];
544                                     end
545                             end
546
547                             assign unsolved[row_genvar][col_genvar]
                                            = board_in[(row_genvar*
                                    GRID_SIZE*4)+(col_genvar*4)+3:
548
549                             assign solved  [row_genvar][col_genvar]
                                            = bcd(one_hot_board_reg[
                                    row_genvar][col_genvar]);
550                     end
551             end
552     endgenerate
553
554     generate
555             genvar solved_genvar;
556             for (solved_genvar = 0; solved_genvar < (GRID_SIZE);
                        solved_genvar = solved_genvar + 1)
557             begin: contains_generator
558                     assign rows_contains[solved_genvar] = (
559
```

560

561

562

563

564

```
565
566
567
568
569                         assign cols_contains[solved_genvar] = (
570
571
```

572

573

574

575

576

577

```
578
579
580                         assign rows_solved[solved_genvar] = &(
                               rows_contains[solved_genvar]);
581                         assign cols_solved[solved_genvar] = &(
                               cols_contains[solved_genvar]);
582                 end
583         endgenerate
584
585         generate
586                 genvar row_square_genvar;
587                 genvar col_square_genvar;
588                 for (row_square_genvar = 0; row_square_genvar < 3;
                       row_square_genvar = row_square_genvar + 1)
589                 begin: squares_generator
590                         for (col_square_genvar = 0; col_square_genvar <
                             3; col_square_genvar = col_square_genvar +
                               1)
591                         begin
592                                 assign squares_contains[(
                                       row_square_genvar * 3) +
                                       col_square_genvar] = (
593
```

596

597

598

599

600

```
601

602

603                         assign squares_solved[(
                               row_square_genvar * 3) +
                               col_square_genvar] = &(
                               squares_contains[(row_square_genvar
```

```
                                        * 3) + col_square_genvar]);
604                         end
605                 end
606         endgenerate
607
608 //
609 // DONE/TIMEOUT
610 //
611
612         assign done_countdown_orred = `OR_GRID(done_countdown);
613         assign timeout = ~|done_countdown_orred;
614         assign done_out = (&{rows_solved, cols_solved, squares_solved})
               ;// | timeout | (|error_detected);
615
616 //
617 // SEQUENTIAL
618 //
619
620         // Naked group and Hidden group alg
621
622         reg  [3:0] row_counter;
623         reg  [3:0] col_counter;
624         wire [3:0] sq_number;
625
626         wire    [3:0]             number_of_ones_in_mask;
627         wire    [3:0]             number_of_zero_masks_rows;
628         wire    [3:0]             number_of_zero_masks_cols;
629         wire    [3:0]             number_of_zero_masks_sqs;
630
631         wire    [(GRID_SIZE-1):0] anded_masks_row [(GRID_SIZE-1):0];
632         wire    [(GRID_SIZE-1):0] anded_masks_col [(GRID_SIZE-1):0];
633         wire    [(GRID_SIZE-1):0] anded_masks_sq  [(GRID_SIZE-1):0];
634
635         wire    [(GRID_SIZE-1):0] pvr_sq                  [(GRID_SIZE
               -1):0];
636
637         wire    [(GRID_SIZE-1):0] hidden_rows_mask;
638         wire    [(GRID_SIZE-1):0] hidden_cols_mask;
639         wire    [(GRID_SIZE-1):0] hidden_sqs_mask;
640         wire    [(GRID_SIZE-1):0] naked_rows_mask_1;
641         wire                [3:0] naked_rows_sum_1 [(GRID_SIZE-1):0];
642         wire    [(GRID_SIZE-1):0] naked_rows_mask_2;
643         wire                [3:0] naked_rows_sum_2 [(GRID_SIZE-1):0];
644         wire    [(GRID_SIZE-1):0] naked_rows_mask;
645         wire    [(GRID_SIZE-1):0] naked_cols_mask_1;
646         wire                [3:0] naked_cols_sum_1 [(GRID_SIZE-1):0];
647         wire    [(GRID_SIZE-1):0] naked_cols_mask_2;
648         wire                [3:0] naked_cols_sum_2 [(GRID_SIZE-1):0];
649         wire    [(GRID_SIZE-1):0] naked_cols_mask;
650         wire    [(GRID_SIZE-1):0] naked_sqs_mask_1;
651         wire                [3:0] naked_sqs_sum_1  [(GRID_SIZE-1):0];
652         wire    [(GRID_SIZE-1):0] naked_sqs_mask_2;
653         wire                [3:0] naked_sqs_sum_2  [(GRID_SIZE-1):0];
654         wire    [(GRID_SIZE-1):0] naked_sqs_mask;
655
656         wire                      hidden_group_detected_row;
657         wire                      hidden_group_detected_col;
658         wire                      hidden_group_detected_sq;
```

```
659         wire                                        hidden_group_detected
                ;
660
661         wire                        naked_group_detected_row;
662         wire                        naked_group_detected_col;
663         wire                        naked_group_detected_sq;
664         wire                                    naked_group_detected;
665
666         generate
667                 genvar a_genvar;
668                 for (a_genvar = 0; a_genvar < (GRID_SIZE); a_genvar =
                        a_genvar + 1)
669                 begin: a_generator
670                         assign anded_masks_row[a_genvar] = pvr[
                            row_counter][col_counter] & pvr[row_counter]
                            [a_genvar];
671                         assign anded_masks_col[a_genvar] = pvr[
                            row_counter][col_counter] & pvr[a_genvar][
                            col_counter];
672                         if (a_genvar < 3)
673                         begin
674                                 assign pvr_sq[a_genvar]
                                                = mux9mask(sq_number,
675
676
677
678
```

```
679
680
681
682
683
684
685                                 assign anded_masks_sq[a_genvar]  = pvr[
```

```
                                        row_counter][col_counter] & pvr_sq[
                                        a_genvar];
686                             end
687                             else if (a_genvar < 6)
688                             begin
689                                     assign pvr_sq[a_genvar]
                                                = mux9mask(sq_number,
690
691
692
693
694
```

```
695
696
697
698
699
700                         assign anded_masks_sq[a_genvar]  = pvr[
                               row_counter][col_counter] & pvr_sq[
                               a_genvar];
701                     end
702                     else
703                     begin
704                         assign pvr_sq[a_genvar]
                                       = mux9mask(sq_number,
```

```
711
712
713
714
715                                 assign anded_masks_sq[a_genvar]  = pvr[
                                        row_counter][col_counter] & pvr_sq[
                                        a_genvar];
716                         end
717                 end
718         endgenerate
719
720         assign sq_number = get_square(row_counter, col_counter);
721
722         assign hidden_rows_mask          = `HIDDEN_GROUP_MASK(
                anded_masks_row);
723         assign hidden_cols_mask          = `HIDDEN_GROUP_MASK(
                anded_masks_col);
724         assign hidden_sqs_mask           = `HIDDEN_GROUP_MASK(
                anded_masks_sq);
```

```
725         // also need to check the ANDED masks
726         `SET_ARR_TO_SUM(naked_rows_sum_1, anded_masks_row);
727         assign naked_rows_mask_1                  = (`NAKED_GROUP_MASK(
               naked_rows_sum_1, number_of_ones_in_mask));
728         `SET_ARR_TO_SUM(naked_rows_sum_2, pvr[row_counter]);
729         assign naked_rows_mask_2                  = (`NAKED_GROUP_MASK(
               naked_rows_sum_2, number_of_ones_in_mask));
730         `SET_ARR_TO_SUM(naked_cols_sum_1, anded_masks_col);
731         assign naked_cols_mask_1                  = (`NAKED_GROUP_MASK(
               naked_cols_sum_1, number_of_ones_in_mask));
732         `SET_ARR_TO_SUM_2D(naked_cols_sum_2, pvr, col_counter);
733         assign naked_cols_mask_2                  = (`NAKED_GROUP_MASK(
               naked_cols_sum_2, number_of_ones_in_mask));
734         `SET_ARR_TO_SUM(naked_sqs_sum_1, anded_masks_sq);
735         assign naked_sqs_mask_1                   = (`NAKED_GROUP_MASK(
               naked_sqs_sum_1, number_of_ones_in_mask));
736         `SET_ARR_TO_SUM(naked_sqs_sum_2, pvr_sq);
737         assign naked_sqs_mask_2                   = (`NAKED_GROUP_MASK(
               naked_sqs_sum_2, number_of_ones_in_mask));
738
739         assign naked_cols_mask           = naked_cols_mask_1 &
               naked_cols_mask_2;
740         assign naked_rows_mask           = naked_rows_mask_1 &
               naked_rows_mask_2;
741         assign naked_sqs_mask            = naked_sqs_mask_1 &
               naked_sqs_mask_2;
742
743         assign number_of_ones_in_mask    = `SUM_L9_MASK(pvr[row_counter
               ][col_counter]);
744         assign number_of_zero_masks_rows = 9-(`SUM_L9_MASK(
               hidden_rows_mask));
745         assign number_of_zero_masks_cols = 9-(`SUM_L9_MASK(
               hidden_cols_mask));
746         assign number_of_zero_masks_sqs  = 9-(`SUM_L9_MASK(
               hidden_sqs_mask));
747
748         assign hidden_group_detected_row = (naked_group_detected == 0)
               && (number_of_ones_in_mask > 1) && (number_of_ones_in_mask <
                9) && (9 == (number_of_ones_in_mask +
               number_of_zero_masks_rows));
749         assign hidden_group_detected_col = (naked_group_detected == 0)
               && (number_of_ones_in_mask > 1) && (number_of_ones_in_mask <
                9) && (9 == (number_of_ones_in_mask +
               number_of_zero_masks_cols));
750         assign hidden_group_detected_sq  = (naked_group_detected == 0)
               && (number_of_ones_in_mask > 1) && (number_of_ones_in_mask <
                9) && (9 == (number_of_ones_in_mask +
               number_of_zero_masks_sqs));
751
752         assign  naked_group_detected  = naked_group_detected_sq  |
               naked_group_detected_col  | naked_group_detected_row;
753         assign  hidden_group_detected = hidden_group_detected_sq |
               hidden_group_detected_col | hidden_group_detected_row;
754
755         assign naked_group_detected_row  = (number_of_ones_in_mask > 1)
                && (number_of_ones_in_mask < 9) && (number_of_ones_in_mask
               == `SUM_L9_MASK(naked_rows_mask));
756         assign naked_group_detected_col  = (number_of_ones_in_mask > 1)
```

```
            && (number_of_ones_in_mask < 9) && (number_of_ones_in_mask
           == `SUM_L9_MASK(naked_cols_mask));
757         assign naked_group_detected_sq    = (number_of_ones_in_mask > 1)
            && (number_of_ones_in_mask < 9) && (number_of_ones_in_mask
           == `SUM_L9_MASK(naked_sqs_mask));
758
759         always @(posedge clk_in)
760         begin
761                 if (reset_in)
762                 begin
763                         `RESET_GRID(gmr, 9'b111_111_111);
764                         row_counter <= 0;
765                         col_counter <= 0;
766                 end
767                 else if (|error_detected)
768                 begin
769                         `RESET_GRID(gmr, 9'b111_111_111);
770                 end
771                 else
772                 begin
773                         if (row_counter == 4'd8)
774                         begin
775                                 if (col_counter == 4'd8)
776                                 begin
777                                         col_counter <= 4'b0;
778                                 end
779                                 else
780                                 begin
781                                         col_counter <= col_counter + 1;
782                                 end
783                                 row_counter <= 4'b0;
784                         end
785                         else
786                         begin
787                                 row_counter <= row_counter + 1;
788                         end
789
790                         if (hidden_group_detected_row)
791                         begin
792                                 `ASSIGN_ARR_9(gmr[row_counter],
                                    anded_masks_row, hidden_rows_mask);
793                         end
794                         if (hidden_group_detected_col)
795                         begin
796                                 `ASSIGN_ARR_TO_DIM_2_9(gmr,
                                    anded_masks_col, col_counter,
                                    hidden_cols_mask);
797                         end
798                         if (hidden_group_detected_sq)
799                         begin
800                                 case(sq_number)
801                                         0:
802                                                 begin
803                                                         gmr[0][0] <=
                                                            hidden_sqs_mask
                                                            [0] ?
                                                            anded_masks_sq
                                                            [0] : gmr[0]
```

```
                                                    [0];
804                                             gmr[0][1] <=
                                                    hidden_sqs_mask
                                                    [1] ?
                                                    anded_masks_sq
                                                    [1] : gmr[0]
                                                    [1];
805                                             gmr[0][2] <=
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[0]
                                                    [2];
806                                             gmr[1][0] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[1]
                                                    [0];
807                                             gmr[1][1] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[1]
                                                    [1];
808                                             gmr[1][2] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[1]
                                                    [2];
809                                             gmr[2][0] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[2]
                                                    [0];
810                                             gmr[2][1] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[2]
                                                    [1];
811                                             gmr[2][2] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[2]
                                                    [2];
812                                         end
813                                 1:
814                                         begin
815                                             gmr[0][3] <=
                                                    hidden_sqs_mask
                                                    [0] ?
                                                    anded_masks_sq
                                                    [0] : gmr[0]
                                                    [3];
```

```
816                                     gmr[0][4] <=
                                            hidden_sqs_mask
                                            [1] ?
                                            anded_masks_sq
                                            [1] : gmr[0]
                                            [4];
817                                     gmr[0][5] <=
                                            hidden_sqs_mask
                                            [2] ?
                                            anded_masks_sq
                                            [2] : gmr[0]
                                            [5];
818                                     gmr[1][3] <=
                                            hidden_sqs_mask
                                            [3] ?
                                            anded_masks_sq
                                            [3] : gmr[1]
                                            [3];
819                                     gmr[1][4] <=
                                            hidden_sqs_mask
                                            [4] ?
                                            anded_masks_sq
                                            [4] : gmr[1]
                                            [4];
820                                     gmr[1][5] <=
                                            hidden_sqs_mask
                                            [5] ?
                                            anded_masks_sq
                                            [5] : gmr[1]
                                            [5];
821                                     gmr[2][3] <=
                                            hidden_sqs_mask
                                            [6] ?
                                            anded_masks_sq
                                            [6] : gmr[2]
                                            [3];
822                                     gmr[2][4] <=
                                            hidden_sqs_mask
                                            [7] ?
                                            anded_masks_sq
                                            [7] : gmr[2]
                                            [4];
823                                     gmr[2][5] <=
                                            hidden_sqs_mask
                                            [8] ?
                                            anded_masks_sq
                                            [8] : gmr[2]
                                            [5];
824                             end
825                     2:
826                             begin
827                                     gmr[0][6] <=
                                            hidden_sqs_mask
                                            [0] ?
                                            anded_masks_sq
                                            [0] : gmr[0]
                                            [6];
828                                     gmr[0][7] <=
```

```
                                    hidden_sqs_mask
                                    [1] ?
                                    anded_masks_sq
                                    [1] : gmr[0]
                                    [7];
829                             gmr[0][8] <=
                                    hidden_sqs_mask
                                    [2] ?
                                    anded_masks_sq
                                    [2] : gmr[0]
                                    [8];
830                             gmr[1][6] <=
                                    hidden_sqs_mask
                                    [3] ?
                                    anded_masks_sq
                                    [3] : gmr[1]
                                    [6];
831                             gmr[1][7] <=
                                    hidden_sqs_mask
                                    [4] ?
                                    anded_masks_sq
                                    [4] : gmr[1]
                                    [7];
832                             gmr[1][8] <=
                                    hidden_sqs_mask
                                    [5] ?
                                    anded_masks_sq
                                    [5] : gmr[1]
                                    [8];
833                             gmr[2][6] <=
                                    hidden_sqs_mask
                                    [6] ?
                                    anded_masks_sq
                                    [6] : gmr[2]
                                    [6];
834                             gmr[2][7] <=
                                    hidden_sqs_mask
                                    [7] ?
                                    anded_masks_sq
                                    [7] : gmr[2]
                                    [7];
835                             gmr[2][8] <=
                                    hidden_sqs_mask
                                    [8] ?
                                    anded_masks_sq
                                    [8] : gmr[2]
                                    [8];
836                         end
837                 3:
838                         begin
839                             gmr[3][0] <=
                                    hidden_sqs_mask
                                    [0] ?
                                    anded_masks_sq
                                    [0] : gmr[3]
                                    [0];
840                             gmr[3][1] <=
                                    hidden_sqs_mask
```

```
                                                    [1] ?
                                                    anded_masks_sq
                                                    [1] : gmr[3]
                                                    [1];
841                                             gmr[3][2] <=
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[3]
                                                    [2];
842                                             gmr[4][0] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[4]
                                                    [0];
843                                             gmr[4][1] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[4]
                                                    [1];
844                                             gmr[4][2] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[4]
                                                    [2];
845                                             gmr[5][0] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[5]
                                                    [0];
846                                             gmr[5][1] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[5]
                                                    [1];
847                                             gmr[5][2] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[5]
                                                    [2];
848                                         end
849                                 4:
850                                         begin
851                                             gmr[3][3] <=
                                                    hidden_sqs_mask
                                                    [0] ?
                                                    anded_masks_sq
                                                    [0] : gmr[3]
                                                    [3];
852                                             gmr[3][4] <=
                                                    hidden_sqs_mask
                                                    [1] ?
```

```
                                                    anded_masks_sq
                                                    [1] : gmr[3]
                                                    [4];
853                                                 gmr[3][5] <=
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[3]
                                                    [5];
854                                                 gmr[4][3] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[4]
                                                    [3];
855                                                 gmr[4][4] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[4]
                                                    [4];
856                                                 gmr[4][5] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[4]
                                                    [5];
857                                                 gmr[5][3] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[5]
                                                    [3];
858                                                 gmr[5][4] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[5]
                                                    [4];
859                                                 gmr[5][5] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[5]
                                                    [5];
860                                         end
861                                 5:
862                                         begin
863                                                 gmr[3][6] <=
                                                    hidden_sqs_mask
                                                    [0] ?
                                                    anded_masks_sq
                                                    [0] : gmr[3]
                                                    [6];
864                                                 gmr[3][7] <=
                                                    hidden_sqs_mask
                                                    [1] ?
                                                    anded_masks_sq
```

```
                                                    [1] : gmr[3]
                                                    [7];
865                                             gmr[3][8] <=
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[3]
                                                    [8];
866                                             gmr[4][6] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[4]
                                                    [6];
867                                             gmr[4][7] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[4]
                                                    [7];
868                                             gmr[4][8] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[4]
                                                    [8];
869                                             gmr[5][6] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[5]
                                                    [6];
870                                             gmr[5][7] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[5]
                                                    [7];
871                                             gmr[5][8] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[5]
                                                    [8];
872                                         end
873                                 6:
874                                         begin
875                                             gmr[6][0] <=
                                                    hidden_sqs_mask
                                                    [0] ?
                                                    anded_masks_sq
                                                    [0] : gmr[6]
                                                    [0];
876                                             gmr[6][1] <=
                                                    hidden_sqs_mask
                                                    [1] ?
                                                    anded_masks_sq
                                                    [1] : gmr[6]
```

```
                                                    [1];
877                                             gmr[6][2] <=
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[6]
                                                    [2];
878                                             gmr[7][0] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[7]
                                                    [0];
879                                             gmr[7][1] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[7]
                                                    [1];
880                                             gmr[7][2] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[7]
                                                    [2];
881                                             gmr[8][0] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[8]
                                                    [0];
882                                             gmr[8][1] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[8]
                                                    [1];
883                                             gmr[8][2] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[8]
                                                    [2];
884                                         end
885                                 7:
886                                         begin
887                                             gmr[6][3] <=
                                                    hidden_sqs_mask
                                                    [0] ?
                                                    anded_masks_sq
                                                    [0] : gmr[6]
                                                    [3];
888                                             gmr[6][4] <=
                                                    hidden_sqs_mask
                                                    [1] ?
                                                    anded_masks_sq
                                                    [1] : gmr[6]
                                                    [4];
```

```
889                                 gmr[6][5] <=
                                        hidden_sqs_mask
                                        [2] ?
                                        anded_masks_sq
                                        [2] : gmr[6]
                                        [5];
890                                 gmr[7][3] <=
                                        hidden_sqs_mask
                                        [3] ?
                                        anded_masks_sq
                                        [3] : gmr[7]
                                        [3];
891                                 gmr[7][4] <=
                                        hidden_sqs_mask
                                        [4] ?
                                        anded_masks_sq
                                        [4] : gmr[7]
                                        [4];
892                                 gmr[7][5] <=
                                        hidden_sqs_mask
                                        [5] ?
                                        anded_masks_sq
                                        [5] : gmr[7]
                                        [5];
893                                 gmr[8][3] <=
                                        hidden_sqs_mask
                                        [6] ?
                                        anded_masks_sq
                                        [6] : gmr[8]
                                        [3];
894                                 gmr[8][4] <=
                                        hidden_sqs_mask
                                        [7] ?
                                        anded_masks_sq
                                        [7] : gmr[8]
                                        [4];
895                                 gmr[8][5] <=
                                        hidden_sqs_mask
                                        [8] ?
                                        anded_masks_sq
                                        [8] : gmr[8]
                                        [5];
896                         end
897                 8:
898                         begin
899                                 gmr[6][6] <=
                                        hidden_sqs_mask
                                        [0] ?
                                        anded_masks_sq
                                        [0] : gmr[6]
                                        [6];
900                                 gmr[6][7] <=
                                        hidden_sqs_mask
                                        [1] ?
                                        anded_masks_sq
                                        [1] : gmr[6]
                                        [7];
901                                 gmr[6][8] <=
```

```
                                                    hidden_sqs_mask
                                                    [2] ?
                                                    anded_masks_sq
                                                    [2] : gmr[6]
                                                    [8];
902                                                 gmr[7][6] <=
                                                    hidden_sqs_mask
                                                    [3] ?
                                                    anded_masks_sq
                                                    [3] : gmr[7]
                                                    [6];
903                                                 gmr[7][7] <=
                                                    hidden_sqs_mask
                                                    [4] ?
                                                    anded_masks_sq
                                                    [4] : gmr[7]
                                                    [7];
904                                                 gmr[7][8] <=
                                                    hidden_sqs_mask
                                                    [5] ?
                                                    anded_masks_sq
                                                    [5] : gmr[7]
                                                    [8];
905                                                 gmr[8][6] <=
                                                    hidden_sqs_mask
                                                    [6] ?
                                                    anded_masks_sq
                                                    [6] : gmr[8]
                                                    [6];
906                                                 gmr[8][7] <=
                                                    hidden_sqs_mask
                                                    [7] ?
                                                    anded_masks_sq
                                                    [7] : gmr[8]
                                                    [7];
907                                                 gmr[8][8] <=
                                                    hidden_sqs_mask
                                                    [8] ?
                                                    anded_masks_sq
                                                    [8] : gmr[8]
                                                    [8];
908                                         end
909                                 default: ;
910                         endcase
911                 end
912                 if (naked_group_detected_row)
913                 begin
914                         `ASSIGN_ARR_9_CONST(gmr[row_counter], ~
                            pvr[row_counter][col_counter], ~
                            naked_rows_mask);
915                 end
916                 if (naked_group_detected_col)
917                 begin
918                         `ASSIGN_ARR_TO_DIM_2_9_CONST(gmr, ~pvr[
                            row_counter][col_counter],
                            col_counter, ~naked_cols_mask);
919                 end
920                 if (naked_group_detected_sq)
```

```
921                     begin
922                             case(sq_number)
923                                     0:
924                                             begin
925                                                     gmr[0][0] <= ~
                                                        naked_sqs_mask
                                                        [0] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[0][0
                                                        ];
926                                                     gmr[0][1] <= ~
                                                        naked_sqs_mask
                                                        [1] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[0][1
                                                        ];
927                                                     gmr[0][2] <= ~
                                                        naked_sqs_mask
                                                        [2] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[0][2
                                                        ];
928                                                     gmr[1][0] <= ~
                                                        naked_sqs_mask
                                                        [3] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[1][0
                                                        ];
929                                                     gmr[1][1] <= ~
                                                        naked_sqs_mask
                                                        [4] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[1][1
                                                        ];
930                                                     gmr[1][2] <= ~
                                                        naked_sqs_mask
                                                        [5] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[1][2
                                                        ];
931                                                     gmr[2][0] <= ~
                                                        naked_sqs_mask
                                                        [6] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
                                                        ] : gmr[2][0
                                                        ];
932                                                     gmr[2][1] <= ~
                                                        naked_sqs_mask
                                                        [7] ? ~pvr[
                                                        row_counter]
                                                        [col_counter
```

```
                                                                ] : gmr[2][1
                                                                ];
933                                                         gmr[2][2] <= ~
                                                                naked_sqs_mask
                                                                [8] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[2][2
                                                                ];
934                                                 end
935                                         1:
936                                                 begin
937                                                         gmr[0][3] <= ~
                                                                naked_sqs_mask
                                                                [0] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[0][3
                                                                ];
938                                                         gmr[0][4] <= ~
                                                                naked_sqs_mask
                                                                [1] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[0][4
                                                                ];
939                                                         gmr[0][5] <= ~
                                                                naked_sqs_mask
                                                                [2] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[0][5
                                                                ];
940                                                         gmr[1][3] <= ~
                                                                naked_sqs_mask
                                                                [3] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[1][3
                                                                ];
941                                                         gmr[1][4] <= ~
                                                                naked_sqs_mask
                                                                [4] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[1][4
                                                                ];
942                                                         gmr[1][5] <= ~
                                                                naked_sqs_mask
                                                                [5] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[1][5
                                                                ];
943                                                         gmr[2][3] <= ~
                                                                naked_sqs_mask
                                                                [6] ? ~pvr[
                                                                row_counter]
```

```
                                                    [col_counter
                                                    ] : gmr[2][3
                                                    ];
944                                                 gmr[2][4] <= ~
                                                    naked_sqs_mask
                                                    [7] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[2][4
                                                    ];
945                                                 gmr[2][5] <= ~
                                                    naked_sqs_mask
                                                    [8] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[2][5
                                                    ];
946                                         end
947                                 2:
948                                         begin
949                                                 gmr[0][6] <= ~
                                                    naked_sqs_mask
                                                    [0] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[0][6
                                                    ];
950                                                 gmr[0][7] <= ~
                                                    naked_sqs_mask
                                                    [1] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[0][7
                                                    ];
951                                                 gmr[0][8] <= ~
                                                    naked_sqs_mask
                                                    [2] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[0][8
                                                    ];
952                                                 gmr[1][6] <= ~
                                                    naked_sqs_mask
                                                    [3] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[1][6
                                                    ];
953                                                 gmr[1][7] <= ~
                                                    naked_sqs_mask
                                                    [4] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[1][7
                                                    ];
954                                                 gmr[1][8] <= ~
                                                    naked_sqs_mask
                                                    [5] ? ~pvr[
```

```
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[1][8
                                                    ];
955                                                 gmr[2][6] <= ~
                                                    naked_sqs_mask
                                                    [6] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[2][6
                                                    ];
956                                                 gmr[2][7] <= ~
                                                    naked_sqs_mask
                                                    [7] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[2][7
                                                    ];
957                                                 gmr[2][8] <= ~
                                                    naked_sqs_mask
                                                    [8] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[2][8
                                                    ];
958                                         end
959                                 3:
960                                         begin
961                                                 gmr[3][0] <= ~
                                                    naked_sqs_mask
                                                    [0] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][0
                                                    ];
962                                                 gmr[3][1] <= ~
                                                    naked_sqs_mask
                                                    [1] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][1
                                                    ];
963                                                 gmr[3][2] <= ~
                                                    naked_sqs_mask
                                                    [2] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][2
                                                    ];
964                                                 gmr[4][0] <= ~
                                                    naked_sqs_mask
                                                    [3] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[4][0
                                                    ];
965                                                 gmr[4][1] <= ~
                                                    naked_sqs_mask
```

```
                                                    [4] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[4][1
                                                    ];
966                                                 gmr[4][2] <= ~
                                                    naked_sqs_mask
                                                    [5] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[4][2
                                                    ];
967                                                 gmr[5][0] <= ~
                                                    naked_sqs_mask
                                                    [6] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[5][0
                                                    ];
968                                                 gmr[5][1] <= ~
                                                    naked_sqs_mask
                                                    [7] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[5][1
                                                    ];
969                                                 gmr[5][2] <= ~
                                                    naked_sqs_mask
                                                    [8] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[5][2
                                                    ];
970                                         end
971                                 4:
972                                         begin
973                                                 gmr[3][3] <= ~
                                                    naked_sqs_mask
                                                    [0] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][3
                                                    ];
974                                                 gmr[3][4] <= ~
                                                    naked_sqs_mask
                                                    [1] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][4
                                                    ];
975                                                 gmr[3][5] <= ~
                                                    naked_sqs_mask
                                                    [2] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[3][5
                                                    ];
976                                                 gmr[4][3] <= ~
```

```
                                    naked_sqs_mask
                                    [3] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[4][3
                                    ];
977                             gmr[4][4] <= ~
                                    naked_sqs_mask
                                    [4] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[4][4
                                    ];
978                             gmr[4][5] <= ~
                                    naked_sqs_mask
                                    [5] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[4][5
                                    ];
979                             gmr[5][3] <= ~
                                    naked_sqs_mask
                                    [6] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[5][3
                                    ];
980                             gmr[5][4] <= ~
                                    naked_sqs_mask
                                    [7] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[5][4
                                    ];
981                             gmr[5][5] <= ~
                                    naked_sqs_mask
                                    [8] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[5][5
                                    ];
982                         end
983                 5:
984                         begin
985                             gmr[3][6] <= ~
                                    naked_sqs_mask
                                    [0] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[3][6
                                    ];
986                             gmr[3][7] <= ~
                                    naked_sqs_mask
                                    [1] ? ~pvr[
                                    row_counter]
                                    [col_counter
                                    ] : gmr[3][7
                                    ];
```

```
987                                     gmr[3][8] <= ~
                                            naked_sqs_mask
                                            [2] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[3][8
                                            ];
988                                     gmr[4][6] <= ~
                                            naked_sqs_mask
                                            [3] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[4][6
                                            ];
989                                     gmr[4][7] <= ~
                                            naked_sqs_mask
                                            [4] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[4][7
                                            ];
990                                     gmr[4][8] <= ~
                                            naked_sqs_mask
                                            [5] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[4][8
                                            ];
991                                     gmr[5][6] <= ~
                                            naked_sqs_mask
                                            [6] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[5][6
                                            ];
992                                     gmr[5][7] <= ~
                                            naked_sqs_mask
                                            [7] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[5][7
                                            ];
993                                     gmr[5][8] <= ~
                                            naked_sqs_mask
                                            [8] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[5][8
                                            ];
994                                 end
995                             6:
996                                 begin
997                                     gmr[6][0] <= ~
                                            naked_sqs_mask
                                            [0] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[6][0
```

```
                                                    ];
998                                             gmr[6][1] <= ~
                                                    naked_sqs_mask
                                                    [1] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[6][1
                                                    ];
999                                             gmr[6][2] <= ~
                                                    naked_sqs_mask
                                                    [2] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[6][2
                                                    ];
1000                                            gmr[7][0] <= ~
                                                    naked_sqs_mask
                                                    [3] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[7][0
                                                    ];
1001                                            gmr[7][1] <= ~
                                                    naked_sqs_mask
                                                    [4] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[7][1
                                                    ];
1002                                            gmr[7][2] <= ~
                                                    naked_sqs_mask
                                                    [5] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[7][2
                                                    ];
1003                                            gmr[8][0] <= ~
                                                    naked_sqs_mask
                                                    [6] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[8][0
                                                    ];
1004                                            gmr[8][1] <= ~
                                                    naked_sqs_mask
                                                    [7] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[8][1
                                                    ];
1005                                            gmr[8][2] <= ~
                                                    naked_sqs_mask
                                                    [8] ? ~pvr[
                                                    row_counter]
                                                    [col_counter
                                                    ] : gmr[8][2
                                                    ];
1006                                        end
```

```
1007                                    7:
1008                                        begin
1009                                            gmr[6][3] <= ~
                                                naked_sqs_mask
                                                [0] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[6][3
                                                ];
1010                                            gmr[6][4] <= ~
                                                naked_sqs_mask
                                                [1] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[6][4
                                                ];
1011                                            gmr[6][5] <= ~
                                                naked_sqs_mask
                                                [2] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[6][5
                                                ];
1012                                            gmr[7][3] <= ~
                                                naked_sqs_mask
                                                [3] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[7][3
                                                ];
1013                                            gmr[7][4] <= ~
                                                naked_sqs_mask
                                                [4] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[7][4
                                                ];
1014                                            gmr[7][5] <= ~
                                                naked_sqs_mask
                                                [5] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[7][5
                                                ];
1015                                            gmr[8][3] <= ~
                                                naked_sqs_mask
                                                [6] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[8][3
                                                ];
1016                                            gmr[8][4] <= ~
                                                naked_sqs_mask
                                                [7] ? ~pvr[
                                                row_counter]
                                                [col_counter
                                                ] : gmr[8][4
                                                ];
```

```
1017                                    gmr[8][5] <= ~
                                            naked_sqs_mask
                                            [8] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[8][5
                                            ];
1018                                end
1019                            8:
1020                                begin
1021                                    gmr[6][6] <= ~
                                            naked_sqs_mask
                                            [0] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[6][6
                                            ];
1022                                    gmr[6][7] <= ~
                                            naked_sqs_mask
                                            [1] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[6][7
                                            ];
1023                                    gmr[6][8] <= ~
                                            naked_sqs_mask
                                            [2] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[6][8
                                            ];
1024                                    gmr[7][6] <= ~
                                            naked_sqs_mask
                                            [3] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[7][6
                                            ];
1025                                    gmr[7][7] <= ~
                                            naked_sqs_mask
                                            [4] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[7][7
                                            ];
1026                                    gmr[7][8] <= ~
                                            naked_sqs_mask
                                            [5] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[7][8
                                            ];
1027                                    gmr[8][6] <= ~
                                            naked_sqs_mask
                                            [6] ? ~pvr[
                                            row_counter]
                                            [col_counter
                                            ] : gmr[8][6
```

```
                                                                ];
1028                                                        gmr[8][7] <= ~
                                                                naked_sqs_mask
                                                                [7] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[8][7
                                                                ];
1029                                                        gmr[8][8] <= ~
                                                                naked_sqs_mask
                                                                [8] ? ~pvr[
                                                                row_counter]
                                                                [col_counter
                                                                ] : gmr[8][8
                                                                ];
1030                                                end
1031                                            default: ;
1032                                    endcase
1033                            end
1034                    end
1035            end
1036
1037            // Candidate line alg
1038            reg [3:0] square_counter;
1039            always @(posedge clk_in)
1040            begin
1041                    if (reset_in)
1042                    begin
1043                            square_counter <= 0;
1044                            `RESET_3_BY_9(candidate_line_rows_reg);
1045                            `RESET_3_BY_9(candidate_line_cols_reg);
1046                    end
1047                    else if (|error_detected)
1048                    begin
1049                            `RESET_3_BY_9(candidate_line_rows_reg);
1050                            `RESET_3_BY_9(candidate_line_cols_reg);
1051                    end
1052                    else
1053                    begin
1054                            case (square_counter)
1055                                    4'd0:
1056                                            begin
1057                                                    candidate_line_rows_reg
                                                        [0][0] <=
                                                        get_exclusive_line_possibilities
                                                        (
1058
```

1059

1060

```
1061
1062                                                    candidate_line_rows_reg
                                                           [0][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1063
```

1064

1065

```
1066
1067                                                    candidate_line_rows_reg
                                                           [0][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1068
1069
```

```
1070
1071
1072                                    candidate_line_cols_reg
                                           [0][0] <=
                                           get_exclusive_line_possibilities
                                           (
1073
```

1074

1075

```
1076
1077                                                    candidate_line_cols_reg
                                                           [0][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1078
```

1079

1080

```
1081
1082                                                    candidate_line_cols_reg
                                                           [0][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1083
1084
```

```
1085
1086
1087                                                    end
1088                                        4'd1:
```

```
1089                                    begin
1090                                            candidate_line_rows_reg
                                                   [1][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1091
1092
```

```
1093
1094
1095                                                    candidate_line_rows_reg
                                                           [1][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1096
```

1097

1098

```
1099
1100                                    candidate_line_rows_reg
                                           [1][2] <=
                                           get_exclusive_line_possibilities
                                           (
1101
```

1102

1103

```
1104

1105                                                    candidate_line_cols_reg
                                                           [1][0] <=
                                                           get_exclusive_line_possibilities
                                                           (
1106

1107
```

```
                                        candidate_line_cols_reg
                                            [1][1] <=
                                            get_exclusive_line_possibilities
                                            (
```

1112

1113

```
1114

1115                                                    candidate_line_cols_reg
                                                           [1][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1116
```

1117

1118

```
1119
1120                                    end
1121                            4'd2:
1122                                    begin
1123                                            candidate_line_rows_reg
                                                    [2][0] <=
                                                    get_exclusive_line_possibilities
                                                    (
1124
1125
```

1126

1127

```
1128                                    candidate_line_rows_reg
                                           [2][1] <=
                                           get_exclusive_line_possibilities
                                           (
1129
1130
```

```
candidate_line_rows_reg
   [2][2] <=
   get_exclusive_line_possibilities
   (
```

1135

1136

```
1137

1138                                                    candidate_line_cols_reg
                                                           [2][0] <=
                                                           get_exclusive_line_possibilities
                                                           (
1139
```

1140

1141

```
1142
1143                                                    candidate_line_cols_reg
                                                           [2][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1144
1145
```

```
                                        candidate_line_cols_reg
                                            [2][2] <=
                                            get_exclusive_line_possibilities
                                            (
```

1150

1151

```
1152
1153                    end
1154            4'd3:
1155                    begin
1156                            candidate_line_rows_reg
                                    [3][0] <=
                                    get_exclusive_line_possibilities
                                    (
1157
```

1158

1159

```
                                        candidate_line_rows_reg
                                            [3][1] <=
                                            get_exclusive_line_possibilities
                                            (
```

1164

1165

```
1166                                    candidate_line_rows_reg
                                           [3][2] <=
                                           get_exclusive_line_possibilities
                                           (
1167
1168
```

```
1169
1170
1171                                                    candidate_line_cols_reg
                                                           [3][0] <=
                                                           get_exclusive_line_possibilities
                                                           (
1172
```

1173

1174

```
1175
1176                                                    candidate_line_cols_reg
                                                           [3][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1177
```

1178

1179

```
1180

1181                                                    candidate_line_cols_reg
                                                           [3][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1182

1183
```

```
1184
1185
1186                                    end
1187                            4'd4:
1188                                    begin
1189                                            candidate_line_rows_reg
                                                   [4][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1190
```

1191

```
1192
1193
1194                                                    candidate_line_rows_reg
                                                           [4][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1195
```

1196

1197

```
1198
1199                                                    candidate_line_rows_reg
                                                           [4][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1200
1201
```

1202

1203

```
candidate_line_cols_reg
    [4][0] <=
    get_exclusive_line_possibilities
    (
```

```
                                        candidate_line_cols_reg
                                            [4][1] <=
                                            get_exclusive_line_possibilities
                                            (
```

1211

1212

```
1213

1214                                                    candidate_line_cols_reg
                                                           [4][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1215
```

1216

1217

```
1218
1219                                    end
1220                            4'd5:
1221                                    begin
1222                                            candidate_line_rows_reg
                                                   [5][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1223
1224
```

```
                                        candidate_line_rows_reg
                                            [5][1] <=
                                            get_exclusive_line_possibilities
                                            (
```

1229

```
1230
1231
1232                                    candidate_line_rows_reg
                                           [5][2] <=
                                           get_exclusive_line_possibilities
                                           (
1233
```

1234

1235

```
1236

1237                                                    candidate_line_cols_reg
                                                           [5][0] <=
                                                           get_exclusive_line_possibilities
                                                           (
1238

1239
```

1240

1241

```
1242                                    candidate_line_cols_reg
                                           [5][1] <=
                                           get_exclusive_line_possibilities
                                           (
1243
1244
```

```
1245
1246
1247                                                    candidate_line_cols_reg
                                                           [5][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1248
```

1249

1250

```
1251
1252                                    end
1253                            4'd6:
1254                                    begin
1255                                            candidate_line_rows_reg
                                                   [6][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1256
```

1257

1258

```
1259
1260                                    candidate_line_rows_reg
                                           [6][1] <=
                                           get_exclusive_line_possibilities
                                           (
1261
1262
```

```
                                        candidate_line_rows_reg
                                            [6][2] <=
                                            get_exclusive_line_possibilities
                                            (
```

1267

```
1268
1269
1270                                    candidate_line_cols_reg
                                           [6][0] <=
                                           get_exclusive_line_possibilities
                                           (
1271
```

1272

1273

```
1274

1275                                                    candidate_line_cols_reg
                                                           [6][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1276

1277
```

1278

1279

```
                candidate_line_cols_reg
                    [6][2] <=
                    get_exclusive_line_possibilities
                    (
```

```
1283
1284
1285                                    end
1286                            4'd7:
1287                                    begin
1288                                            candidate_line_rows_reg
                                                   [7][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1289
```

1290

1291

```
1292
1293                                                    candidate_line_rows_reg
                                                           [7][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1294
```

1295

1296

```
1297
1298                                    candidate_line_rows_reg
                                           [7][2] <=
                                           get_exclusive_line_possibilities
                                           (
1299
1300
```

```
                                        candidate_line_cols_reg
                                            [7][0] <=
                                            get_exclusive_line_possibilities
                                            (
```

1305

```
1306
1307
1308                                                    candidate_line_cols_reg
                                                           [7][1] <=
                                                           get_exclusive_line_possibilities
                                                           (
1309
```

1310

1311

```
1312
1313                                                    candidate_line_cols_reg
                                                           [7][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1314
1315
```

1316

1317

```
1318                                    end
1319                            4'd8:
1320                                    begin
1321                                            candidate_line_rows_reg
                                                   [8][0] <=
                                                   get_exclusive_line_possibilities
                                                   (
1322
1323
```

```
                                candidate_line_rows_reg
                                   [8][1] <=
                                   get_exclusive_line_possibilities
                                   (
```

1328

1329

```
1330
1331                                                    candidate_line_rows_reg
                                                           [8][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1332
```

1333

1334

```
1335
1336                                                    candidate_line_cols_reg
                                                           [8][0] <=
                                                           get_exclusive_line_possibilities
                                                           (
1337
1338
```

```
                                        candidate_line_cols_reg
                                            [8][1] <=
                                            get_exclusive_line_possibilities
                                            (
```

1343

```
1344

1345

1346                                                    candidate_line_cols_reg
                                                           [8][2] <=
                                                           get_exclusive_line_possibilities
                                                           (
1347
```

1348

1349

```
1350
1351                                    end
1352                            default: ;
1353                    endcase
1354
1355                    if (square_counter < 8)
1356                    begin
1357                            square_counter <= square_counter + 1;
1358                    end
1359                    else
1360                    begin
1361                            square_counter <= 0;
1362                    end
1363            end
1364    end
1365
1366    // the lowest number of ones in the grid
1367    reg     [3:0]
           min_ones_reg;
1368    always @(posedge clk_in)
1369    begin
1370            if (reset_in)
1371            begin
1372                    min_ones_reg <= 4'd10;
1373                    guess_number <= 16'b0;
1374                    `RESET_PREVS(guess_col, 10);
1375                    `RESET_PREVS(guess_row, 10);
1376                    `RESET_PREVS(guesses, 0);
1377            end
1378            else
1379            begin
1380                    if ((|error_detected) & (|guess_number))
1381                    begin //error at guess_number-1
1382                            min_ones_reg <= 4'd10;
1383                            guess_number <= guess_number - 1;
1384                            guesses[guess_number-1]    <= `GET_LSB(
                                   pvr_prevs[guess_row[guess_number-1]]
                                   [guess_col[guess_number-1]][
                                   guess_number-1]);
1385                    end
1386                    else if (timeout)
```

```
                        begin
                                // if ((guess_row[guess_number] == 10)
                                   & ~(done_out))
                                // begin
                                //      $display("INVALID GUESS");
                                //      $finish;
                                // end
                                min_ones_reg <= 4'd10;
                                guess_number <= guess_number + 1;
                        end
                        if ( ( (number_of_ones_in_mask < min_ones_reg)
                           && (number_of_ones_in_mask > 1) )
                                || ((~valid_one_hot(pvr[row_counter][
                                   col_counter]))
                                     && valid_one_hot(pvr[guess_row[
                                        guess_number]][guess_col[
                                        guess_number]]) ) )
                        begin
                                min_ones_reg                    <=
                                   number_of_ones_in_mask;
                                guesses[guess_number]   <= `GET_LSB(pvr
                                   [row_counter][col_counter]);
                                guess_col[guess_number] <= col_counter;
                                guess_row[guess_number] <= row_counter;
                        end
                end
        end
endmodule
```

```
`timescale 1ns / 1ps
//`default_nettype none

module top(
    input wire CLK_100M,
        input wire [15:0] SW,
        output wire [15:0] LED,
        output wire RGB1_Blue, RGB1_Green, RGB1_Red,
        output wire RGB2_Blue, RGB2_Green, RGB2_Red,
        output wire [7:0] SEG, [7:0] AN, //7 segment LED display
        input wire CPU_RESETN, BTNC, BTNU, BTNL, BTNR, BTND, //buttons
        inout wire [7:0] JA, JB, JC, JD, //PMOD headers
        //input wire [3:0] XA_N, XA_P, //analog inputs
        output wire [3:0] VGA_R, VGA_G, VGA_B, //VGA outputs
        output wire VGA_HS, VGA_VS
    );
    localparam   GRID_SIZE = 9;

    wire reset;
//    assign reset = 0;

    wire video_clk;

    wire VGA_VSYNC;
    assign VGA_VS = VGA_VSYNC;

    //camera signals
    wire camera_pwdn;
    wire camera_clk_in;
```

```
30      wire camera_clk_out;
31      wire [7:0] camera_dout;
32      wire camera_scl, camera_sda;
33      wire camera_vsync, camera_hsync;
34      wire [15:0] camera_pixel;
35      wire camera_pixel_valid;
36      wire camera_reset;
37      wire camera_frame_done;
38
39      assign camera_clk_in = video_clk;
40      assign camera_pwdn = 0;
41      assign camera_reset = 1'b1;//~reset;
42
43  //assign camera outputs
44     assign JA[0] = camera_pwdn;
45     assign camera_dout[0] = JA[1];
46     assign camera_dout[2] = JA[2];
47     assign camera_dout[4] = JA[3];
48     assign JA[4] = camera_reset;
49     assign camera_dout[1] = JA[5];
50     assign camera_dout[3] = JA[6];
51     assign camera_dout[5] = JA[7];
52
53     assign camera_dout[6] = JB[0];
54     assign JB[1] = camera_clk_in;
55     assign camera_hsync = JB[2];
56     assign JB[3]= camera_sda;
57     assign camera_dout[7] = JB[4];
58     assign camera_clk_out = JB[7];
59     assign camera_vsync = JB[5];
60     //assign JB[7] = camera_scl;
61
62     wire [11:0] memory_read_data;
63     wire [11:0] memory_write_data;
64
65     wire [18:0] video_read_addr;
66     wire [18:0] memory_read_addr;
67     wire [18:0] img_read_addr;
68
69  //    assign img_read_addr = 0;
70
71     wire [18:0] memory_write_addr;
72     wire memory_write_enable;
73
74     // No longer camera shit lmao
75
76     //  instantiate 7-segment display;
77     wire [31:0] data;
78     wire [6:0] segments;
79     display_8hex display(.clk(video_clk),.data(data), .seg(segments), .
           strobe(AN));
80     assign SEG[6:0] = segments;
81     assign SEG[7] = 1'b1;
82
83     reg[3:0] state = 1;
84
85      // 104
86     reg[9:0] x1 = 110;
```

```
87      reg[9:0] y1 = 24;
88      reg[9:0] x2 = 541;
89      reg[9:0] y2 = 455;
90
91      // Synchronizers and power-on-reset
92
93      pwr_reset reset_1 (.clk(CLK_100M), .reset_input(SW[15]), .reset(
            reset));
94
95       // FSM shit
96
97
98       wire[323:0] recg_sudoku;
99
100      reg [4*(GRID_SIZE)*(GRID_SIZE)-1:0] board = {4'd0, 4'd0, 4'd4, 4'd0
             , 4'd0, 4'd0, 4'd0, 4'd9, 4'd0,
101                                  4'd0, 4'd1, 4'd0, 4'd0, 4'd0, 4'd5, 4'
                                         d8, 4'd0, 4'd0,
102                                  4'd8, 4'd6, 4'd0, 4'd0, 4'd0, 4'd0, 4'
                                         d1, 4'd0, 4'd0,
103                                  4'd0, 4'd3, 4'd0, 4'd0, 4'd0, 4'd1, 4'
                                         d0, 4'd0, 4'd0,
104                                  4'd0, 4'd0, 4'd7, 4'd5, 4'd4, 4'd0, 4'
                                         d0, 4'd0, 4'd0,
105                                  4'd0, 4'd0, 4'd0, 4'd7, 4'd0, 4'd0, 4'
                                         d0, 4'd5, 4'd0,
106                                  4'd0, 4'd0, 4'd2, 4'd0, 4'd9, 4'd0, 4'
                                         d0, 4'd7, 4'd0,
107                                  4'd0, 4'd0, 4'd0, 4'd0, 4'd0, 4'd6, 4'
                                         d3, 4'd0, 4'd0,
108                                  4'd0, 4'd0, 4'd0, 4'd0, 4'd0, 4'd0, 4'
                                         d0, 4'd0, 4'd8};
109      wire [4*(GRID_SIZE)*(GRID_SIZE)-1:0] board_solved;
110
111      wire sudoku_invalid;
112      wire sudoku_done;
113 //     wire sudoku_reset = ~SW[3];
114
115      reg[3:0] selected_x = 4;
116      reg[3:0] selected_y = 4;
117
118      reg  [4:0] reset_reg = 0;
119      always @(posedge video_clk)
120      begin
121          reset_reg = {reset_reg[3], reset_reg[2], reset_reg[1],
                reset_reg[0], (btnc_rise && state == SOLVING)};
122      end
123
124      wire[323:0] board_solved_100m;
125      wire done_100m;
126      wire sudoku_invalid_100m;
127
128 //     synchronize sync_1 (.clk(CLK_100M), .in(board), .out(board_sync))
     ;
129
130      soduku_solver my_love(.clk_in(video_clk), .reset_in(reset_reg[4]),
            .board_in(board), .board_out(board_solved), .done_out(
            sudoku_done), .invalid_out(sudoku_invalid));
```

```
131
132 //    synchronize sync_2 (.clk(video_clk), .in(board_solved_100m), .out
    (board_solved));
133 //    synchronize sync_3 (.clk(video_clk), .in(done_100m), .out(
    sudoku_done));
134 //    synchronize sync_4 (.clk(video_clk), .in(sudoku_invalid_100m), .
    out(sudoku_invalid));
135
136
137     // States
138     parameter IDLE = 0;
139     parameter CHOOSE_XY1 = 1;
140     parameter CHOOSE_XY2 = 2;
141     parameter RESIZING = 3;
142     parameter RECOGNIZING = 4;
143     parameter CONFIRMING = 5;
144     parameter FIXING = 6;
145     parameter SOLVING = 7;
146     parameter OUTPUT = 8;
147     parameter TUTORIAL = 9;
148
149     assign state_out = state;
150
151     wire btnc_cln;
152     wire btnl_cln;
153     wire btnr_cln;
154     wire btnu_cln;
155     wire btnd_cln;
156
157     debounce deb_btnc (.clk(video_clk), .reset(reset), .noisy(BTNC), .
        clean(btnc_cln));
158
159     debounce deb_btnl (.clk(video_clk), .reset(reset), .noisy(BTNL), .
        clean(btnl_cln));
160     debounce deb_btnr (.clk(video_clk), .reset(reset), .noisy(BTNR), .
        clean(btnr_cln));
161     debounce deb_btnu (.clk(video_clk), .reset(reset), .noisy(BTNU), .
        clean(btnu_cln));
162     debounce deb_btnd (.clk(video_clk), .reset(reset), .noisy(BTND), .
        clean(btnd_cln));
163
164     wire btnc_held;
165
166     debounce #(.DELAY(1_000_000)) deb_btnc_held (.clk(video_clk), .
        reset(reset), .noisy(BTNC), .clean(btnc_held));
167
168 //    assign LED[15] = btnc_held;
169
170     wire btnl_rise;
171     wire btnr_rise;
172     wire btnu_rise;
173     wire btnd_rise;
174
175     rise btnl_rise_1 (.clk(video_clk), .in(btnl_cln), .out(btnl_rise));
176     rise btnr_rise_1 (.clk(video_clk), .in(btnr_cln), .out(btnr_rise));
177     rise btnu_rise_1 (.clk(video_clk), .in(btnu_cln), .out(btnu_rise));
178     rise btnd_rise_1 (.clk(video_clk), .in(btnd_cln), .out(btnd_rise));
179
```

```
180     reg frame_parser_start = 0;
181
182     reg last_btnc = 0;
183     wire btnc_rise = (btnc_cln && ~last_btnc);
184
185     wire clk_ps;
186     clk_prescale clk_prescale_1 (.clk(video_clk), .clk_ps(clk_ps));
187     wire frame_parser_done;
188     reg char_rec_start = 0;
189
190     reg frame_parser_started = 0;
191     reg char_rec_started = 0;
192     wire char_rec_done;
193
194     reg[3:0] tutorial_guess = 1;
195
196     always @(posedge video_clk) begin
197         if(SW[15]) begin
198             state <= 0;//IDLE;
199             frame_parser_started <= 0;
200             char_rec_started <= 0;
201         end
202
203         else if (VGA_VS) begin
204             case(state)
205                 IDLE: begin
206                     if(btnc_rise) begin
207                         state <= CHOOSE_XY1;
208                     end
209                 end
210                 CHOOSE_XY1: begin
211                    if(btnc_rise) begin
212                       state <= CHOOSE_XY2;
213                     end else if(clk_ps) begin
214                        if (btnl_cln) begin
215                             x1 <= x1 - 1;
216                         end else if (btnr_cln) begin
217                             x1 <= x1 + 1;
218                         end else if (btnu_cln) begin
219                             y1 <= y1 - 1;
220                         end else if (btnd_cln) begin
221                             y1 <= y1 + 1;
222                         end
223                     end
224                 end
225                 CHOOSE_XY2: begin
226                    if(btnc_rise) begin
227                      state <= RESIZING;
228                    end else if(clk_ps) begin
229                       if (btnl_cln) begin
230                            x2 <= x2 - 1;
231                        end else if (btnr_cln) begin
232                            x2 <= x2 + 1;
233                        end else if (btnu_cln) begin
234                            y2 <= y2 - 1;
235                        end else if (btnd_cln) begin
236                            y2 <= y2 + 1;
237                        end
```

```
                    end
                end
                RESIZING: begin
                    if(btnr_cln) begin
                        frame_parser_start <= 1;
                        frame_parser_started <= 1;
                    end else begin
                        frame_parser_start <= 0;
                    end

//                    if(~frame_parser_started) begin
//                        frame_parser_start <= 1;
//                        frame_parser_started <= 1;
//                    end

                    if(frame_parser_done && frame_parser_started) begin
                        state <= RECOGNIZING;
                        frame_parser_start <= 0;
                    end
                end
                RECOGNIZING: begin
//                    if(~char_rec_started) begin
//                        char_rec_start <= 1;
//                        char_rec_started <= 1;
//                    end

                    if(btnl_cln) begin
                        char_rec_start <= 1;
                        char_rec_started <= 1;
                    end

                    if(char_rec_done && char_rec_started) begin
                        board <= recg_sudoku;
                        state <= FIXING;
                    end
                end
                CONFIRMING: begin
                    if(btnl_rise) begin
                        x1 <= x1 - 1;
                        x2 <= x2 - 1;
                        state <= RESIZING;
                        frame_parser_started <= 0;
                        char_rec_started <= 0;
                    end else if (btnr_rise) begin
                        x1 <= x1 + 1;
                        x2 <= x2 + 1;
                        state <= RESIZING;
                        frame_parser_started <= 0;
                        char_rec_started <= 0;
                    end else if (btnu_rise) begin
                        y1 <= y1 - 1;
                        y2 <= y2 - 1;
                        state <= RESIZING;
                        frame_parser_started <= 0;
                        char_rec_started <= 0;
                    end else if (btnd_rise) begin
                        y1 <= y1 + 1;
                        y2 <= y2 + 1;
```

```
296                         state <= RESIZING;
297                         frame_parser_started <= 0;
298                         char_rec_started <= 0;
299                     end else if (btnc_rise) begin
300                         state <= FIXING;
301                     end
302                 end
303                 FIXING: begin
304                     if(SW[14]) begin
305                         if(btnu_rise) begin
306                             board[(4 * (selected_x + (selected_y *
                                GRID_SIZE)))+3-:4] =
307                                 board[(4 * (selected_x + (selected_y *
                                    GRID_SIZE)))+3-:4] < 9 ?
308                                 {board[(4 * (selected_x + (selected_y *
                                     GRID_SIZE)))+3-:4] + 1} : 0;
309                         end else if (btnd_rise) begin
310                             board[(4 * (selected_x + (selected_y *
                                GRID_SIZE)))+3-:4] =
311                                 board[(4 * (selected_x + (selected_y *
                                    GRID_SIZE)))+3-:4] > 0 ?
312                                 {board[(4 * (selected_x + (selected_y *
                                     GRID_SIZE)))+3-:4] - 1} : 9;
313                         end
314                     end else begin
315                         if(btnl_rise && selected_x > 0) begin
316                             selected_x <= selected_x - 1;
317                         end else if (btnr_rise && selected_x < 8) begin
318                             selected_x <= selected_x + 1;
319                         end else if (btnu_rise && selected_y > 0) begin
320                             selected_y <= selected_y - 1;
321                         end else if (btnd_rise && selected_y < 8) begin
322                             selected_y <= selected_y + 1;
323                         end else if (btnc_held) begin
324                             state <= SOLVING;
325                         end
326                     end
327                 end
328                 SOLVING: begin
329                     if(btnc_rise) begin
330                         state <= OUTPUT;
331                     end
332                 end
333                 OUTPUT: begin
334 //                      if(sudoku_invalid) begin
335 //                          state <= FIXING;
336 //                      end
337                     if(SW[14]) begin
338                         state <= TUTORIAL;
339                     end
340                 end
341                 TUTORIAL: begin
342                     if(SW[14]) begin
343                         if(btnu_rise) begin
344                             if(tutorial_guess < 9) tutorial_guess <=
                                tutorial_guess + 1;
345                         end else if (btnd_rise) begin
346                             if(tutorial_guess > 0) tutorial_guess <=
```

```
                                tutorial_guess - 1;
                        end else if (btnc_rise) begin
                            board[(4 * (selected_x + (selected_y *
                                GRID_SIZE)))+3-:4] = tutorial_guess;
                        end else if (btnl_rise) begin
                            board[(4 * (selected_x + (selected_y *
                                GRID_SIZE)))+3-:4] = 0;
                        end
                    end else begin
                        if(btnl_rise && selected_x > 0) begin
                            selected_x <= selected_x - 1;
                        end else if (btnr_rise && selected_x < 8) begin
                            selected_x <= selected_x + 1;
                        end else if (btnu_rise && selected_y > 0) begin
                            selected_y <= selected_y - 1;
                        end else if (btnd_rise && selected_y < 8) begin
                            selected_y <= selected_y + 1;
                        end
                    end
                end
            endcase
        end
        last_btnc <= btnc_cln;
        if(char_rec_start) char_rec_start <= 0;
        if(frame_parser_start) frame_parser_start <= 0;
    end

//    assign wrong_guess = SW[4];
   // Camera shit

    //clock generation
    video_clk video_clk_1 (
        .clk_in1(CLK_100M),
        .clk_out1(video_clk)
         );

    wrong_guess_gen wrong_guess_gen_1 (
        .input_board(board),
        .solved_board(board_solved),
        .state(state),
        .wrong_guess(wrong_guess));

    camera camera_1 (
        .video_clk(video_clk),
        .camera_start(SW[15]),
        .sioc(JB[6]),
        .siod(JB[3]),
        .capture_frame(state == 0),
        .camera_vsync(camera_vsync),
        .camera_hsync(camera_hsync),
        .camera_dout(camera_dout),
        .camera_clk(camera_clk_out),
        .camera_pixel(camera_pixel),
        .camera_pixel_valid(camera_pixel_valid),
        .camera_frame_done(camera_frame_done),
        .memory_write_data(memory_write_data),
        .memory_write_addr(memory_write_addr),
        .memory_write_enable(memory_write_enable));
```

```
402
403     wire[11:0] rescaled_pix_data;
404     wire[11:0] rescaled_fb_dout;
405     wire switch_vid = SW[1];
406
407     wire[9:0] hcount;
408     wire[9:0] vcount;
409     wire blank;
410     wire use_staff = SW[10];
411
412     video_playback video_playback_1 (
413         .pixel_data(memory_read_data),
414         .rescaled_pix_data(rescaled_fb_dout),
415         .video_clk(video_clk),
416         .memory_addr(video_read_addr),
417         .vsync(VGA_VSYNC),
418         .hsync(VGA_HS),
419         .hcount_out(hcount),
420         .vcount_out(vcount),
421         .blank_out(blank),
422         .video_out({VGA_R, VGA_G, VGA_B}),
423         .x1(x1), .y1(y1), .x2(x2), .y2(y2),
424         .state(state),
425         .switch_vid(switch_vid),
426         .board_in((state == OUTPUT) ? board_solved : board),
427         .selected_x(selected_x),
428         .selected_y(selected_y),
429         .wrong_guess(wrong_guess),
430         .use_staff(use_staff)
431         );
432
433
434
435     // Memory shit
436
437     assign memory_read_addr = (state == 0 || state == 1 || state == 2)
        ? video_read_addr : img_read_addr;
438
439     //frame buffer memory
440     frame_buffer frame_buffer_1 (
441         .clka(camera_clk_out),
442         .wea(memory_write_enable),
443         .addra(memory_write_addr),
444         .dina(memory_write_data),
445         .clkb(video_clk),
446         //.enb(1'b1),
447         .addrb(memory_read_addr),
448         .doutb(memory_read_data)
449         );
450
451     wire rescaled_fb_we;
452     wire[14:0] rescaled_fb_vid_addr = video_read_addr[14:0];
453     wire[14:0] rescaled_fb_write_addr;
454     wire[14:0] char_ram_addr;
455
456     wire[11:0] rescaled_fb_din;
457
458     wire[14:0] rescaled_fb_addr =
```

```
459             switch_vid                                          ?
                   rescaled_fb_vid_addr :
460             ((~(SW[2])) ? rescaled_fb_write_addr : char_ram_addr);
461
462     assign LED[0] = switch_vid;
463     assign LED[1] = (state == RESIZING);
464
465     frame_parser frame_parser_1 (
466         .clk(video_clk),
467         .start(frame_parser_start),
468         .x1(x1), .y1(y1),
469         .x2(x2), .y2(y2),
470
471         .img_read_data_in(memory_read_data),
472         .img_read_addr_out(img_read_addr),
473
474         .img_write_data_out(rescaled_fb_din),
475         .img_write_addr_out(rescaled_fb_write_addr),
476
477         .we_out(rescaled_fb_we),
478
479         .done(frame_parser_done)
480         );
481
482     rescaled_frame_buffer rescaled_fb_1 (
483         .clka(video_clk),
484         .addra(rescaled_fb_addr),
485         .dina(rescaled_fb_din),
486         .douta(rescaled_fb_dout),
487         .wea(rescaled_fb_we));
488
489     char_rec char_rec_1 (
490         .clk(video_clk),
491         .start(char_rec_start),
492         .done(char_rec_done),
493         .img_ram_addr(char_ram_addr),
494         .img_ram_data(rescaled_fb_dout),
495         .recg_sudoku(recg_sudoku));
496
497
498     assign LED[2] = sudoku_done;
499     assign data[3:0] = state;
500     assign data[31:28] = tutorial_guess;
501 endmodule
```

```
1  module video_playback(
2      input wire [11:0] pixel_data,
3      input wire [11:0] rescaled_pix_data,
4      input wire video_clk,
5      output wire [18:0] memory_addr,
6      output reg vsync,
7      output reg hsync,
8      output wire [11:0] video_out,
9      output[9:0] hcount_out,
10     output[9:0] vcount_out,
11     output blank_out,
12     input[9:0] x1, y1,
13     input[9:0] x2, y2,
```

```
14      input[3:0] state,
15      input switch_vid,
16      input[323:0] board_in,
17      input[3:0] selected_x,
18      input[3:0] selected_y,
19      input wrong_guess,
20      input use_staff
21      );
22
23      `include "display_lib.v"
24
25      parameter IDLE = 0;
26      parameter CHOOSE_XY1 = 1;
27      parameter CHOOSE_XY2 = 2;
28      parameter TARGET_WIDTH = 144;
29
30      localparam SCREEN_WIDTH = 640;
31      localparam SCREEN_HEIGHT = 480;
32      localparam CELL_PIXELS = 48;
33      localparam GRID_PIXELS = CELL_PIXELS*9;
34      localparam GRID_START_X = (SCREEN_WIDTH - GRID_PIXELS)/2;
35      localparam GRID_START_Y = (SCREEN_HEIGHT - GRID_PIXELS)/2;
36
37      // horizontal: 800 pixels total
38      // display 640 pixels per line
39      reg hblank,vblank;
40      wire hsyncon,hsyncoff,hreset,hblankon;
41      reg [9:0] hcount = 0;
42      reg [9:0] vcount = 0;
43      reg blank;
44
45      assign blank_out = blank;
46      assign hcount_out = hcount;
47      assign vcount_out = vcount;
48
49      wire[11:0] rgb_out;
50
51      display_grid #(.CELL_PIXELS(CELL_PIXELS)) display_grid_1 (
52                  .clk_in(video_clk),
53                  .x_in(hcount - GRID_START_X),
54                  .y_in(vcount - GRID_START_Y),
55                  .board_in(board_in),
56                  .rgb_out(rgb_out),
57                  .selected_x(selected_x),
58                  .selected_y(selected_y),
59                  .state(state),
60                  .use_staff(use_staff));
61
62      //kludges to fix frame alignment due to memory access time
63      reg blank_delay;
64      reg blank_delay_2;
65      reg hsync_pre_delay;
66      reg hsync_pre_delay_2;
67      reg vsync_pre_delay;
68      reg vsync_pre_delay_2;
69
70      wire[11:0] sudoku_rgb;
71
```

```
72      assign sudoku_rgb = `INSIDE(hcount,
73                                  vcount,
74                                  GRID_START_X,
75                                  GRID_START_Y,
76                                  GRID_PIXELS,
77                                  GRID_PIXELS) ?  rgb_out : (wrong_guess) ?
                                       12'hF00 : 12'h000;
78
79      // Excessive? Maybe.
80      assign video_out = blank_delay_2
                                                        ? 12'b0              :
81  //                            (state == 5 && (hcount > GRID_START_Y +
        GRID_PIXELS || hcount < GRID_START_X)) ? 12'hFFF  :
82                              (state == 5 || state == 6 || state == 7 || state
                                    == 8 || state == 9)                ?
                                   sudoku_rgb           :
83                              (switch_vid)
                                                                          ?
                                   (rescaled_pix_data[3:0] + rescaled_pix_data[7
                                   :4] + rescaled_pix_data[11:8] > 20 ? 12'hFFF
                                   : 12'h000) :
84                              ((hcount == x1 || vcount == y1) && state ==
                                   CHOOSE_XY1) ? ~pixel_data       :
85                              ((hcount == x2 || vcount == y2) && state ==
                                   CHOOSE_XY2) ? ~pixel_data       :
86                              ((hcount == x1 || vcount == y1) && state ==
                                   CHOOSE_XY2) ? 12'hF00           : pixel_data;
87
88
89      assign hblankon = (hcount == 639);    //blank after display width
90      assign hsyncon = (hcount == 655);  // active video + front porch
91      assign hsyncoff = (hcount == 751); //active video + front portch +
            sync
92      assign hreset = (hcount == 799); //plus back porch
93
94      // vertical:  525 lines total
95      // display 480 lines
96      wire vsyncon,vsyncoff,vreset,vblankon;
97      assign vblankon = hreset & (vcount == 479);
98      assign vsyncon = hreset & (vcount == 489);
99      assign vsyncoff = hreset & (vcount == 491);
100     assign vreset = (hreset & (vcount == 524));
101
102     // sync and blanking
103     wire next_hblank,next_vblank;
104     assign next_hblank = hreset ? 0 : hblankon ? 1 : hblank;
105     assign next_vblank = vreset ? 0 : vblankon ? 1 : vblank;
106     // trust in this line
107     assign memory_addr = (switch_vid) ? (((hcount < TARGET_WIDTH) && (
            vcount < TARGET_WIDTH)) ? hcount+(vcount*TARGET_WIDTH) : 0) :
            hcount+(vcount*640);
108
109     always @(posedge video_clk) begin
110         blank_delay <= blank;
111         blank_delay_2 <= blank_delay;
112         hsync_pre_delay_2 <= hsync_pre_delay;
113         vsync_pre_delay_2 <= vsync_pre_delay;
114         vsync <= vsync_pre_delay_2;
```

```
115         hsync <= hsync_pre_delay_2;
116          //hcount
117          hcount <= hreset ? 0 : hcount + 1;
118          hblank <= next_hblank;
119          hsync_pre_delay <= hsyncon ? 0 : hsyncoff ? 1 :
                hsync_pre_delay;  // active low
120
121          vcount <= hreset ? (vreset ? 0 : vcount + 1) : vcount;
122          vblank <= next_vblank;
123          vsync_pre_delay <= vsyncon ? 0 : vsyncoff ? 1 :
                vsync_pre_delay;  // active low
124
125          blank <= next_vblank | (next_hblank & ~hreset);
126     end
127 endmodule
```

```
1  `timescale 1ns / 1ps
2
3
4  module wrong_guess_gen(
5          input[323:0] input_board,
6          input[323:0] solved_board,
7          input[3:0] state,
8          output wrong_guess
9      );
10
11 wire[80:0] different;
12
13 genvar i;
14 generate
15
16 for(i = 0; i < 81; i = i + 1) begin: diff_gen
17     assign different[i] = (input_board[(4 * i) + 3-:4] != solved_board[
           (4 * i) + 3-:4]) && (input_board[(4 * i) + 3-:4]);
18 end
19 endgenerate
20
21 assign wrong_guess = (|different) && (state == 9);
22 endmodule
```