Application Note

Complete Simulation of Single Chip Camera Design
("Light In - NTSC Out")

Desert Microtechnology Associates, Inc.
16611 North 91st Street, Suite 103
Scottsdale, AZ 85260-1523 USA
Phone: 480/515-0005 FAX: 480/515-0788
Email: ngilbert@desertmicro.com

1. Introduction

With the increasing complexity of mixed-signal chip design and the increase in mask costs, the need to perform full-chip simulations has become a virtual necessity. The time required to perform full-chip simulations in SPICE simulators, and even in Verilog-A simulators, is oftentimes overwhelming and does not give the engineer feedback quick enough to make meaningful corrections and gain a better understanding of how the designed circuit functions in the system. The true mixed-signal simulation capability of SMASH™ allows full-chip simulations through the use of mixed-language simulation. This allows the designer to model each portion of the entire design with varying degrees of accuracy depending on the aspect of the design that is being simulated and the time required to get data back for analysis.

SMASH™ supports a variety of languages, allowing the designer to model the continuous time and event driven portions of the design at hand. For more accurate simulations, SPICE models are preferred. SMASH™ also provides GUITAR (Graphic User Interface for Trading-off Accuracy and Rapidity) to help with the accuracy versus time trade-off in transient simulations. To speed up the simulations and understand functionality of the analog designs, macro-modeling languages such as ABCD (a C-based modeling language particular to SMASH™), VHDL-AMS, and Verilog-A/Verilog-AMS are available depending on the designer's preference. For logic simulations, both Verilog and VHDL are available. For this application note, the designer decided to use Verilog and Verilog-A for the macro-modeling of the event driven and continuous time circuits. This decision was based on the amount of code already available to the designer in these languages and his familiarity with these languages. This also demonstrates the flexibility of SMASH™ to supply the user with various language choices so that he can perform his tasks with the greatest efficiency.

In this application note, an example of a full-chip simulation is given. The circuit used in the example is a complex single-chip video camera. The input to the device is light, through a color CMOS pixel array, and the output is analog NTSC video. The operation of the chip is first explained to give the reader a basic understanding of the design. Then the process used to decide how the chip should be modeled and which parts should be modeled in Verilog, Verilog-A, or SPICE is discussed. This discussion focuses on two simulations: the simulation of one line of video exploring circuit interactions and a simulation of the full chip exploring image artifact due to timing in simulation.

The advantages of SMASH™ when performing these simulations are presented throughout and are summarized at the end of the document.

2. Circuit Operation

The example circuit used to illustrate the mixed-signal capability of SMASH is a full NTSC video camera on a single chip. The design includes a CMOS pixel array, a sense amplifier and an Analog Signal Processor (ASP). The operation of each section is controlled with a digital timing controller.

Figure 1 shows the design block diagram.

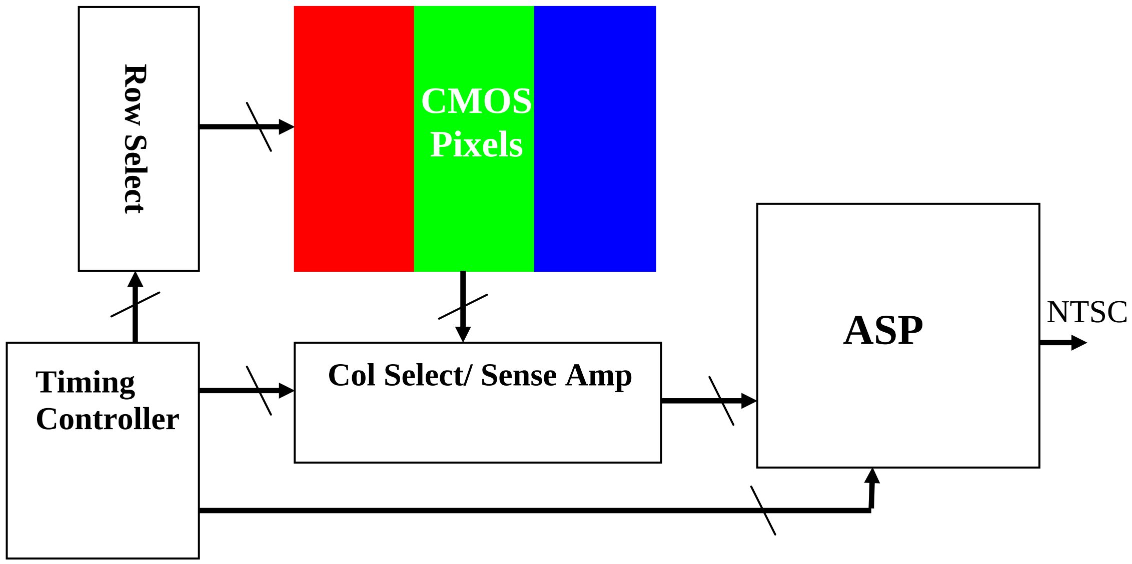

Figure 1: Block diagram of single chip image sensor with NTSC video output.

The signal path begins with the exposure of the CMOS pixels to light controlled by the Row Select. The Row select also chooses the row to read, which is then amplified by the Sense Amplifiers in the column. The signal is then processed by the ASP to provide an analog NTSC video signal. The ASP consists of gain control, gamma correction and a RGB-to-NTSC encoder. The timing for the Row Select, Sense Amplifier, and ASP are controlled by the Timing Controller circuit.

The two circuit simulations chosen to demonstrate the mixed-mode capability of SMASH™ are a Verilog-A/Verilog/SPICE simulation of one line of video and a Verilog-A/Verilog simulation of the full-chip to discover image artifacts in simulation. The simulation of a Single Line of Video (which will be referred to as SLV) consists of a single row select, a single row of pixels, the Sense Amplifiers, and part of the Timing Controller and the ASP. This subset of the full simulation is used to focus on the high speed operations within a single line of video. It also allows more accurate simulations of various blocks.
The Full-Chip Simulation (referred to as FCS) allows testing the functionality of the circuit and can be used for algorithmic testing such as the Automatic Exposure control (AE) and Automatic White Balance (AWB).

3. Choice of Simulation Language

From the block diagram of the design in **Figure 1**, it is clear both analog and logic circuits are needed to accomplish the task at hand. The real question is what languages should be used to model each section of the circuit. The answer to this question is completely dependant on the desired outcome of the simulation. The image sensor operations are happening on human time scales while the processing is occurring in fractions of micro seconds which leads to inherently long simulation time and painstaking trade-offs between simulation time and accuracy. Clearly, block-level SPICE simulations of the key circuit elements are the highest property in the design phase but far too often the interaction of these blocks is passed over in simulation due to the CPU time and engineering time needed to set-up the simulation and run it. For instance, on a 3.5 GHz processor with 2 GB of SDRAM, for carrying out a SPICE-level simulation of only 63 pixels and just the ASP – where all of the timing was implemented with ideal pulse generators –, more than 10 hours were needed just to run the simulation. When this same circuit was run with all the blocks implemented with Verilog-A, except for the 63 pixels, only 30 minutes of simulation time was needed. This clearly shows the time advantage of high level simulation with mixed-language simulation. Another great advantage is that the Verilog-A simulation could be performed using the same set of schematics as the SPICE simulations. Also, the same simulator, SMASH™, can be used for both simulations.

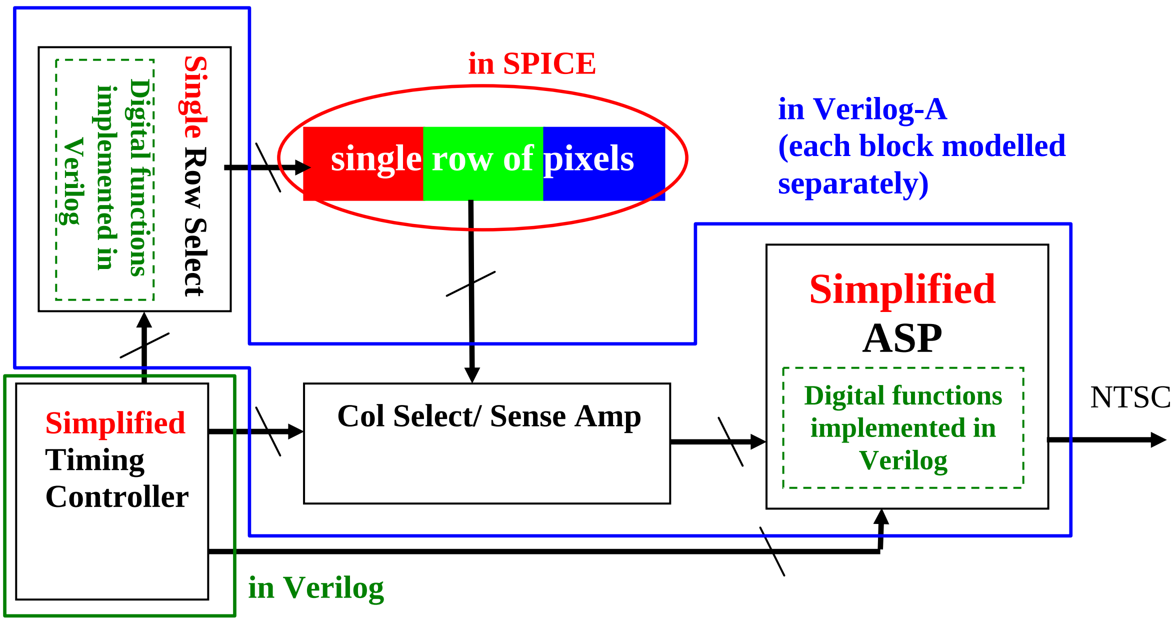

Figure 2: Language selection for SLV simulation: SPICE/Verilog/Verilog-A

For the SLV simulation, the desired outcome was an analysis of the interaction of specific parts of the circuit, modeled accurately, with the system, being modeled with Verilog or Verilog-A. The pixel array was modeled with transistor-level SPICE models since this gave the most accurate pixel representation. Charge injection in the pixel and its affect on the ASP was the focus of the simulation. The column sense amplifier and the ASP were modeled with Verilog-A to reduce simulation time. In this simulation, as shown in **Figure 2**, all of the major circuit blocks were modeled in Verilog-A separately so that any block could be replaced by a transistor level model to show how that circuit operated in the signal processing chain. Therefore, accuracy in the models and their placement in the design hierarchy were maintained. The simulation could be simplified further by modeling the entire ASP with a Verilog-A model. For some high level simulations, this might be prudent. Within the ASP, there were several digital functions implemented that normally would have to be done in Verilog-A. Though Verilog-A can handle digital functions better than

SPICE, in terms of simulation processing time, Verilog is the best choice for fast simulation time for logic circuits. The true mixed-signal and mixed-language capability of SMASH allows both Verilog-A, SPICE and Verilog models to be instantiated within a single SPICE netlist through SPICE sub-circuit calls, which allows for all of the logic functions within the Row Select and ASP to be implemented in Verilog. The Verilog and Verilog-A codes are included with `.INCLUDE` or `.LIB` directives in the `.PAT` file (pattern file of SMASH, which is associated with the netlist file `.NSX`, and contains the simulation environment such as the stimuli and the directives/commands for simulation settings) This also makes it possible to netlist the macro-model simulation from the same schematics used for the actual transistor-level design without worrying about partitioning the design into logic and analog portions or creating new schematics in a new tool.

Though the simulation time is improved greatly while maintaining a desired level of accuracy, FCS is not practical and a higher level of abstraction is needed.

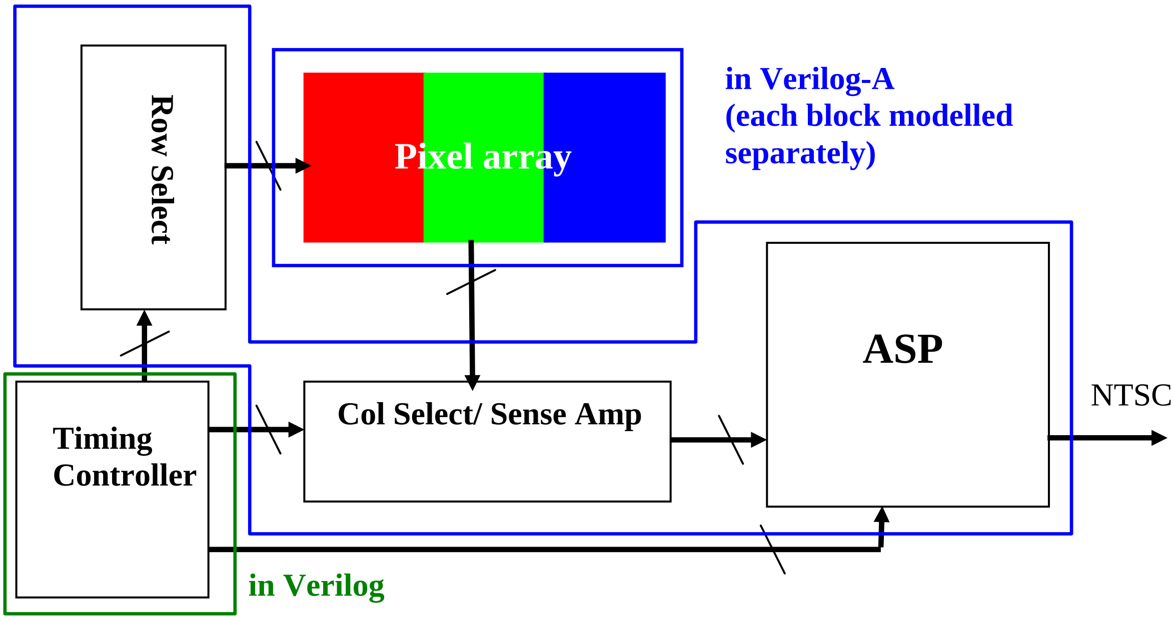

Figure 3: Language selection for FCS simulation: Verilog and Verilog-A

With a CMOS imager, the majority of the circuit operation is exercised within the line time and FCS are not necessarily helpful for verification of the ASP. However, the interaction of the Timing Controller with the Row Select and line time circuitry is helpful to check. Issues in image presentation can be resolved in simulation and more easily located since you can look anywhere in a circuit with a simulation. Another helpful FCS would be the AE (Automatic Exposure control) and AWB (Automatic White Balance). These operations require a feedback loop between the output of the ASP and the Timing Controller often on a frame-by-frame time scale which is 16.6 ms per field and 33.3 ms per frame for the NTSC standard while still requiring fractions of microseconds for the ASP. This is inherently a greedy consumer of CPU time, but allows for vital information that can save much time in software development and evaluation time. Clearly to perform a simulation of this magnitude, a Verilog-A model is needed to model both the pixel array and the ASP. The Timing Controller would be modeled in Verilog. SPICE models would generally not be useful in such a simulation. The SMASH™ / SIMULINK simulation interface can also be helpful to allow real image date to be input into the SMASH simulation and then exported back to MATLAB for viewing. For simplicity, the image data can be input into SMASH™ with text files and exported to MATLAB with the Text Dump option. This option in SMASH™ is very useful when using MATLAB for post processing of simulation data mainly because the format and interval can be controlled thus reducing the work needed in MATLAB to achieve a result.

4. Simulation Results

The schematic for the SLV simulation is given in **Figure 4**. As explained earlier, several levels of abstraction could be implemented depending on the desired outcome of the simulation.

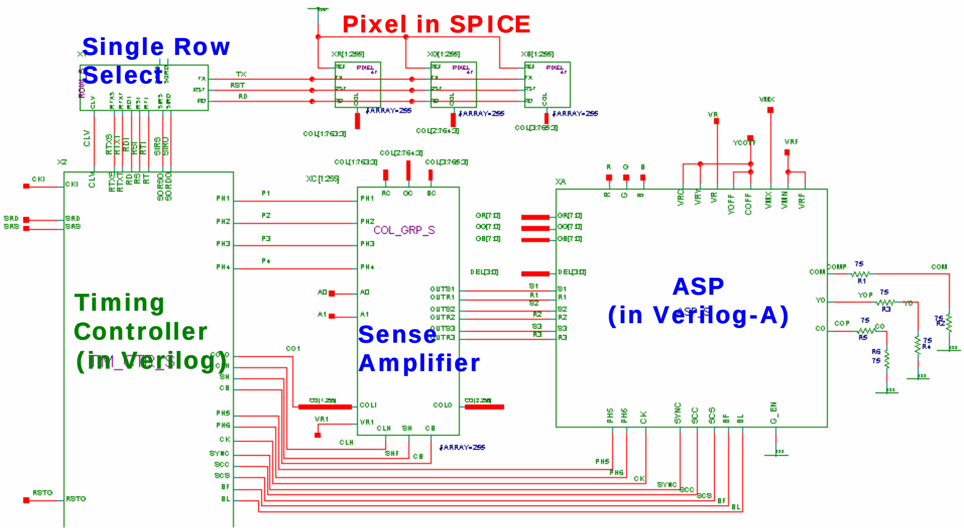

Figure 4: The SLV simulation schematic.

In this simulation, the Timing Controller is simplified for the SLV readout and is modeled with Verilog. The other processing circuits are modeled with Verilog-A, while the pixel is modeled in SPICE.

The SMASH™ waveform viewer allows for easy display of any necessary signals, as seen in **Figure 5**. Some of the benefits to the design effort by performing this simulation were:

1) Clock phase in the sense amplifier was confirmed
2) A sub carrier timing issue was resolved
3) Pixel and NTSC timing alignment was verified
4) Charge injection effect on sensing scheme was determined

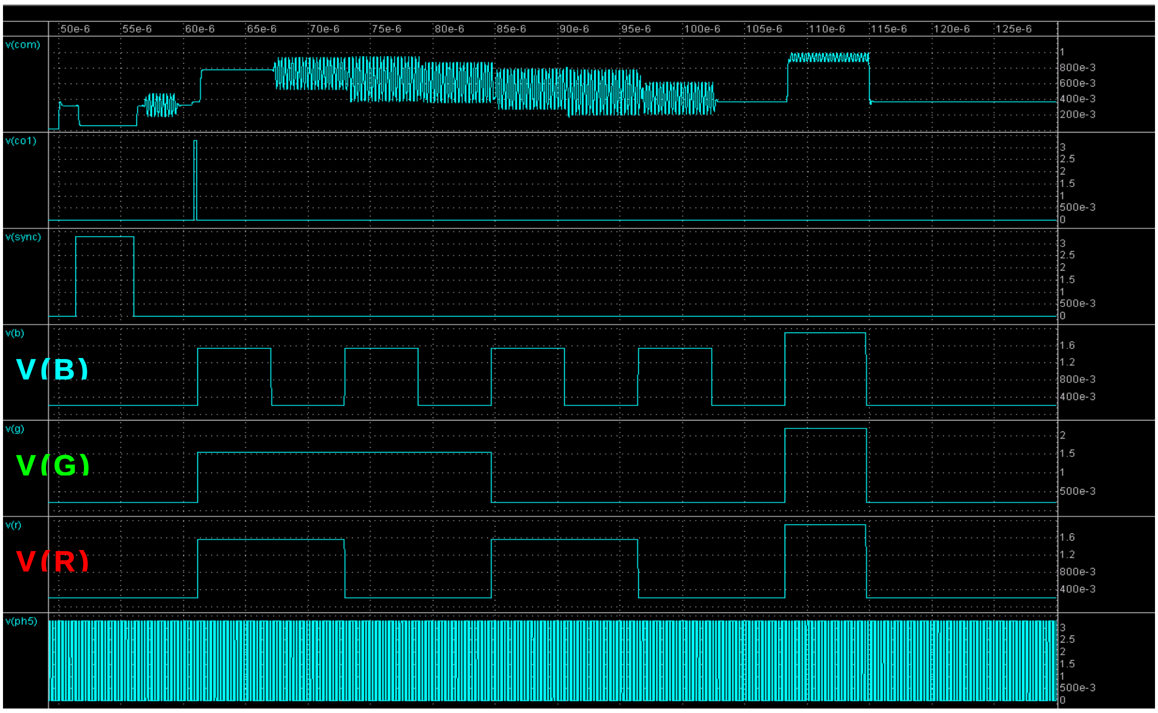

Figure 5: Results form the SLV simulations. The NTSC signals (at top) are easily displayed with any other necessary signals for fast debug.

As stated earlier, clearly time is saved by being able to perform such a complex simulation and debug various parts of the circuit using Verilog-A, Verilog, and SPICE models, for more accuracy. As can be seen from **Figure 5**, the signals are easily displayed and can be moved for timing comparisons of digital inputs and their corresponding analog outputs. Moreover, a myriad of post processing functions are available in SMASH™ and controlled data dumps allow for further processing in other post processing programs such as MATLAB.

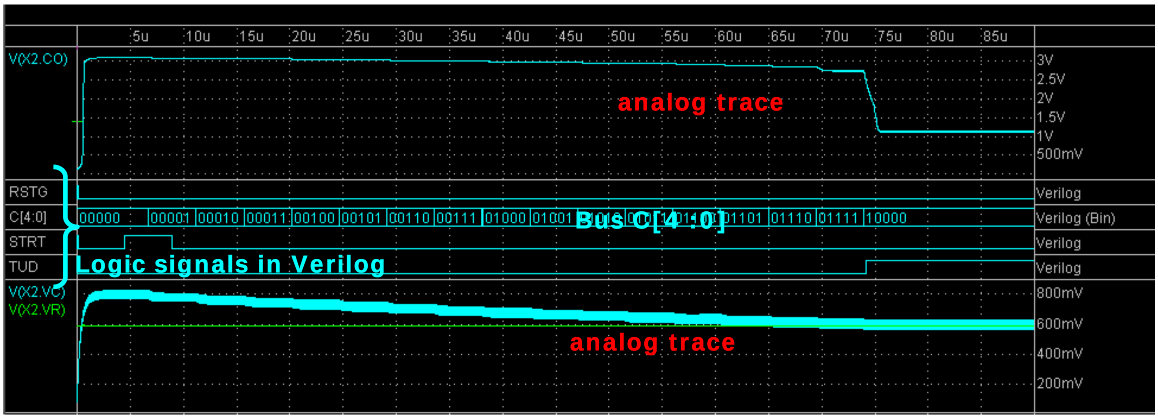

Figure 6: Simulation of a Sub-set of the FCS showing the ease of displaying mixed-signal simulations.

Figure 6 displays the results of a sub-set simulation of the SLV simulation. The simulation contained a Verilog model for the logic portion and a SPICE model for the analog portion. The analog and logic functions are shown in the same display window which greatly helps with the timing interactions.

One extremely helpful feature of SMASH™ is the use of bus notation in SPICE sub-circuits. For example, all of the Verilog code was implemented at RTL level. These files were included as libraries in the *.PAT file. The logic and register blocks were then netlisted as sub-circuits within the SPICE level circuits. The bus, C[4:0], in **Figure 6** was an interface between analog and logic circuits and was linked with SPICE sub-circuits. Normally in SPICE and Verilog-A viewers each element of the bus would need to be displayed making the presentation of the data more complicated. Here a Verilog bus notation can be used to simplify the display of data to show the logic value at which the analog signal toggles. This is also very useful for bus inputs in RTL netlisted in SPICE sub-circuits.

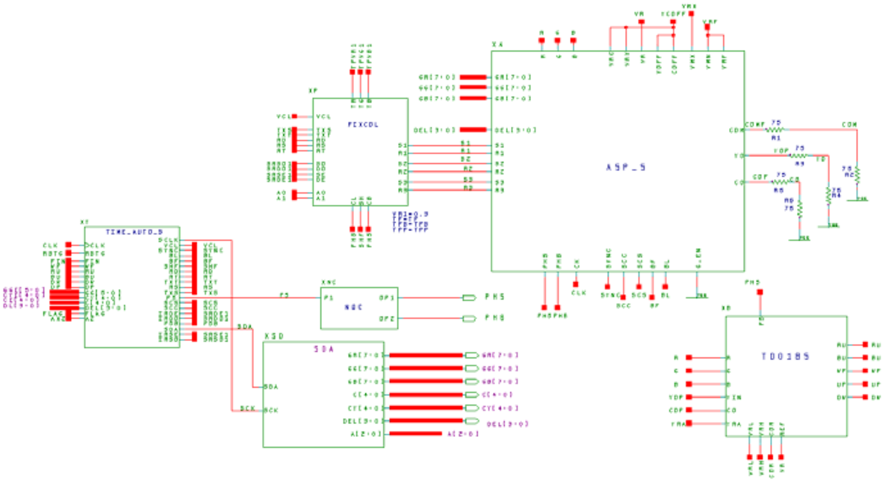

Figure 7: Schematic of the FCS. The Analog portion is simplified with Verilog-A code while the digital portions are implemented with Verilog code.

The FCS schematic is shown in **Figure 7**. The array and ASP were greatly simplified with Verilog-A code in order to speed up this particular simulation result. Speed was of the greatest concern in this simulation since the end goal was to have imaged data displayed in MATLAB for debug of possible artifacts due to timing. The simulation results are shown in **Figure 8**. When one looks at **Figure 8**, not much information can be obtained. This output is showing the RGB data and timing signals for an entire frame of NTSC video.

Figure 8: Simulation results of the FCS. The display of both digital and analog signals can be viewed simultaneously.

By exporting the RGB data to MATLAB and performing some post processing, the data can be viewed as an image as shown in **Figure 9**. As stated earlier, the text dump function allows for ease in post processing in MATLAB.

One might ask the question, "Where should I zoom in on **Figure 8**?" **Figure 9** holds the key. The input to the array was set up to be a toggle between black and white at the pixel rate. From **Figure 9**, it is clear that there are some problems with this toggle. In addition, the black region at the right of the image was incorrect. The correspondence between the image and the output data is easily understood and allows the designer to focus on those portions of the simulation results to understand why the data is not toggling between black and white as expected and the cause of the black columns. The design may be then modified to get the correct image. Normally one would have to have silicon and display apparatus to perform such debug techniques in image processing. Also, it would be difficult and time-consuming to perform detailed probing of the circuit to find where the issue is being generated. Potentially several weeks of lab time can be saved by solving the problem in simulation. With a true mixed-mode simulator with macro modeling capability, this information can be found in a few minutes to a few hours depending on the level of abstraction used.

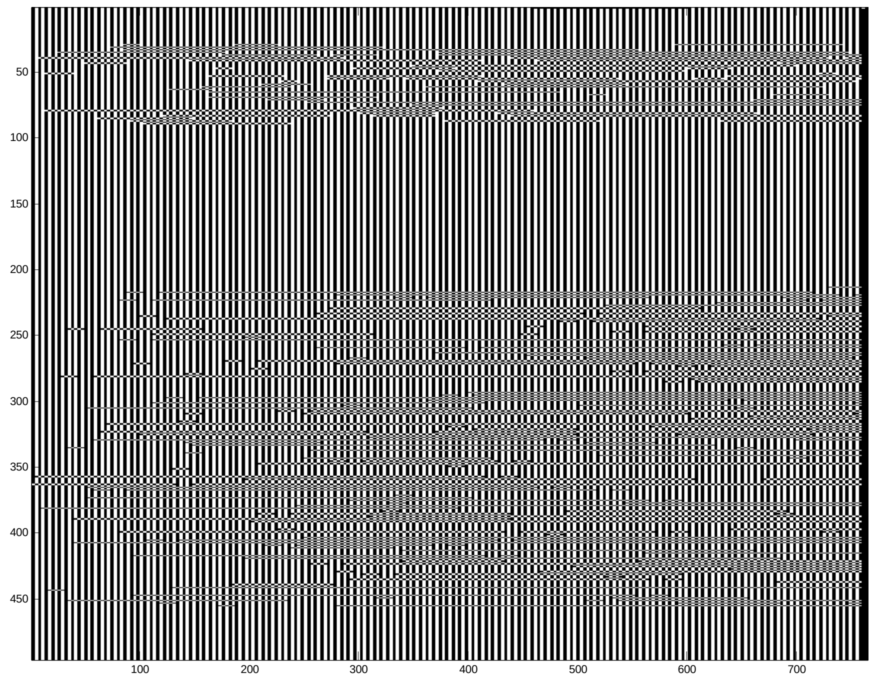

Figure 9: Image data from the FCS SMASH data displayed in MATLAB.

5. Conclusion

SMASH™ has shown great advantages as a mixed-signal, mixed-language and multi-level simulator helping to solve complex analog / logic circuit problems in simulation. The use of Verilog, Verilog-A, and SPICE in several levels of abstraction were demonstrated with circuit simulations for an image sensor project.
We have seen that the same set of schematics can be used for higher level simulations reducing time and reducing errors. SMASH™ allows for efficient display of both analog and logic signals in a single viewer and allows the user to easily export the data for further post processing.