A Sophisticated Memory Test Engine for LCD Display Drivers

Oliver Spang[1], Hans-Martin von Staudt[2], Michael G. Wahl[1]
[1] University of Siegen, Computer Science, Siegen, Germany
[2] Dialog Semiconductor, Nabern, Germany

Abstract

Economic testing of small devices like LCD drivers is a real challenge. In this paper we describe an approach where a production tester is extended by a memory test engine (MTE). This MTE, which consists of hardware and software components allows testing the LCD driver memory at speed, allowing at the same time the concurrent execution of other tests. It is fully integrated into the tester. The MTE leads to a significant increase of memory test quality and at the same time to a significant reduction of the test time. The test time reduction that was achieved by executing the memory test in parallel to other analog tests lead to the test cost reduction, which was the impetus for developing the MTE.

1 Introduction

LCD drivers are devices that on one hand are controlled by the video controller and on the other hand are directly linked to the lines and columns of the LCD itself. They are needed in great numbers for all LCD displays. The LCD drivers discussed here were designed for handheld devices with displays up to 65,000 colors. Low power consumption is a must. They consist of the control logic that takes care of interfacing to the system, and a RAM that contains the color values for the connected lines of the display matrix, and the divers for the LCD cells.

The test of tiny and valuable semiconductor devices like mobile LCD drivers are an economic challenge to deal with. For highly complex chips a 'big iron' tester is well accepted and the tester manufacturers provide testers matching all specifications. For a mass product like the LCD drivers with a small gain margin the constraints are quite different. Test costs are the primary target of this effort, but of course the minimization of the test cost must not reduce the test quality.

Improving the test quality and reducing the test time is usually the impetus for implementing memory BIST. This is for sure the correct strategic approach. On the other hand, memory BIST increases the silicon area and possibly the pin count, too. Both factors drive the cost beyond the cost critical point. Thus, the necessity of improving memory test efficiency as well as quality leads to the extension of the existing production test system. The standard way of extending the tester is buying a commercially available add-on. This approach would solve the problems, but it also has two main drawbacks: high investment cost and a tighter binding to a specific tester.

The alternative, which we describe here, is building a specific memory test engine that matches the technical requirements and allows the parallelization of the memory test as well as other tests in the same way as the commercial test extension, but at significantly lower cost. The key requirements are: support of all standard test algorithms, evaluation of the results, and repair of defect memory lines.

Some further constraints influencing the design of the MTE were limitations in space and power consumption. The MTE had to fit into the remaining space of the test head and for supplying the MTE only the standard power supplies of the ATE were available. A further constraint was the ability to disconect the MTE from the device under test (DUT) for the contingency test.

2 Testing

Testing the LCD drivers consists of four steps, which are more or less independent from each other. These are: DC test, analog test, digital test, and memory test. In this paper we focus on the aspect of executing the memory test in parallel to other tests. This requires additional hardware/software, which is described in the rest of the paper.

For testing memories different test algorithms are used, starting with basic algorithms like checkerboard and ending with the well known march tests. For a detailed description of the tests and their properties please refer to [5]. All memory tests are performed using only very few basic operations: set address, increment address, decrement address, write '1', write '0', read '1', and read '0'.

978-3-9810801-2-4/DATE07 © 2007 EDAA

When implementing a test, it is always necessary to be aware of the physical layout of the memory, because the tests refer to the physical neighborhood of the memory cells, which is not necessarily the logical neighborhood. This effect prohibits simply using a memory address increment/decrement function.

Usually the test algorithms are programmed in the ATE and then executed. Using the MTE, the tests are executed by the MTE. For performing the tests, the MTE must be programmed accordingly. The MTE requires a specification of the type of test that shall be executed. This is not a complete test description, because all standard tests are predefined in the firmware of the ATE and only configured. The mapping of the logical addresses to the physical addresses is required, too. Based on the chosen algorithm and the mapping of the logical bits to the physical bits the test sequence is calculated as described below.

The LCD drivers support different addressing schemes. All of them allow direct addressing of a memory row, although the address is applied in six subsequent cycles, followed by a data write or data read. This is much more complex like e.g. accessing an SRAM. Some designs support an auto increment and/or auto decrement mode, thus reducing the data read/write to one cycle. These device specific properties must be taken into account during the setup of the test, too.

3 MTE hardware

As lined out, the MTE had to be integrated into the existing test concept. This was a very strict requirement because the current tests had to be ported to the MTE for guaranteeing the continuity od the test quality, before the improvements of the MTE could come into play. It was also necessary to disconnect the MTE from the DUT for contingency tests. These requirements lead to the concept of attaching the MTE as a side controller, linked to the connection of tester and DUT.

A closer look at the hardware shows that the separation is not so strict. The link from the MTE to the ATE-DUT connection is implemented using switches. The reason for this is that at the beginning of the DUT test a contingency test is necessary. Because the MTE and the ATE are connected in parallel to the DUT, the MTE must be disconnected from the DUT during this time.

Finding a solution for the switches took some time. Relays would have had the advantage of a galvanic separation and a high signal quality, but the limitation in space and the comparatively high switching current ruled this option out. The limited reliability of relays is also an issue, because the number of relays on a DUT board for testing four DUTs in parallel is quite high. This limited the choice to analog switches.

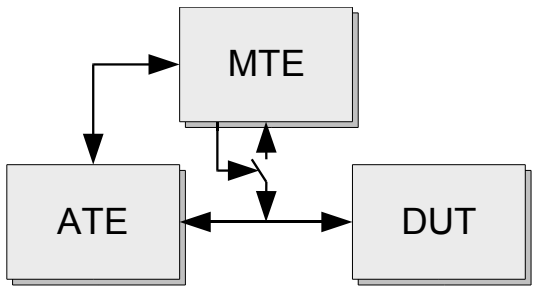

Figure 1: Test hardware configuration

The selection of the switches was much more cumbersome than expected. The selection criteria were a low on resistance combined with a small capacity. The selected analogue switches fulfilled the requirements well. In particular, the signal quality was not affected in such a way that the test frequency hat to be reduced.

It is also possible to control some pins of the DUT from the ATE, while the MTE executes a memory test using other pins. This is the precondition for being able of scheduling tests running in parallel.

The hardware of the MTE consists of standard components, as shown in figure 2. For the communication with the ATE and the setup of the FPGA a microcontroller from ATMEL is used. As communication path between the ATE and the MTE some standard tester pins are used. The actual memory tester is realized in an FPGA. The FPGA includes a microcode driven state machine, which is programmed by the ATMEL microcontroller. The captured data is stored in the SRAM connected directly to the FPGA. The microcontroller does not only handle the communication with the ATE. It also analyzes the test results stored in the SRAM by the FPGA.

For the power supply of the MTE the supplies provided by the ATE had to be used. Using an external power supply was impossible due to the limited number of connectors to the DUT board. Of course, the supply voltages provided by the ATE were not those required by the MTE hardware. Thus a power supply was implemented on the MTE. For generating the 5V, a step down converter was used. This converter had to be tailored towards low RF disturbance, limited height, and low temperature, which made the selection of the

coil quite difficult. Several coils hat to be tested before we could make a final decision. For limiting the RF emissions, a shielding layer was implemented in the PCB and an external housing was implemented. The other voltages were derived from the 5V using standard serial controllers.

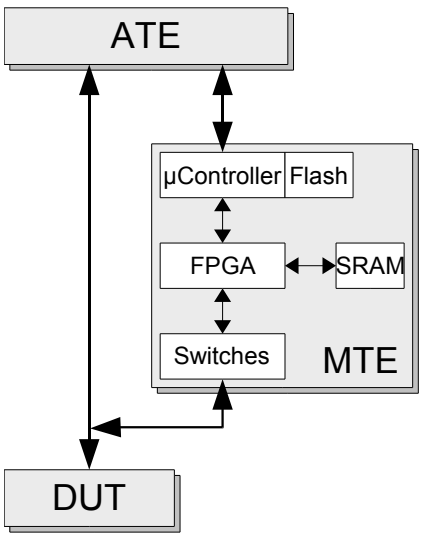

Figure 2: MTE Hardware

Unfortunately, the ATE supported power supplies cannot drive a high current. This influenced the complete design of the MTE. Low power always was a target, not only with the decision of using relays or analog switches. This consequent low power design finally lead to consumption that could be handled easily by the ATE supplies. Due to the efficient design, the overall power consumption could be reduced to approximately 5W, which is significantly lower than the limit given by the ATE.

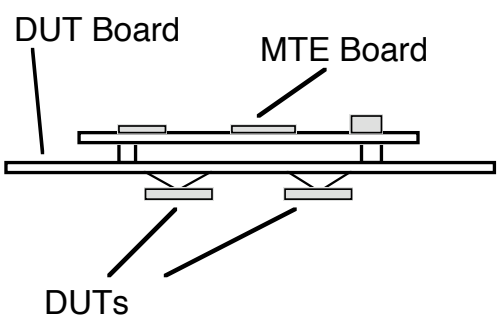

Figure 3: Linking the load board and the MTE

The complete MTE hardware is implemented on a circular PCB which is set on top of the load board. For using the MTE, a special load board had to be manufactured that carried the connectors for the MTE. Figure 4 shows the MTE.

4 Test software

The MTE is designed as an autonomous system. It consists primarily of the microcontroller and the FPGA. As stated above, the microcontroller takes care of the communication with the ATE. It receives coded information about the DUT, in particular the memory size, physical memory configuration, pin configuration, and the selected test algorithm. The microcontroller knows a set of predefined test algorithms that are then tailored according to the DUT specific parameters. The predefined test algorithms are stored in the flash memory of the microcontroller.

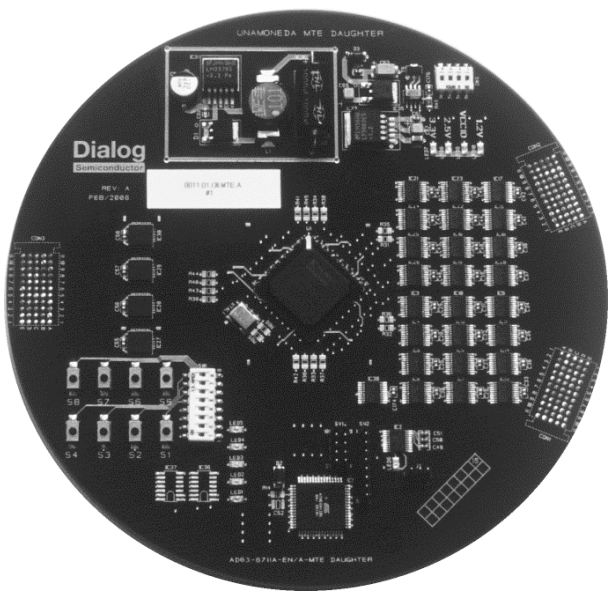

Figure 4: Picture of the MTE

In case that a test algorithm is required that is not one of the predefined ones, the appropriate microcode can be sent to the microcontroller using a special command set. This does neither require an update of the microcontroller program nor a reprogramming of it. This concept allows a quick ramp up using the predefined algorithms as well as a high flexibility by extending the set of algorithms if necessary.

This concept allows ample flexibility by assembling the memory test according to the requirements of the DUT. The final step would be modifying the FPGA structure, which provides all degrees of freedom a software developer can expect, but requires of course the highest degree of knowledge.

The software of the microcontroller itself is located in the internal flash memory of the microcontroller. The operating system of the microcontroller can be upgraded easily using an update routine. Greater updates can be performed for both, the microcontroller and the FPGA, using their JTAG interfaces.

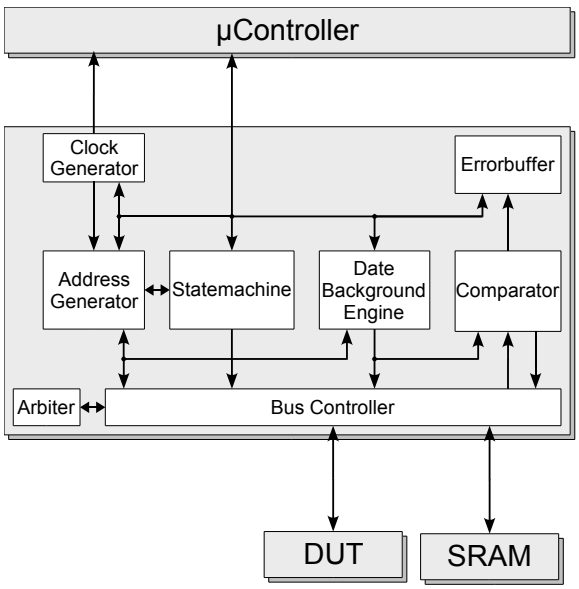

Figure 5: Internal structure of the FPGA

The default communication between the ATE and the MTE is realized by using an I²C interface bus built into the microcontroller. Due to the limited number of ATE pins available for additional tasks, the communication to the MTE had to be implemented as a serial connection. From the microcontrollers side an I^2C bus is available. Although this is a convenient way of transferring the commands to the MTE, it is by far too slow for transferring test results, in particular if a complete memory map is sent back to the tester for detailed analysis. This would take away the advantages in test time gained by using the MTE. For such an amount of information the data transferred to the ATE will be sent over the tester channels that usually connect the DUT to the ATE, having the DUT disabled.

For speed reasons, the test itself is executed in an FPGA. The communication between the microcontroller and the FPGA consists on the download side primarily of a set of microinstructions that lead to the execution of the test algorithms using the selected DUT configuration.

A test using the MTE is executed as follows: First, the ATE sends the instructions describing the test to the MTE. The microcontroller then generates the necessary codes for the state machine which is implemented in the FPGA, the address generation module and the test data generation module. The codes are then transferred into the program memories (PM) of the modules. The modules themselves are defined independently with well specified interfaces for easy maintenance. For each module several status registers (SR) are available for storing parameters and results.

The address generator (AG) produces the next address according to the signaling from the core state machine. This can simply mean that the next address is generated with the next clock cycle. According to the memory mapping defined in the SR the next address is generated. If the DUT supports the feature of autoincrement / autodecrement of the address, this feature can be used by the test algorithm if appropriate. Otherwise the address is taken from the AG module.

The data generator (DG) calculates the correct data word according to the selected test algorithm. The algorithm is specified through the SR of the data generator.

If the test algorithm requires a read after write of a memory cell or row, then the data read is compared to the generated data. If a march test is used where in one run the memory bits are set and in a second run the bits are read, then the written data is generated again and compared to the measured values. For the evaluation of the test results a comparator is used.

In case of an error, the fault location and the fault value are stored in the error buffer inside the FPGA. This error buffer is quite small, compared to the size of the memory. In addition, all data read is stored in the so called picture buffer. The picture buffer is implemented in the SRAM. At the end of a test run the microcontroller analyzes the sampled data from the error buffer. After finishing the analysis, the MTE signals the completion of the test to the ATE. For engineering purposes the error data can be read out of the picture buffer on demand.

Based on the analysis, the MTE can also perform a memory repair action, if the DUT is equipped with appropriate spare rows.

Because the MTE supports testing up to four DUTs in parallel, there are four error buffers and the SRAM is large enough carrying at least four complete maps of the DUT memories in the picture buffers.

The state machine in the FPGA controls the actions of the AG, the DG and the bus controller (BC). The BC provides the signals to the DUT and for the communication with the ATE over the device bus and it accepts the appropriate control signals.

5 Test scheduling

Test scheduling is an important issue concerning cost. Partly, there is an obvious schedule, because first you need to know if the die works at all, and then you get to more detailed tests. In this second phase the sequence of tests is not necessarily fixed, which allows designing different schedules with different priorities. The test sequence for the LCD driver chips is divided into subgroups of related test functions:

DC	contingency test
	leakage
	stand by current
Analog	charge pump
	voltage reference
	DAC test
Digital test	I/O
	Scan
Memory test	

Table1: Test sections

Each of the tests is mostly independent of the other tests. The tests themselves have mostly the same structure: Vectors for setting the test up, execution of the test and analysis of the results. For being able to optimize a test sequence two effects can be used: (1) stopping the test after the first failure and (2) parallel execution of different tests. The first approach is not useful here because four dies are tested in parallel, and time can only be saved if all four devices fail in an early test. In the case of the LCD driver chips, DC, digital and memory test cause high activities on the data bus, whereas the analog test does not use the data bus continuously.

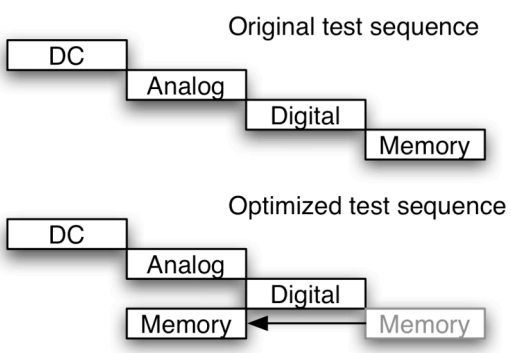

Figure 6: Test schedules

The basic impetus for designing the MTE was the idea to run several groups of tests in parallel to reduce the test time. An analysis of the requirements of the four main test groups showed that the memory test can be executed in parallel to the analog tests, because there are no competing accesses between these two groups on the device bus. Running the memory test in parallel to the analog test de facto saves the time for memory test, which is significant.

6 Application results

Based on the fact that the MTE is designed saving test time and therefore saving test costs, the results are very promising. First experiments showed that the MTE is fully functional and that the test could be executed in parallel as planned. Nevertheless, the path to this point had been cumbersome due to some undocumented features of components used for the MTE.

Although in these experiments the MTE could not be used to its full extent, the benefit was clearly visible. At the beginning of a test run it took a while to initiate the MTE and sending the commands from the ATE to it. When executing the memory test in parallel to the analog test, the MTE could save several 100 milliseconds at every test run, depending on the applied test algorithm and the device under test. For a whole wafer a benefit of many minutes could be achieved.

Another significant profit is the detailed information on the tested LCD driver memory by reading the generated map into the picture memory. Using more sophisticated engineering tools, the picture RAM was read by the ATE and the content was used for detailed failure analysis. This feature allows identifying critical layouts or process uniformity in the memory design.

7 Summary

The MTE as described here was used in a production environment. It could be shown that a parallel operation of two different tests on the same test device is possible and results in a gain of test time.

For the present, the results of an accelerated test flow are just the beginning. Future developments with the support for special addressing functions for acceleration of the communication with the DUT and highly matched test algorithms will lower the test time even more.

The next step is bringing the MTE from the prototype state to a generally used extension, which is theoretically quite simple, but still challenging.

8 References

[1] Boutobza, S.; Nicolaidis, M.; Lamata, K.; Costa, A.: *"Programmable Memory BIST"*. ITC 2005, Austin,TX

[2] Du, X.; Mukherjee, N.; Cheng, W-T.: *"Full -Speed Field-Programmable Memory BIST Controller"*. ITC 2005, Austin,TX,

[3] Kim, H.C.; Jun, H.-S.; Gu, X.; Chung, S.S.: *"At-Speed Interconnect Test and Diagnosis of External Memories on a System"*. ITC 2004, Charlotte, NC

[4] Schulz, Holger; Wahl, Michael: *"Doubling memory tester capability by efficient expansion of hardware and software"*. ETW 2003, Maastricht, NL

[5] Van de Goor, A.: *"Testing semiconductor memories - Theory and Practice"*. Wiley, Chichester, 1991

[6] Yamasaki, K.; et al: *"External Memory BIST for System-in-Package"*. ITC 2005, Austin,TX