R400 IKOS Status

LATEST STATUS	**Diagnostics on Netlist 4.1: Many failed test cases now passed on the latest patched netlist** **BIOS on Netlist 4.1: Successfully boot BIOS from ROM, but some VGA modes working while some others not working - under investigation**
DATE (MM/DD/YYYY)	**DETAIL STATUS**
Thu 01/02/2003	Netlist 3.0 released
Sat 01/11/2003	**[16:41 Ron White]** We have the reset block working correctly now. The target PC boots and the R400 chip is responding to configuration reads and writes. Operations to non-pci configuration registers fail. We are investigating the problem.
Wed 01/15/2003	**[15:23 Ron White]** Thanks to Gabe, we found another mistake in the Star memory models we created for the R400. The models are fixed and we are recompiling. We'll test it after supper. **[22:42 Ron White]** At least some of the R400 registers are working in the latest IKOS compile. (BIOS scratch and VGA) We will test the registers more extensively tomorrow morning. The next step is to get the memory interface to work. Is it worth having a BIOS guy start his work on a second R400 setup if the memory is not working yet ?
Thu 01/16/2003	**[10:48 Gabriel Abarca]** Frank Hering was just suggesting Ron that the best way of getting an operational MC and memory pads for IKOS was to get the RTL we use for regression and compile in into IKOS. Ron was saying he would like to try the idea, as this would give him both things in one shot. Please correct me if I am wrong, but I believe the most stable path to get the good MC and pad RTL models is: /proj/crayola_vol3_nobackup/run_regress/crayola_sim_chip_247_linux_rtl/devel
Fri 01/17/2003	**[10:00 Liang Shen]** Jae/James started playing with IKOS (register access) to prepare for BIOS bringup and found VGA is not enabled in the current setting. Ron's team is going to turn VGA on shortly.
Mon 01/20/2003	**[11:52 Ron White]** We have a VGA enabled R400 database that can be used for beginning the R400 BIOS work. Damon or I can demo the database whenever it is convenient. **[15:00 James Huang]** Continue verifying register access and noticed some are read/write ok, some are not.
Tue 01/21/2003	**[12:26 Gabriel Abarca]** The intention of this e-mail is to have a script for IKOS to initialize MC-MH. I cut and pasted the init sequence from our chip level tests, took out comments and clock related sequences and put it in the document below. I highlighted in bold blue the register writes I think are needed. When registers were written several times, I only highlighted the last time, assuming only one write is needed. Is that right? I would appreciate if you can review this and tell us is something is missing or is not needed.
Wed 01/22/2003	**[12:39 Jae Chong]** I have done the following test before IKOS had to shut down for wiring of the lab. VGA was enabled. The following VGA related registers work properly with proper default values and read/write VGA_MEM_WRITE_PAGE_ADDR(0x38) VGA_MEM_READ_PAGE_ADDR(0x3C) VGA_SEQUENCER_RESET_CONTROL(0x304) VGA_MODE_CONTROL(0x308) VGA_SURFACE_PITCH_SELECT(0x30c) VGA_MEMORY_BASE_ADDRESS(0x310) VGA_DISPBUF1_SURFACE_ADDR(0x318) VGA_DISPBUF2_SURFACE_ADDR(0x320) VGA_HDP_CONTROL(0x328) All BIOS scratch registers work fine initially, but sometimes stop working after some other register read/writes. The following registers do not work properly: BUS_CNTL(0x30) BUS_CNTL1(0x34) CONFIG_CNTL(0xe0)

	RBBM_CNTL(0xec) RBBM_SOFT_RESET(0xf0) RBBM_SKEW_CNTL(0xf4) CONFIG_MEMSIZE(0xf8) VGA_RENDER_CONTROL(0x300) MM_INDEX write works but read does not work BUS_CNTL, CONFIG_CNTL are some of the registers that must be working before anything else.
Fri 01/24/2003	**[12:38 Ron White]** The R400 Bus control register problem is fixed. The bug was due to a bad rom controller stub model that broke the R400 readback chain. We have another database that is ready to be used for the BIOS work. It's time for a more exhaustive R400 register test. The new RTL memory controller database will be tried after lunch. **[14:58 Ron White]** The RTL memory controller IKOS database is not working in-circuit. Memory controller register ops are working but there is no activity on the RAS/CAS lines after initializing the memory controller. In addition, there seems to be a naming mismatch between the RTL MC pads and the RTL MC in the netlist. We are investigating further.
Wed 01/29/2003	**[16:33 Ron White]** The R400 memory controller is alive and Bei is checking the waveform timing. Once she is satisfied with the timing and tells us how to change the memory init script, we will plug in our RAM chips and see if they work.
Tue 02/04/2003	**[13:49 Ron White]** Colin and Damon had to compile around some broken hardware yesterday. They will be trying the RAMs at the end of today.
Thu 02/06/2003	**[11:21 Gina Seto]** I spoke to Brian Leblanc in marl. He tells me that he will check in the final version of the register test today before lunch. I will probably take some time this afternoon and try simulating it on the emulator. Then it should be ready to go for IKOS. The register test tests specific blocks - so those blocks that IKOS is support can be tested individually. **[11:34 Colin Stewart]** We have some good news. We have a working database where the memory controller is almost working. This is with real DRAM chips attached and using the script that Bei sent us. Memory writes work and the reads are almost working. The problem with the reads is that the data strobes are not delayed properly. This is a problem that we have seen before in previous chips and know how to work around. We are working on a compile and hope to test something tomorrow.
Fri 02/07/2003	**[13:29 Colin Stewart]** The first memory channel seems to be working properly now. Channel M0 seems to be working properly but channel M1 has some bits that are failing. We are investigating this to see if it is something wrong with our target board. Channel M2 and M3 are not currently connected.
Mon 02/17/2003	**[16:15 Ron White]** We have debugged the new in-circuit target card (2 problems) and we now have a single channel of R400 memory working in the IKOS lab. We have to recompile tonight but we (optimistically) expect to have the hardware for 4 memory channels available tomorrow around noon.
Wed 02/19/2003	**[13:26 Ron White]** We have 4 channels working now but we also have minor refresh problem. Brian LeBlanc is using the setup this afternoon. We'll debug the refresh issue tomorrow.
Thu 02/20/2003	**[11:52 Colin Stewart]** We have increased the speed of the system and we now have 4Mb of memory passing the refresh rate test. Since the test takes a while we will try testing more memory later but we have more confidence that the memory is working well. We are still using the ikos box for one more small experiment and will be releasing it soon.
Tue 02/25/2003	**[08:15 Brian Leblanc]** I ran the register tests overnight on the partial Ikos net list. There are still problems in the diagnostic itself, but I do believe that the values reported in the following logs (log file folder) are valid. The CP block test caused a hang, this is not very bad considering, the block is not entirely populated in this net list. It is important to remember that the test is based on the current register specification, and the net list is based on the register specification as it was from December 23. There have been many changes to registers in that time. Due to problems with the tools that extract the register information for the diagnostics, it is no possible to get the correct register information to match this net list. This will hopefully not be an issue on future net lists. **[17:39 James Huang]** We tried followings on IKOS 30M today: - Video memory R/W through MM INDEX verified. - Created a VGA mode preset script file, all VGA surface/GRPH/Control registers are set correctly after running the script. - Run VGA mode set core, the system hangs up. - Access VRAM through VGA aperture, R/W seems not working. Tomorrow's work:

	-Find out why VGA mode set hangs. -Find out why VGA aperture is not accessible. -Go through the VGA preset sequence with Gabriel to see if there is anything missing or wrong there.
Wed 02/26/2003	**[08:57 Brian Leblanc]** I have spent the evening debugging the diagnostics environment. I have found some inconsistencies between the emulator and ikos paths that are taken in the diagnostics. There are still several so for unexplained issue I will have to debug. **[16:56 Gina Seto]** Before running a diag display test, I needed to ensure that we were reading/writing to the DCP block registers properly. Since the r4 register test results was showing a large number of registers in the DCP not passing, I examined some of those registers individually using db128. For example, I tried writing to D1GRPH_PRIMARY_SURFACE_ADDRESS: - aperbase=e0120000 - #=0 - mmr 1844 Read MM Reg 0_1844 (e0126110): 00000000 - mmw 1844 12345678 Write MM Reg 0_1844 (e0126110): 12345678 - mmr 1844 Read MM Reg 0_1844 (e0126110): 00000000 The write did not work. Gabriel and Damon just came by to capture a waveform of the memory write, and Gabriel will be looking at it. **[17:27 James Huang]** I still can't understand completely today what makes VGA mode set hang. But one problem has been known, after enable MC and VGA preset, accessing VRAM via VGA aperture will make it hang. I can repro that consistently by: 1. Power up IKOS 30M. 2. Run Bei_Fix5.db followed by PreVGA.db, which set both MC and VGA surface, HDP, control etc. 3. Enable VGA memory access from CPU. 4. Launch Debug, and type -d a000:0 command, system freezes right away. The problem has been reported and showed to Gabriel. Besides this problem, it seems to me that stack was over flown when executing VGA mode set, reason is unknown yet.
Thu 02/27/2003	**[12:41 James Huang]** -Can't read registers correctly after write to them, which confirms what Gina found yesterday. -In the current simulation environment, CPU register BP must be 0. Once it changes to non zero, Bios code will be screwed up right away. But it's not acceptable to us because Bios does use BP for many cases. **[18:34 Gabriel Abarca]** Stephen Bagshaw found that the reason of the register problem is that we are accessing registers before the display clock generator is programmed, so this defaults to using the crystal clock for display, and the crystal clock is not forced to run in IKOS. We do not want to ask for changes in the BIOS sequences, there should not be such restriction, so if this is not going to delay the bring up of the 4.0 netlist, we would like to modify the netlist 3.0 so that this chip level signal is also forced to PCI clock: IO_DC_xtalin **[19:09 Stephen Bagshaw]** To further comment on Gabriel's suggestions: Based on what I'm seen from some IKOS waveforms, it seems that since the pixel clocks for both display controllers are not running on IKOS, accesses to certain "special" registers that require data to be resynchronized from the pixel clock domain to the clock domain of the registers (core clock) cause the chip to get into a state where it would not accept further register operations (basically a register cycle complete signal (what we call a "ready" signal) never gets sent acknowledging the end of the register cycle for this "special" read or write operation. The R400 hardware that handles register cycles has some "timeout" circuitry that happens if this register cycle complete signal is not received. This is what is happening on IKOS when people see that register reads of a register that has just previously been written with a value always return a value of all 0s. The temporary workaround would be to make a small script that is run after the R400 chip boots on IKOS to program the pixel clocks of both display controllers to be running. As Gabriel mentioned below, in the current IKOS netlist, the pixel clocks default to be connected to the XTALIN input of the chip. In the current IKOS netlist, this XTALIN input of the chip is not connected to the clock. As a result, the resynchronization logic for these "special" register reads and register writes does not occur properly and the chip gets into a condition where it will timeout on any register cycles. The correct workaround, as Gabriel mentioned, is for Ron's IKOS group to connect the "IO_DC_xtalin" pin to a clock signal and recompile the R400 netlist 3.0 netlist. In the real chip this pin will always be hooked up to a clock oscillator or clock crystal. After this new netlist is compiled, there should be no need for the temporary workaround mentioned above.

Fri 02/28/2003	**[10:39 Minghua Zhu]** Only the following registers in CRTC need PCLK running in order to read correct value back without RBBMIF timeout. DxCRTC_STATUS_POSITION DxCRTC_STATUS_FRAME_COUNT DxCRTC_STATUS_VF_COUNT DxCRTC_STATUS_HV_COUNT DCP is running off SCLK and do not have this requirement. So DCP register like D1GRPH_PRIMARY_SURFACE_ADDRESS can be accessed correctly even when PCLK is off. Please try on these registers. **[10:59 James Huang]** I tried registers below, none of them can read back properly with "18Feb_r400_net3_VLE30M_5.03_..._debug.vmw". Some of them seem to work with "8Feb_r400_net3_VLE30M_5.03_..._ronus.vmw" though: VGA_SEQUENCER_RESET_CONTROL, VGA_RENDER_CONTROL, VGA_MEMORY_BASE_ADDRESS, VGA_SURFACE_0_BASE, VGA_DISPBUF2_SURFACE_ADDR, D1GRPH_PRIMARY_SURFACE_ADDRESS, D2GRPH_PRIMARY_SURFACE_ADDRESS, VGA_SURFACE_1_BASE, VGA_DISPBUF1_SURFACE_ADDR, D1GRPH_SECONDARY_SURFACE_ADDRESS, D2GRPH_SECONDARY_SURFACE_ADDRESS, VGA_SURFACE_0_SIZE, VGA_SURFACE_1_SIZE, VGA_SURFACE_0_INFO, VGA_SURFACE_1_INFO Also, all of those MC registers can't read back correctly after programming them with the current vmw, although the programming seems effective since we can access frame buffer after. On the other hand, all scratch registers function well. **[11:38 Damon Tohidi]** We have a second R400 setup available on VLE5M for the BIOS group. **[16:57 James Huang]** I just tried this VLE5M. Nor VGA registers neither ext. registers (except Bios Scratch register) are accessible. Scratch register and VRAM access seems fine. **[17:03 Colin Stewart]** The VLE5M and the VLE30M are based on the same netlist. **[17:13 James Huang]** I think so too. What I did was: launch Debug after running bei_fix5.db , and read 3C3, which should be the IO base address of the ASIC, but it gave me "FF" all the time. A few other VGA registers read same thing, while VGA is enabled by set "VGA Disable" to "Z". **[18:38 Gabriel Abarca]** The IKOS netlist fix for the register is being compiled over the weekend
Mon 03/03/2003	**[11:12 Colin Stewart]** The fixed database for the 30M with the IO_DC_xtalin clock tied to PCICLK, is ready. It is located: /net/.../02Mar_r400_net3_VLE30M_5.03_..._debug.vmw.
Tue 03/04/2003	**[14:28 Brian Leblanc]** I have been trying out the 5M and have had difficulties, even more so than on the 30M with all the register writes and reads. I have log files in file://ma_bleblanc/ikos/logs for runs each of the systems. The directories are labeled for which IKOS box and which net list was used, 3_0_0 referring to the net list without the clocks added in. **[18:11 Gabriel Abarca]** Colin is working on porting the fix he did in the 30M to the 5M. The display registers Gina and I tried a few moments ago, which were broken before, work now in 30M. So porting that to the 5M will help. Also there may be register issues in the 5M due to the blocks we stubbed. Colin is working on that too.
Wed 03/05/2003	**[13:52 Gina Seto]** No display yet but I was able to run the palette read/write test on IKOS. Both the test for the standard palette and the PWL palette passed. The details of the test are in the attached word file PaletteRWTest.doc - if you have any questions about the test please ask. In addition, I ran a quick sanity test on db128: I turned on autofill, checked that it completed, then read a few lut values. The values I checked read back correctly: - aperbase=e0120000 - #=0 - mmr 1928 Read MM Reg 0_1928 (e01264a0): 00000000 - mmw 1928 1 Write MM Reg 0_1928 (e01264a0): 00000001 - mmr 1928 Read MM Reg 0_1928 (e01264a0): 00000002 - mmr 1925

	Read MM Reg 0_1925 (e0126494): 00000000 - mmr1925 ^Error- Invalid command - mmr 1925 Read MM Reg 0_1925 (e0126494): 00401004 - mmr 1925 Read MM Reg 0_1925 (e0126494): 00802008 - mmr 1925 Read MM Reg 0_1925 (e0126494): 00c0300c - mmr 1925 Read MM Reg 0_1925 (e0126494): 01004010 - mmr 1925 Read MM Reg 0_1925 (e0126494): 01405014 Lastly, I tried running a basic display test (640x480, 8bpp) but the test fails and hangs. I need to investigate this more. **[17:40 James Huang]** Still trying to bring up VGA mode 3 on 30M. A lot of progress have been mode in last two days: - Reading and writing to VGA registers like Sequencer, Graphics, Attribute, CRTC and General are verified OK within Bios code; - VGA mode set sequence is verified; - Programming to other extended registers (Bios scratch/Grph_surfacex/Memory_Base/HDP...) that used to bring up VGA mode are verified; - Understood why Bios code would fly away when one routine inside Bios gets called, it has nothing to do with IKOS or Netlist. However, access to VGA host aperture still hangs up the system. I don't know if it's caused by missing steps in SW or the netlist is not good enough yet. Once this one gets solved and display capture is ready, we should be able to bring up the first VGA mode with CRT on Crayola. **[18:39 Gabriel Abarca]** Mahendra found an issue with a MDFFHQX1 and/or TLATNTSCAX12 cell which is being investigated, this can be the VGA problem.
Thu 03/06/2003	**[11:04 Brian Leblanc]** I have logged off the Ikos box. I was debugging the register test using indexed accesses and writing and reading using 8/16/32 bit accesses. The tests produced the expected results for the 3.0 net list and the program is starting to look pretty good. I have not removed registers from the test that are failing, whether they fail for real reasons or net list issues is not important for now. I will start looking at the register failures seriously once the 4.0 or 4.1 net list is available. I still need to debug the defaults and endian tests. **[15:25 Gina Seto]** No display yet. - reran palette r/w test on the 3.0.1 database. tests passed. - Fixed the hang (bug in shell) and attempted to run a basic display test: 640x480, 8bpp indexed mode. - During the test, I checked a bunch of registers and these have been written as expected (see list below) - The program writes to the frame buffer, then waits for a transition in V_BLANK. It hangs here. (15 minutes+). - Colin will be making a new database based on 3.0.1 that will allow me to do display captures Registers written: HDP_FB_START = 0xf0000000 FB_START = 0xf0000000 FB_SIZE = 0x0 D1GRPH_CONTROL = 0x0 D1GRPH_ENABLE = 0x1 D1GRPH_PRIMARY_SURFACE_ADDRESS = 0xf0000000 D1GRPH_X_START = 0x0 D1GRPH_X_END = 0x280 D1GRPH_Y_START = 0x0 D1GRPH_Y_END = 0x1e0 D1CRTC_CONTROL = 0x101 D1CRTC_H_TOTAL = 0x340 D1CRTC_H_SYNC_A = 0x870000 D1CRTC_H_SYNC_B = 0x870000 D1CRTC_H_BLANK_START_END = 0x880308 D1CRTC_V_TOTAL = 0x1fd D1CRTC_V_SYNC_A = 0x30000 D1CRTC_V_SYNC_B = 0x30000 D1CRTC_V_BLANK_START_END = 0x1c01fc DACA_ENABLE = 0x1 **[18:09 Colin Stewart]** A database with the fixed display clock and data capture for the 30M is available.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.