



# DDR SDRAM Automated Memory Qualification Using Kozio kDiagnostics™

## Overview

This application note provides an explanation of how one uses Kozio kDiagnostics™ to automatically determine DDR controller timing window values and automatically qualify memory operations using those timing values.

In short, this document defines a procedure for using Kozio software for memory qualification. Memory qualification in this document is defined as: ***Determining what DDR SDRAM controller settings and ranges, if any, that allow a particular memory part to function properly.***

The steps required to perform the memory qualification are outlined. kScript™ is the proprietary scripting language accepted by Kozio products allowing users to dynamically create new test procedures.

This document provides an example of running the memory qualification procedure on a platform using an AMCC PowerPC™ 440EP processor. The platform used for this application note contained a 440EP, DDR SDRAM, SRAM, Flash, Serial, and other components not pertinent to this discussion.

## Qualification Procedure

Memory qualification using Kozio's test suite can be described in three steps:

- Step 1 - Initialization
- Step 2 - Rough Timing Scan & Stage Selection
- Step 3 - Full Qualification and Timing Window Determination

The rest of this document details each step and describes how to use the Kozio interfaces to complete the memory qualification procedure.

The only requirement prior to initialization is that you must be operating on known good hardware. If the hardware, or module, has yet to be verified, Kozio's diagnostics test suites can be used to comprehensively test the module and aid in debugging any issues found.



## Step 1 - Initialization

Primarily, the initialization step requires knowledge of the DDR SDRAM memory being qualified and the conditions under which it is designed to operate. These details can be found from the manufacturer's datasheet for the memory, or through the manufacturer directly.

Kozio's memory qualification suite only needs a subset of the typical Serial Presence Detect (SPD) data in order to operate. If DIMM(s) will be used, the SPD data can be read from the EEPROM on the DIMM and no user input is required. However, if your design uses memory soldered onto the PCB without any EEPROM, a few parameters need to be initialized in the qualification script so that the software will be able to select optimal settings.

These parameters are:

Name	Description
Row Addr	Number of Row Addresses
Col Addr	Number of Column Addresses
Banks	Number of Banks on Board (Not banks internal to the SDRAM chips)
Cycle @ 2.5	Cycle Time at CAS 2.5
ECC	ECC enabled or disabled
Refresh	Refresh Rate
WCSBC	Min Clock Delay for Back to Back Rand Column Addresses
CL	CAS Latency
Cycle @ 2.0	Cycle Time at CAS 2.0
Cycle @ 1.5	Cycle Time at CAS 1.5
Trp	Minimum Row Precharge Time
Tred	Minimum RAS to CAS delay
Tras	Minimum Active to Precharge Time
Bank Size	Bank size

Also during setup, it is useful to determine which tests from Kozio's library will be useful to run during the qualification. All available tests are documented in the Kozio Command Reference document, and Technical Support is available if you need assistance in choosing the right set of tests.

Kozio SDRAM test procedures include: SDRAM Address Bus Test, SDRAM Burst Test, SDRAM Data Bus Burst Noise Test, SDRAM Byte Test, SDRAM Data Zero Test, SDRAM Data One Test, SDRAM Word Test, SDRAM Data Bus Noise Test, SDRAM March Test, SDRAM Transitional Fault Test, SDRAM Simultaneous Switching Output (SSO) Test.



One other step that may be part of initialization is to set the external module conditions, such as voltage or temperature. External conditions may be set though a variety of interfaces, such as I<sup>2</sup>C, external busses, or SPI. All of these interfaces can be accessed through Kozio's kScript™ command line interface, and may be included as part of your qualification script during initialization.

To aid in creation of the initialization portion of the memory qualification, the following partial kScript can be referenced to help determine the information needed during initialization step. (A kScript is a text file containing formatted commands understood by Kozio software products.)

```
\ Create an SDRAM test suite which will be used to test one set of parameters
=> test.sdr.am.data0
=> test.sdr.am.data1
=> test.sdr.am.data.noise
=> test.sdr.am.address
=> test.sdr.am.byte
=> test.sdr.am.word
=> test.sdr.am.sso
=> test.sdr.am.burst.noise
8 tests test.group.qual

S" Kozio Memory Qualification Suite" => test.group.qual SUITE test.sdr.am.qual

\ Create the structure to store typical SPD data
SDRAM{ NEW.MEMORY
    13          , \ number of row addr
    9          , \ number of col addr
    1          , \ number of banks
    0x40       , \ cycle time @ CAS 2.5
              , \ [(high nibble * 1ns)+(low nibble*0.1ns)]
    CONST.SDRAM.NO_ECC , \ ecc [NO_ECC, ECC]
    CONST.SDRAM.RFRSH_1P00 , \ refresh rate
              , \ [1P00, 0P25, 0P50, 2P00, 4P00, 8P00]
    1          , \ wcsbc
              , \ Min Clock Delay,
              , \ Back to Back Rand Col Addr
    CONST.SDRAM.CAS_3_0 , \ CAS Latency
              , \ [CAS_1_5, CAS_2_0, CAS_2_5, CAS_3_0]
    0xA0       , \ cycle time @ CAS 2.0
              , \ [(high nibble * 1ns)+(low nibble*0.1ns)]
    0xC0       , \ cycle time @ CAS 1.5
              , \ [(high nibble * 1ns)+(low nibble*0.1ns)]
    64         , \ Trp (0.25 ns)
    64         , \ Trcd (0.25 ns)
    36         , \ Tras (ns)
    16         , \ bank size (4MB * val)
}SDRAM

NEW.MEMORY var.sdr.am.config write \ sets up SDRAM using parameters
              \ of "NEW.MEMORY"

1 var.sdr.am.dimms write \ sets SDRAM to only expect a single
```

**Figure 1. Partial kScript™ for Initialization**

## Step 2 - Rough Timing Scan & Stage Selection

There are only two commands and one variable that control the rough timing scan and stage selection; however, the details obtained from this step are crucial to understanding the rest of the memory qualification.

There are three options for controlling the point at which data is physically read from the DDR SDRAM, referred to as Stage 1, Stage 2, and Stage 3. Only one stage should be used, and the appropriate stage is determined by the size of the window available when compared to the period of the memory clock. In general, the largest window will yield the most reliable results by selecting a value in the middle of that window.

When Stage 1 timing is used, the **DDR Clock Delay Tuning** (DCDT) parameter is configured to control the amount of delay applied to both the read and the write cycles. With Stage 2 and Stage 3 timing, the **Read Clock Delay Tuning** (RDCT) parameter is configured to control the amount of delay applied for reading data from SDRAM. All three stages can be visualized as a pipeline, with operations completing the quickest using Stage 1, and incrementally slower when using Stage 2 or Stage 3.

Kozio's software scans for available timing windows in all three Stages, and then allows the user to select the best stage based on the data gathered. The kDiagnostics command **sdram.timing.scan** will determine the windows, and then the kDiagnostics command **sdram.display.settings** allows the results to be viewed. The following is a sample output following the execution of these two commands.

```
kdiags-ins> 0 var.sdram.stage.select write
kdiags-ins> sdram.timing.scan
kdiags-ins> sdram.display.settings
Rough Windows:
  Stage  Parameter      Min    Max    Average    % MemClk
  ----  -
  1      DCDT             0     85     42         24%
  2      RDCT            283   319    301         10%
  3      RDCT            146   147    146          0%

Selected Window:
  Stage  Min    Max    Average    % MemClk
  ----  -
  1      0     85     42         24%

Register Dump:
SDRAM_BESR0 = 0x00000000    SDRAM_BESR1 = 0x00000000
SDRAM_BEAR  = 0x00000000    SDRAM_MIRQ  = 0x00000000
SDRAM_SLI0  = 0x00000000    SDRAM_CFG0  = 0x84000000
SDRAM_CFG1  = 0x00000000    SDRAM_DEVOPT = 0x00000000
SDRAM_MCSTS = 0xA0000000    SDRAM_PMIT  = 0x04080000
SDRAM_PMIT  = 0x00000000    SDRAM_UABBA = 0x00000000
SDRAM_B0CR  = 0x00082001    SDRAM_B1CR  = 0x00000000
SDRAM_B2CR  = 0x00000000    SDRAM_B3CR  = 0x00000000
SDRAM_TR0   = 0x41894012    SDRAM_TR1   = 0x00000000
SDRAM_CLKTR = 0x4000002A    SDRAM_WDDCTR = 0x00000000
SDRAM_DLYCAL = 0x200000AC    SDRAM_ECCESR = 0x00000000
SDRAM_CID   = 0x320B0000    SDRAM_RID   = 0x00003200
```

Figure 2. kDiagnostics™ Screen Shot

As mentioned previously, the largest window available will generally yield the most reliable results. The maximum size of the window is limited by the design of the DDR SDRAM controller. The **DCDT**



maximum window is 1/4 of the memory clock period. The **RDCT** maximum window is 1/2 of the memory clock period. The precise memory clock period can be modified by external conditions and is difficult to calculate. However, Kozio's software can reliably list the window sizes as a percent of the memory clock period based on calibration data available in the DDR SDRAM controller.

In this case, Stage 1 is the most desirable setting with a window size encompassing 24% of the memory clock period. Stage 2 has a window size of 11% of the memory clock period, and Stage 3 only covers 1% of the memory clock period.

After determining what window works best in your design, the kDiagnostics variable **var.sdram.stage.select** can be used to force Kozio's software to use a particular stage, as long as it is available. If the stage being forced is not available, then the fastest stage available will be used. If the variable is set to a value of zero, the software will automatically select the fastest stage available.

**NOTE:** Changes in **var.sdram.stage.select** are not applied until **sdram.timing.scan** is executed. The currently selected stage will be displayed using the command **sdram.display.settings**.

The following partial kScript can be used as a reference when determining how to use the kDiagnostics commands just described as part of the rough timing scan and stage selection.

```
0 var.sdram.stage.select write      \ specify the read stage to use
                                     \ 0 - automatic, selects fastest
                                     \ 1 - select stage one if available
                                     \ 2 - select stage two if available
                                     \ 3 - select stage three if available

sdram.timing.scan                   \ determines the DDR clock delay
                                     \ tuning window

sdram.display.settings
```

**Figure 3. kScript™ Example of kDiagnostics Commands**



### **Step 3 - Full Qualification and Timing Window Determination**

Once the rough windows have been determined and a stage has been selected, the full qualification of the memory can be run within that stage to determine the final window.

During the full qualification test, a much more comprehensive memory test is run for each value of the rough window within the stage selected. This test can catch memory problems not found by the methods used to determine the rough window. Via scripting, the full qualification will determine the final valid window to be used with a particular type of memory.

During this step, two kDiagnostics' variables and one kDiagnostics' command are used to modify and test the settings within the SDRAM controller. The variables **var.sdram.timing.min** and **var.sdram.timing.max** store the minimum and maximum values for the currently selected window.

The kDiagnostics' command **sdram.timing.set** will apply a new timing value to the DDR SDRAM controller. It is important to note that this command will modify a different set of registers depending on the stage that has been selected.

The following sample kScript will loop through the rough window values starting at **var.sdram.timing.min** up to **var.sdram.timing.max**. This loop checks for the first time the qualification suite passes, and sets the new minimum window value to track where the new window starts. As the test continues passing, the new window maximum value is updated. When the test fails the loop exits and we have found the new valid window.



```
0 VALUE val.new.min
0 VALUE val.new.max

: do_qual
  -1 -> val.new.min
  -1 -> val.new.max
  \ loop through timing window
  var.sdram.timing.max read 1+ var.sdram.timing.min read do
  \ set the timing to the next value
  i sdram.timing.set
  \ run the qualification suite
  test.sdram.qual
  \ check for errors
  errorlevel.passed?
  if
    \ check if the min/max are equal
    val.new.min val.new.max =
    if
      \ set the new min if the two value are equal
      i -> val.new.min
    else
      \ otherwise set the max
      i -> val.new.max
    then
  else
    \ see if max is greater than min
    val.new.max val.new.min >
    if
      \ if it is, window is found, exit
      leave
    then
  then
  \ continue the loop
  loop

  \ check to see if only min was updated
  val.new.min val.new.max >
  if
    \ if so, set max to min
    val.new.min -> val.new.max
  then

  \ update the global min
  val.new.min var.sdram.timing.min write
  \ update the global max
  val.new.max var.sdram.timing.max write

  \ set the timing to min
  val.new.min sdram.timing.set

  sdram.display.settings
;
```

**Figure 4. Partial kScript™ Example for Timing Window Determination**

While running, many progress bars will be displayed on the screen as tests repeat for each setting of the memory window. When the script completes, it automatically runs **sdram.display.settings** which will show you the final values that have been determined during qualification. Figure 5. shows that Stage 2 timing was selected and the full qualification determined a window size of 6% – a significant reduction from the 11% determined by the rough timing scan.

In the examples presented in this document, Stage 1 timing is actually the best option, covering 24% of the memory clock period. However, the rough window and the full qualification window are the same size with Stage 1 in the example. Therefore, Stage 2 timing was selected to demonstrate how the final window can differ from the rough windows.



```
Rough Windows:
  Stage  Parameter      Min  Max  Average  % MemClk
  -----  -----  ---  ---  -----  -
    1     DCDT           0   85    42     24%
    2     RDCT          274  314   294     11%
    3     RDCT          140  146   143      1%

Selected Window:
  Stage  Min  Max  Average  % MemClk
  -----  ---  ---  -----  -
    2     287 309   298      6%

Register Dump:
SDRAM_BESR0 = 0x00000000    SDRAM_BESR1 = 0x00000000
SDRAM_BEAR  = 0x00000000    SDRAM_MIRQ  = 0x00000000
SDRAM_SLI0  = 0x00000000    SDRAM_CFG0  = 0x84000000
SDRAM_CFG1  = 0x00000000    SDRAM_DEVOPT = 0x00000000
SDRAM_MCSTS = 0xA0000000    SDRAM_PMIT  = 0x04080000
SDRAM_PMIT  = 0x00000000    SDRAM_UABBA = 0x00000000
SDRAM_B0CR  = 0x00082001    SDRAM_B1CR  = 0x00000000
SDRAM_B2CR  = 0x00000000    SDRAM_B3CR  = 0x00000000
SDRAM_TR0   = 0x41894012    SDRAM_TR1   = 0x4040011F
SDRAM_CLKTR = 0x40000000    SDRAM_WDDCTR = 0x00000000
SDRAM_DLYCAL = 0x200000AC    SDRAM_ECCESR = 0x00000000
SDRAM_CID   = 0x320B0000    SDRAM_RID   = 0x00003200
```

Figure 5. Results Displayed by kDiagnostics™ Command “sdram.display.settings”



## Summary

Kozio's software, SDRAM test routines, and script interface provide an excellent mechanism to find the optimal DDR controller settings.

Commands are provided to determine a rough timing window for each of the three memory Stages followed by commands to select a desired Stage and then execute full qualification for each timing value in that Stage's window.

The full qualification runs a suite of SDRAM memory tests to determine if memory operates correctly at an individual timing value. Once completed, a safe window is determined for the current operating parameters, such as temperature, voltage and frequency.

Using additional Kozio commands, one can create a script that executes the full memory qualification process over a range of temperatures or voltages.

Using embedded software running on the target platform at true speeds provides the best mechanism for qualifying memory. Kozio software products delivery turnkey solutions at a fraction of the cost it would take to develop them internally.

## Contacts for Additional Information

Kozio, Inc.  
2400 Trade Centre Ave  
Longmont, CO 80503  
+1 (303) 776-1356 x1  
[sales@kozio.com](mailto:sales@kozio.com)  
[www.kozio.com](http://www.kozio.com)

Kozio, kDiagnostics, kManufacturing, kPOST and kScript are registered trademarks of Kozio, Inc. All other trademarks are property of their respective owners.

### **About Kozio, Inc.**

Kozio delivers unique comprehensive hardware test software for custom ARM/XScale, PowerPC, MIPS32, and DSP based circuit boards. Kozio delivers proven test software for board bring-up, manufacturing test, and self-test saving months of effort. kDiagnostics™ produces a full hardware test report, component by component, within minutes of running. It is configured and delivered ready-to-run before your prototype arrives. kManufacturing™ provides extensive functional testing in an automated stand-alone embedded application. Streamline your product testing and serialization while reducing cost. kPOST™ easily integrates with your embedded application and ships with your product reducing in-field support costs while providing customer assurance.