Technical notes on using Analog Devices DSPs, processors and development tools
Visit our Web resources http://www.analog.com/ee-notes and http://www.analog.com/processors or e-mail processor.support@analog.com or processor.tools.support@analog.com for technical support.

Interfacing NAND Flash Memory with ADSP-2126x SHARC® Processors

Contributed by Aseem Vasudev Prabhugaonkar *Rev 1 – November 17, 2005*

Introduction

NAND flash provides an alternative to the hard drives, especially for portable and handheld systems. Cellular phones, personal digital assistants (PDAs), digital cameras, MP3 players and other mobile computing, communications, and consumer products use flash memory as their primary storage media. Applications that use NAND flash memory to store information such as images, data, and music require much higher density.

This application note describes how to interface NAND flash memory to ADSP-2126x SHARC® processors, even though the ADSP-2126x does not have a NAND flash controller on-chip and its parallel port has multiplexed address and data signals. This interface is demonstrated by code examples, which are validated on an EZ-KIT Lite® evaluation system. The memory device part used to demonstrate this interface is the K9F1208U0B, manufactured by Samsung Electronics.

NAND Flash versus NOR Flash

The most important parameter regarding memories is cost per bit. In semiconductor memory, bit cost depends on memory cell area per bit. The cell area of NAND flash memory is smaller than that of NOR flash, and hence NAND flash can be less expensive than NOR flash.

Functionally, the primary difference between NAND flash and NOR flash is that NAND flash is block accessible and NOR flash is byte accessible.

NAND flash has the advantage of short erasing and programming times. The programming current into the floating gate is very small because NAND flash uses Fowler-Nordheim tunneling for erasing and programming. Therefore, power consumption for programming does not significantly increase, even as the number of memory cells being programmed increases. As a result, many NAND flash memory cells can be programmed simultaneously, decreasing the programming time per byte. Conversely, NOR flash can be programmed by byte or word only. Since it uses the hot electron injection mechanism for programming, it also consumes more power and the programming time per byte is longer. Typically, the programming time for NOR flash is more than an order of magnitude greater than that of NAND flash.

From the system designer's point of view, the most striking difference between NAND flash and NOR flash is the hardware interface. NOR flash has a fully memory-mapped random access interface like an EPROM, with dedicated address lines and data lines. This interface makes it easy to "boot" NOR flash systems. On the other hand, NAND flash has no dedicated address lines. It is controlled using an indirect I/O-like interface and by sending commands and addresses through an 8-bit bus to an internal command and address

register. Therefore, NOR flash memories are mainly used to boot-load a processor, and NAND flash memories are used to store large amounts of audio and video data.

Overview of the ADSP-2126x Processor's Parallel Port

There is a major difference between the external port found in earlier SHARC processors (such as the ADSP-2106x and ADSP-2116x generations of processors) and the parallel port found on ADSP-2126x processors. There are no dedicated address lines and data lines, and there are no dedicated memory select strobes (/MSx) in ADSP-2126x processors.

The ADSP-2126x has a parallel port that allows bidirectional transfers between it and external parallel devices. Using the parallel port bus and control lines, the processor can interface to 8- or 16-bit-wide external memory devices. The parallel port provides a DMA interface between internal and external memory and can support core-driven data transfer modes. Regardless whether 8- or 16-bit external memory devices are used, the internal data word size is 32 bits (normal word addressing) and the parallel port employs packing to place the data appropriately. The parallel port has address lines and data lines multiplexed. It supports interfacing to 8- and 16-bit-wide devices and hence provides 8-to-32 and 16-to-32 bit data packing. NO-PACK mode is not available. Since the internal data word size is always 32 bits, the data is driven out on the parallel port as four 8-bit words in 8-to-32 bit packing mode (two 16-bit words in 16-to-32 bit packing mode). The parallel port comprises the following signals.

- Address/Data AD[15:0] pins: The address and data information is time-multiplexed on AD[15:0] lines.
- ALE (Address Latch Enable): This signal is an indication of address being driven on the multiplexed address/data bus, and hence, it is used to latch the address bits. This signal can be programmed as an active high or an active low signal.
- /RD and /WR strobes: /RD indicates a read access, and /WR indicates a write access to the device. The data is latched into the processor on the rising edge of the /RD strobe. The rising edge of the /WR strobe can be used by external devices like memory to latch data.

There are no memory select strobes (/MSx). If a design requires memory select strobes, they should be generated from the higher address bits using decoder ICs.

K9F1208U0B NAND Flash

The K9F1208U0B is offered in a 64M x 8-bit configuration. The K9F1208X0B is 512 Mbits with spare 16 Mbits capacity. The device is offered in 1.8V, 2.7V, and 3.3V supply voltage ranges. Its NAND cell provides the most cost-effective solution for solid-state mass storage. A program operation can be performed in 200 µs (typical) on the 528-byte page. An erase operation can be performed in 2 ms (typical) on a 16-Kbyte block. Data in the page can be read out at 50 ns (60 ns for K9F1208R0B) cycle time per byte. The I/O pins serve as the ports for address and data input/output as well as command input.

NAND Flash Interface

NAND flash, which is controlled using I/O pins, does not have dedicated address and data lines. Thus, the programmer must use additional flash control signals to send commands that perform various tasks like read and write. The basic commands supported by NAND flash memories are block erase, block program, read status, and block read. Figure 1 shows the pinout of NAND flash in a TSOP1 package.

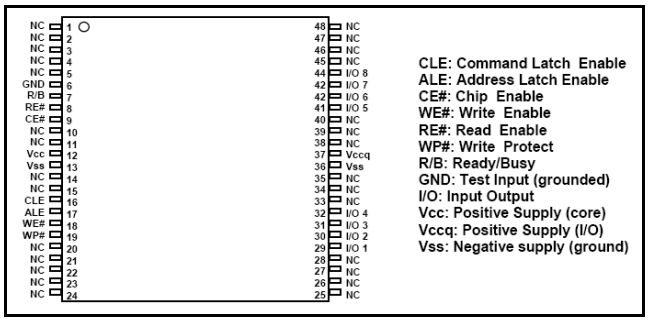

Figure 1. NAND Flash Pinout in a TSOP1 Package

Assert chip enable (/CE) logic low to access the device. It enables the NAND flash to accept bytes written to the chip when write enable (/WE) is asserted low (and to enable the output of a data byte when read enable (/RE) is asserted low). When /CE is high, the chip ignores /RE and /WE and the I/O is three-stated. Command Latch Enable (CLE) is used to send commands to the device when /CE is asserted. Address Latch Enable (ALE) is used to latch the address into the flash's address register. Table 1 indicates the selected internal register for valid combinations of CLE and ALE.

ALE	CLE	Register Selected
0	0	Data Register
0	1	Command Register
1	0	Address Register
1	1	Not Defined

Table 1. Combinations of ALE and CLE, when CS# is Asserted

The flash memory is accessed in terms of column, page, and block. Read and program operations take place on a per-page basis. An erase operation takes place on a block basis. One page consists of 528 bytes. The size of the data register is therefore 528 bytes. One block comprises 32 such pages. There are three basic operations in a NAND flash: reading a page, programming a page, and erasing a block. Figure 2 shows the organization of K9F1208U0B flash memory in terms of pages and blocks.

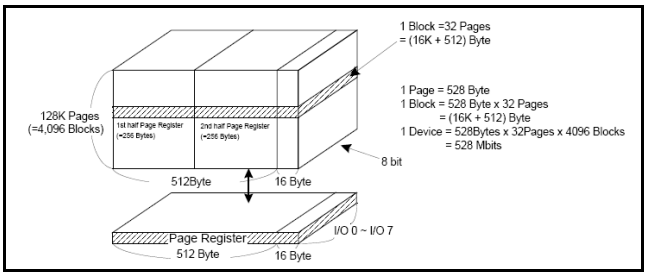

Figure 2. K9F1208U0B Flash Memory Organization

The command phase comprises a single byte transfer to the command register. The address phase comprises a series of byte transfers, which depend on the size of the flash. The address phase consists of the column address byte followed by a series of row address bytes. The number of row address bytes depends on the number of blocks, and hence the size of the memory device.

Flash Commands

This discusses the primary NAND flash commands. The primary and very commonly used flash memory commands are read page, program page, read status, and erase block. Every command (except read status) consists of a command write phase followed by address write phase. The read status command does not have an address write phase. The command is written into the flash's command register followed by the start address for the read or program operation latched into address register.

READ Operation

After issuing the read command and the address, the read operation is performed from the flash memory into its internal data register. Once the internal read operation is completed, as indicated by the R/B# (BUSY) signal, the data register can be read over the external data bus. The various phases are explained as follows.

- **Command Phase**: With /CS asserted, CLE=1, and ALE=0, the command byte (00H) is placed on the I/O pins. This is a command

write operation. On the rising edge of /WE, the "read mode 1" command is latched into the command register.

- **Address Phase**: With CS# asserted, ALE=1, and CLE =0, the column and row address is latched into the address register. This operation is usually a series of writes. The byte-wide addresses are latched on the rising edge of every WR# pulse. The first address byte is the column address, and the subsequent bytes are the row addresses. The five least-significant bits of first row address indicate the page number within a block. The three most-significant bits of the first row address and the rest of the higher row address bits determine the block.
- **Data Transfer Phase**: With CS# asserted, CLE=0, and ALE=0, the device goes into a busy state to transfer data from memory into the on-chip data register. This is indicated by the BUSY (R/B#) signal. R/B# goes to logic low and remains at logic low until the data-transfer phase is completed. Once the R/B# goes back to logic level high, the data can be read from data register.
- **Read-Out Phase**: This is indicated by R/B# going to logic level high after the busy period. With CS# asserted, ALE=0, and CLE=0, the data can be read over I/O with a series of RD# pulses.

Figure 3 shows the timing diagram for a read operation.

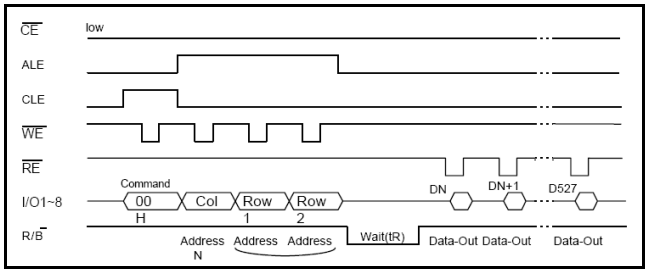

Figure 3. Read Operation Timing

Program Operation

This is a memory write operation. The data in the on-chip data register is written into memory. Similar to a read operation, the device asserts R/B# to logic level low, indicating its busy status. The various phases of this operation are explained below.

- **Command Phase**: With CS# asserted, CLE=1, and ALE=0, the command byte (0x80) is placed on the I/O pins. This is a command write operation. On the rising edge of /WE, the "serial data input" command is latched into the command register.
- **Address Phase**: This is similar to the address phase (See READ Operation on page 1). With CS# asserted, ALE=1, and CLE =0, the column and row addresses are latched into the address register.
- **Data Input Phase**: With CS# asserted, ALE=0, and CLE=0, the data is written into the data register. The data is driven on the I/O lines, and every byte is latched on the rising edge of WR#.
- **Program Phase**: With CS# asserted, ALE=0, and CLE=1, the auto program command is written into the flash's command register. The device then goes to the BUSY state, indicated by R/B# going to a logic level low state.
- **Timeout Check Phase**: The status is checked after the completion of the program phase to determine whether the programming was successful.

Figure 4 shows the timing diagram for a program operation.

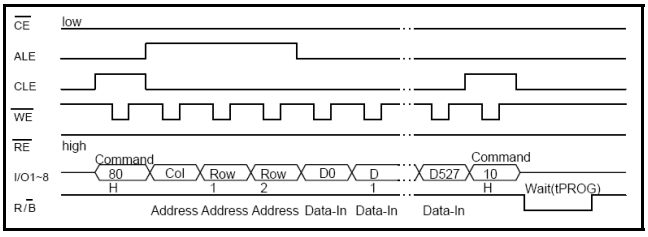

Figure 4. Program Operation Timing

Block Erase Operation

In a block erase operation, a group of consecutive pages (typically 32) is erased in a single operation. Although programming turns bits from "1" to "0", block erasure is necessary to turn bits from "0" back to "1".

- **Command Phase**: With CS# asserted, ALE=0, and CLE=1, command byte 0x60 (auto block erase) is written to the flash's command register. This is done by placing the command byte (0x60) on the I/O lines. The rising edge of WR# latches this value into the command register.
- **Address Phase**: Since this is a block erase, column and page addresses are not required. With CS# asserted, ALE=1, and CLE=0, the block address is written to the address register with series of writes into the flash.
- **Erase Phase**: With CS# asserted, ALE=0, and CLE=1, the auto block erase confirm command (0xD0) is written to the command register. The device then goes to a BUSY state to complete the block erase operation.
- **Timeout Check Phase**: The status is checked after the completion of the program phase to determine whether the erasure was successful.

Figure 5 shows the timing diagram for a block erase operation.

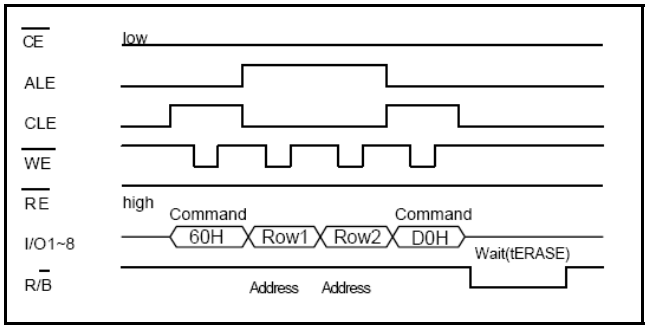

Figure 5. Block Erase Operation Timing

Interfacing to ADSP-2126x Processors

The NAND flash interfaces with the ADSP-2126x processor on its parallel port. The ADSP-2126x processor does not have a dedicated on-chip NAND flash controller; therefore, you have to use software driver code to control and use the NAND flash. Standard NAND flash memories require that the chip enable remains asserted during the read busy period. Thus, using the processor's FLAG pin to drive the flash's chip requires that the driver code is compatible with both standard NAND flash and CEDC (chip enable don't care) NAND flash memories. Moreover, since the memory select strobes are not supported by the ADSP-2126x, using the FLAG signal to drive the flash's chip select helps avoid additional glue logic such as a decoder.

Figure 6 shows a block diagram of the interface between the ADSP-2126x and the NAND flash memory.

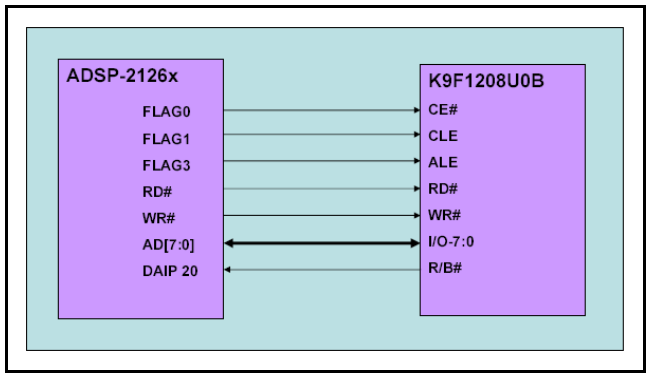

Figure 6. ADSP-2126x Processor and Flash Memory Interface Block Diagram

The FLAG0, FLAG1, and FLAG2 signals from the processors drive the /CE, CLE, and ALE signals of the flash memory, respectively. The processor's parallel port is configured in 8-bit mode and hence 8(external)-32(internal) packing is enabled. The R/B# (BUSY) signal from the flash memory is connected to DAIP20 of the processor. This DAI pin, which is configured as an input, is connected internally to MISCA0 through SRU (Signal Routing Unit) signal mapping. The DAI

low-priority interrupt is enabled and is configured to trigger an interrupt on the rising edge. R/B# is driven logic low while the flash is performing an operation and goes to logic level high when the operation completes. This triggers the DAI interrupt, indicating that the flash is ready.

Figure 7 shows the main code flow. This consists of setting up FLAGs, configuring SRU registers, setting up the DAI interrupt and parallel port DMA interrupt. The data to be written to the flash memory is initialized in the processor's internal memory. The various flash memory commands are initialized in internal memory as arrays.

Required parameters are passed onto the subroutines, which are called from the main program. The parameters include an index or pointer to command array, column and row addresses, and the parallel port DMA parameters.

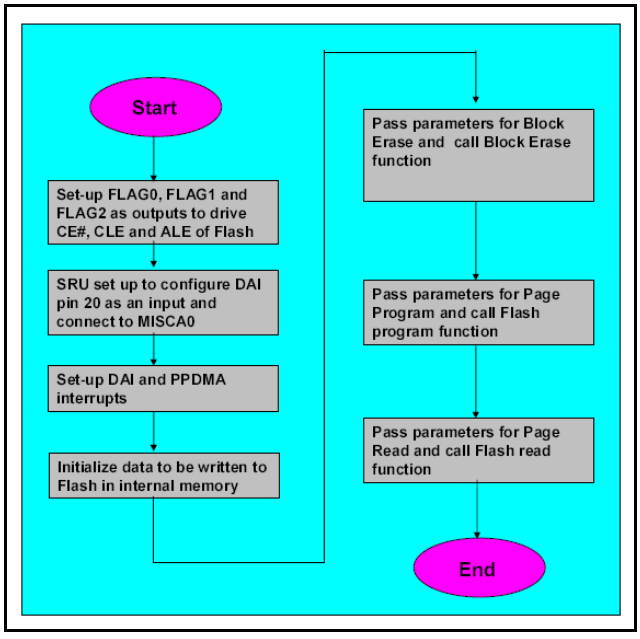

Figure 7. Flowchart of Main Application Code Flow

Figure 8 shows the flowchart for the block erase function implementation. Parallel port DMA is used to write commands, column and row addresses, write data and read data to/from the flash memory. For a single byte command, the DMA parameters are programmed such that the external byte count is "1". The command byte should be the least-significant byte of the 32-bit internal memory word. The DMA interrupt is generated when the internal and external DMA counts expire. In this case (with external byte count as "1"), only the least-significant byte is transferred and the DMA stops. This approach ensures that only a single WR# pulse is generated and that no spurious/extra WR# pulses are generated by the processor. This approach greatly helps, especially because this processor does not support NO PACK mode for the parallel port.

For this interface, program the hold cycle for the parallel port to ensure that the flash timing requirements are met.

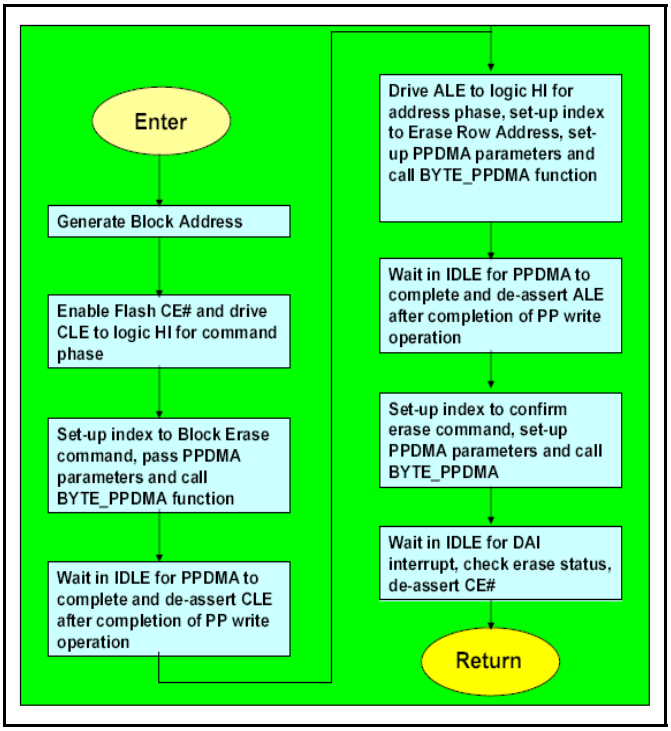

Figure 8. Flowchart of Block Erase Function

Figure 9 shows the flowchart for the page program function implementation.

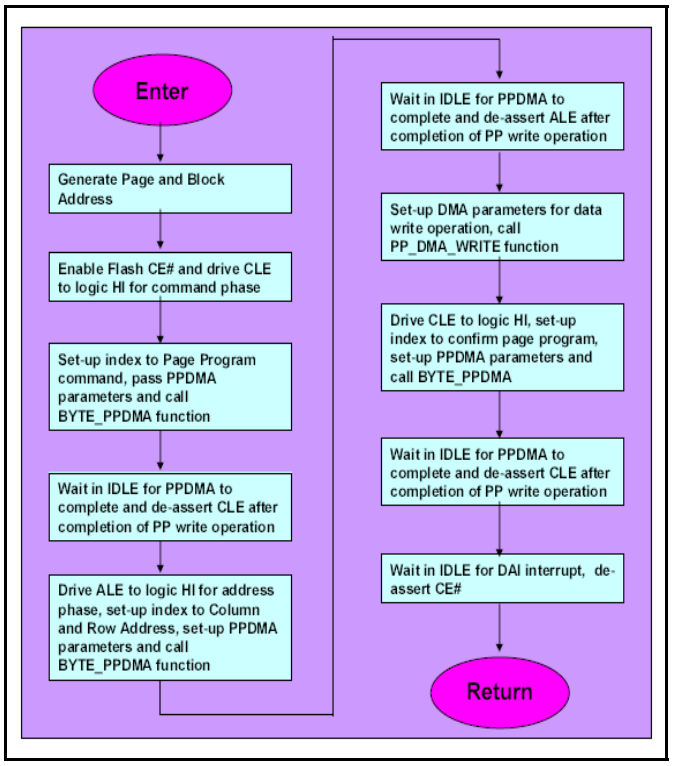

Figure 9. Flowchart of Page Program Function

If the data to be written to the flash and read back from the flash is 8-bit data, it must be packed properly as a 32-bit internal memory word for data write operations (and unpacked into four 8-bit bytes from the received 32-bit word for data read operations).

Figure 10 shows the flowchart for the page read function implementation.

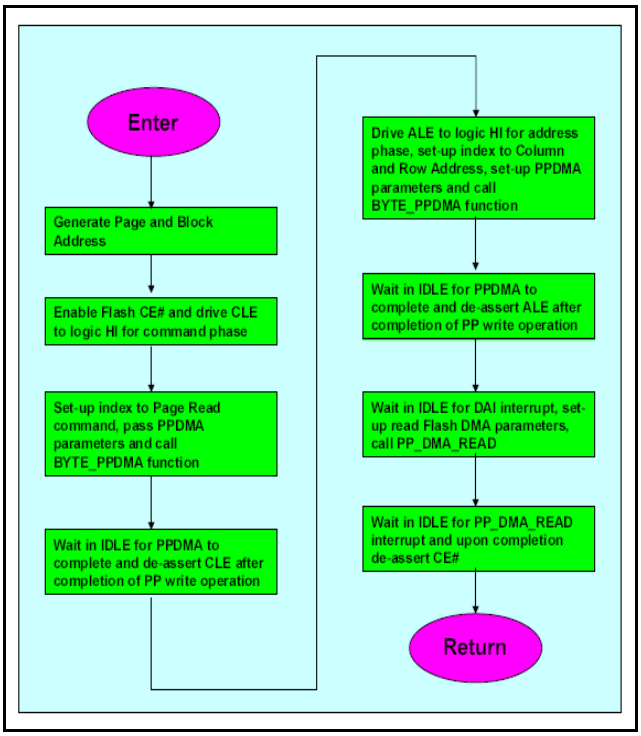

Figure 10. Flowchart of Page Read Function

Logic analyzer screenshots representing timing diagrams for this interface are shown in the following figures.

Summary

The interface between an ADSP-2126x SHARC processor and a NAND flash can be achieved seamlessly without using any external glue logic even though this processor does not have an on-chip flash memory controller. The parallel port DMA features and modes make this interface easy to achieve in terms of both hardware interface and the software driver code.

Software Code

Code is supplied in the `.ZIP` file that is associated with this EE-Note.

Appendix: Logic Analyzer Plots

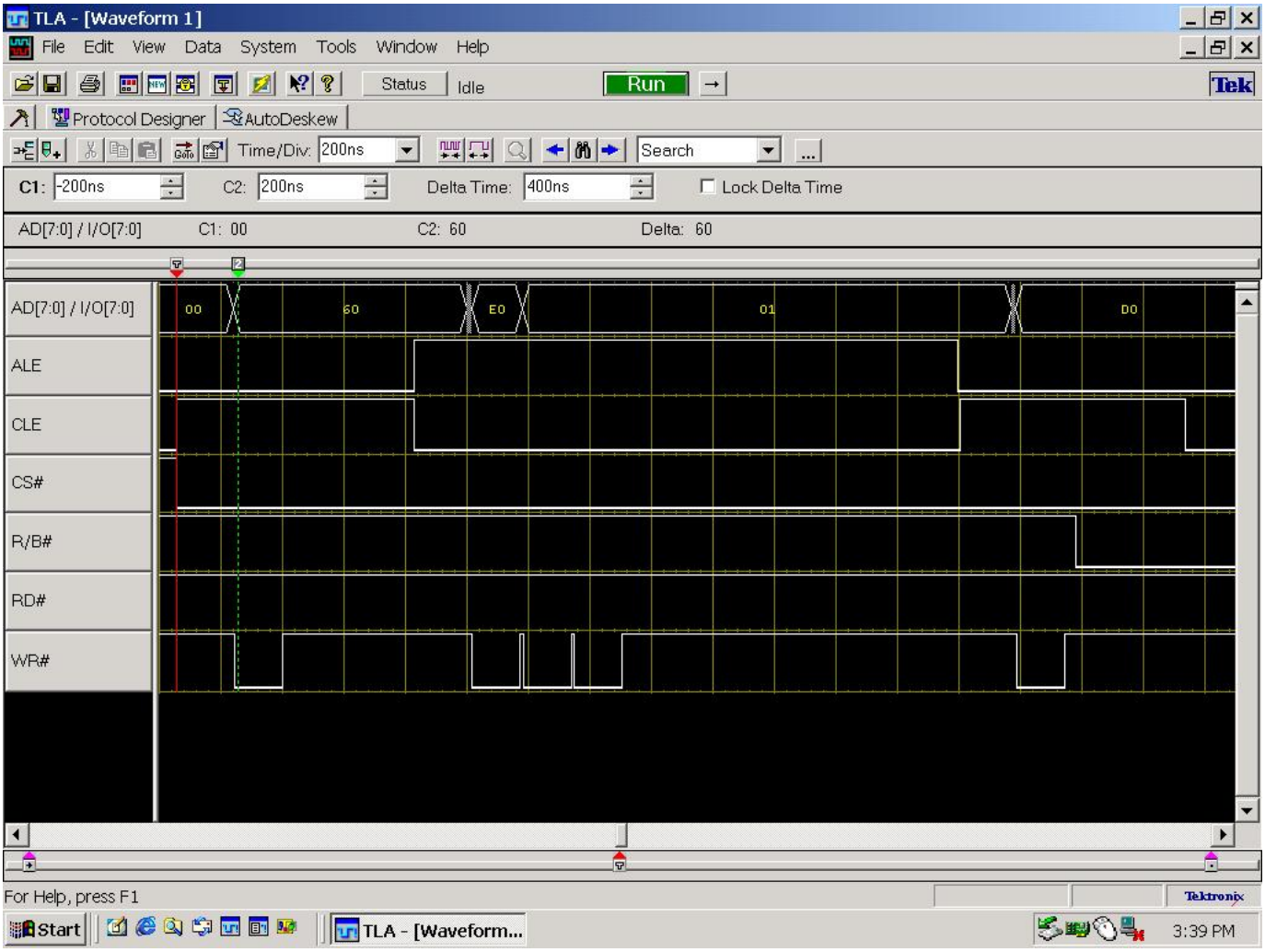

Figure 11. Start of Erase Command Followed by Erase Command Confirmation

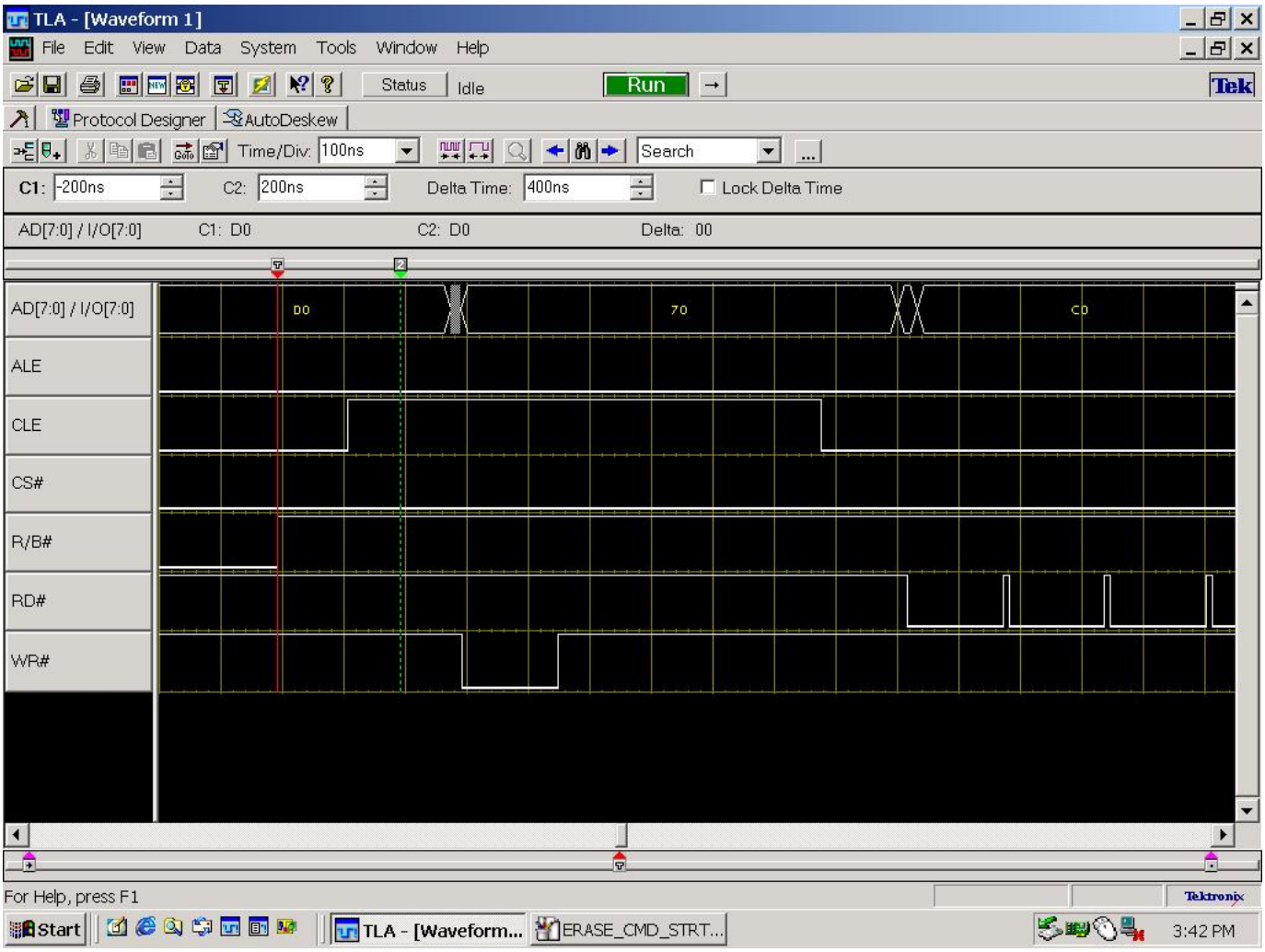

Figure 12. End of Erase Operation as Indicated by R/B# Followed by Check Erase Status Command

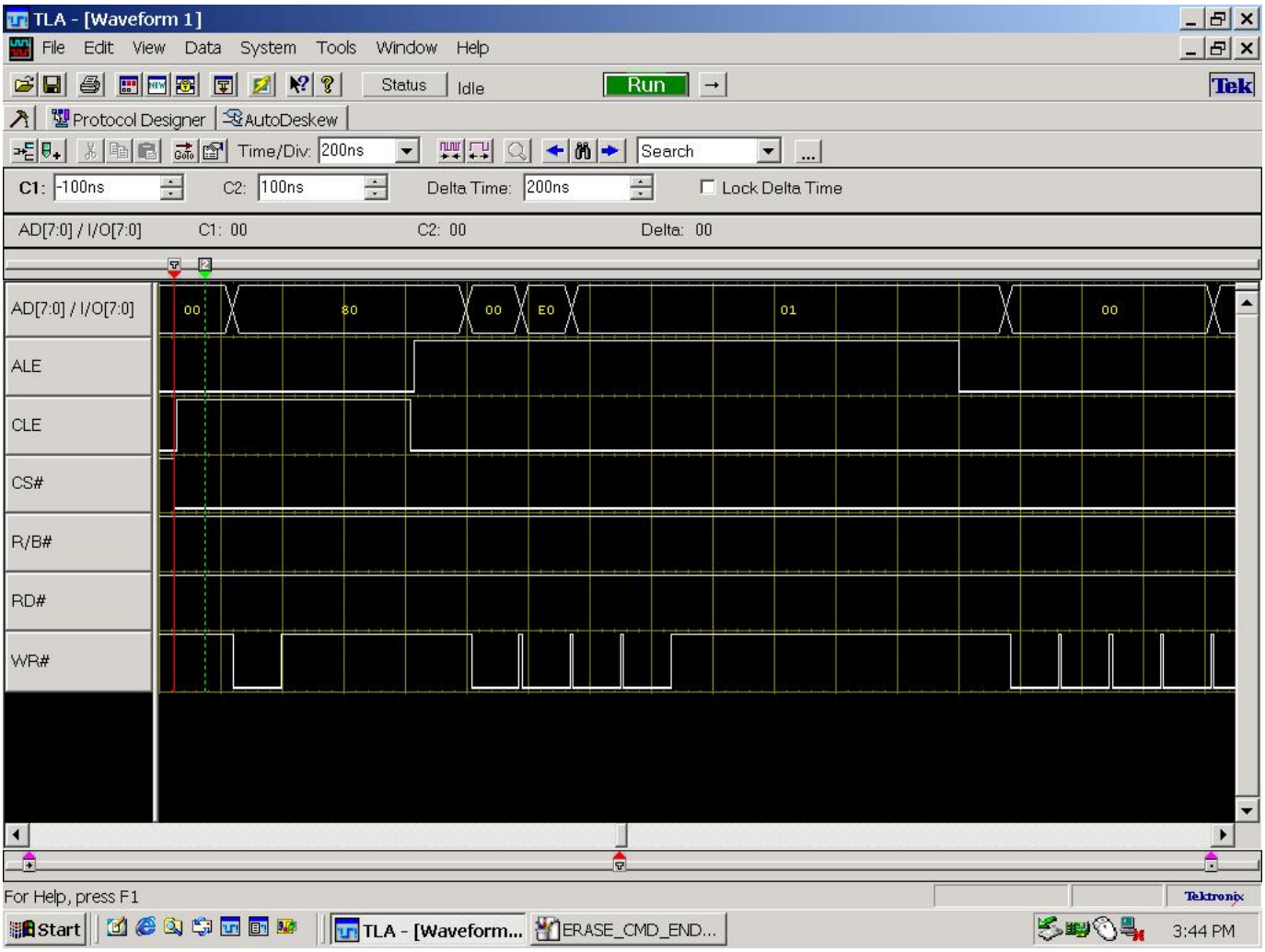

Figure 13. Page Program Command Followed by Data to be Programmed in to Flash

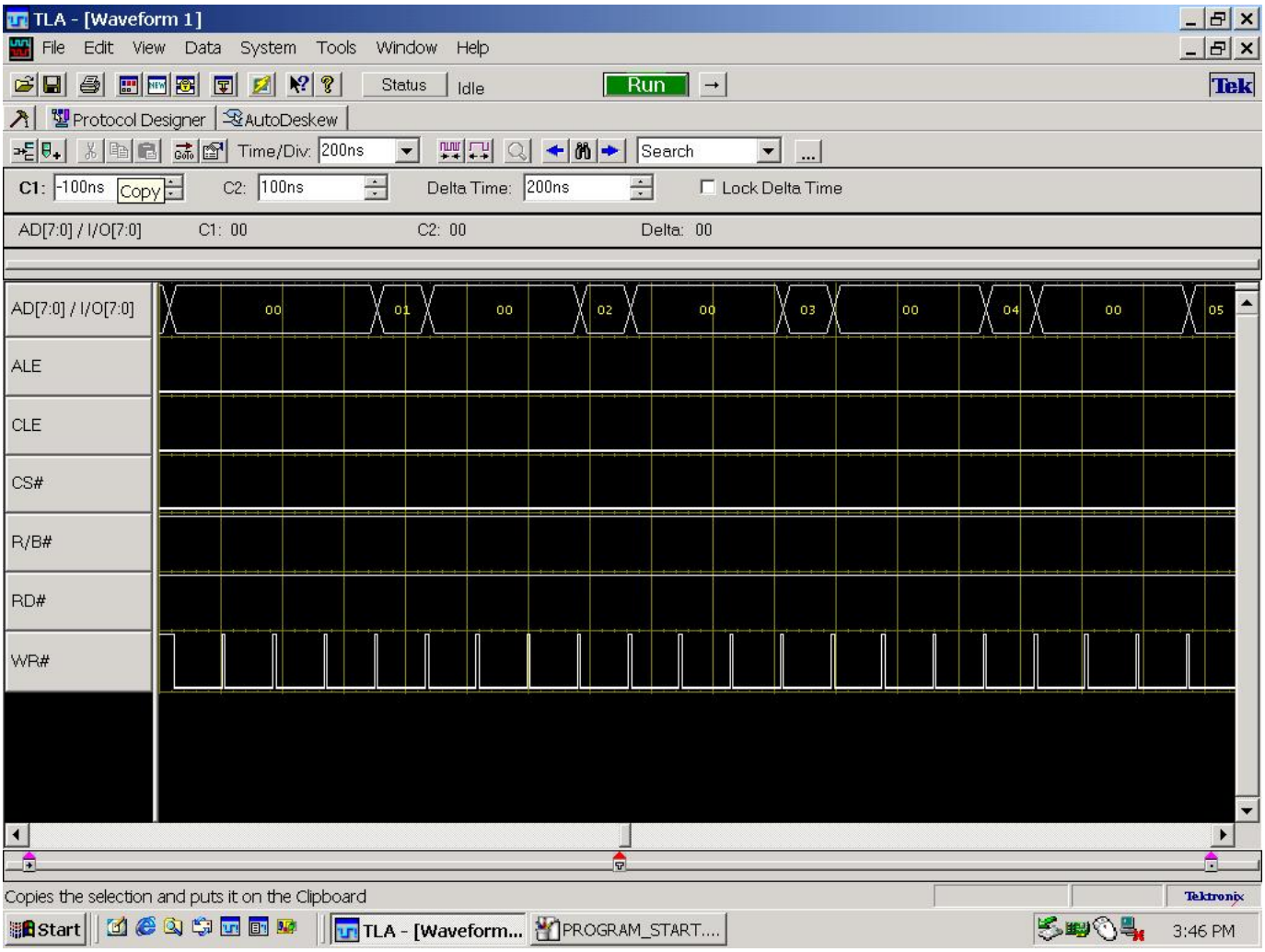

Figure 14. Data Being Written to the Flash Memory Data Register

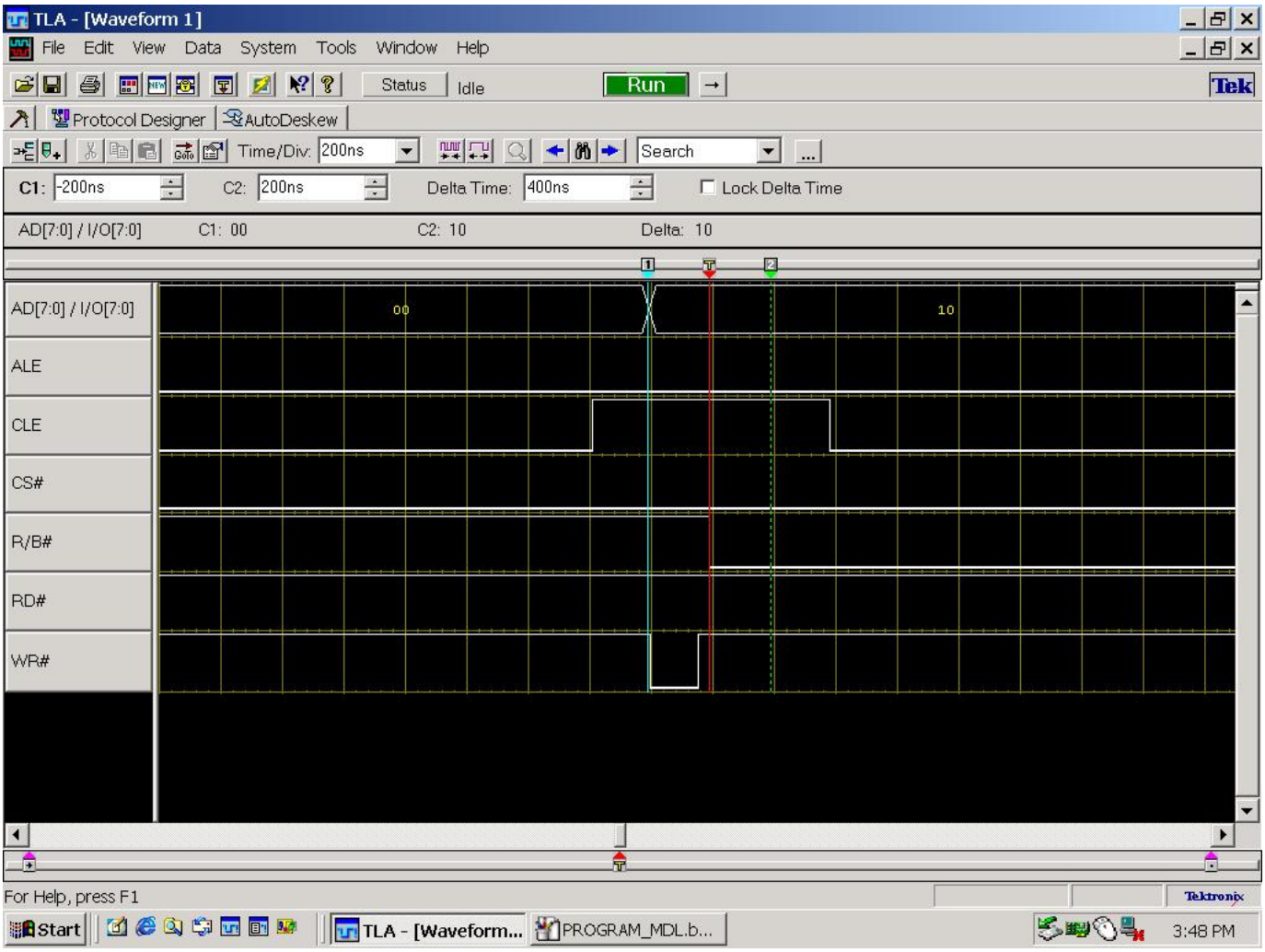

Figure 15. Page Program Confirmation Command Followed by R/B# Going to Logic Low State

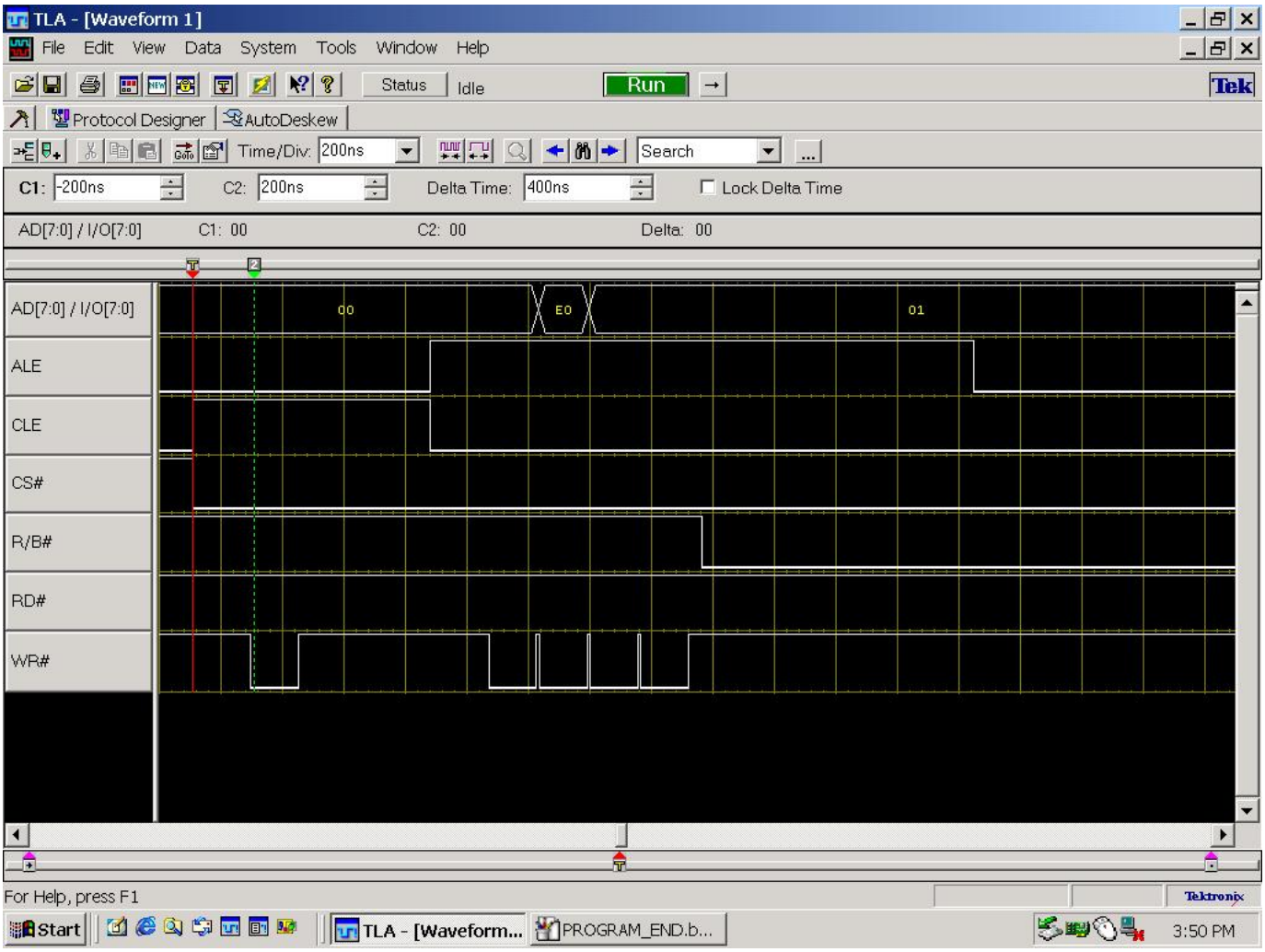

Figure 16. Page Read Command

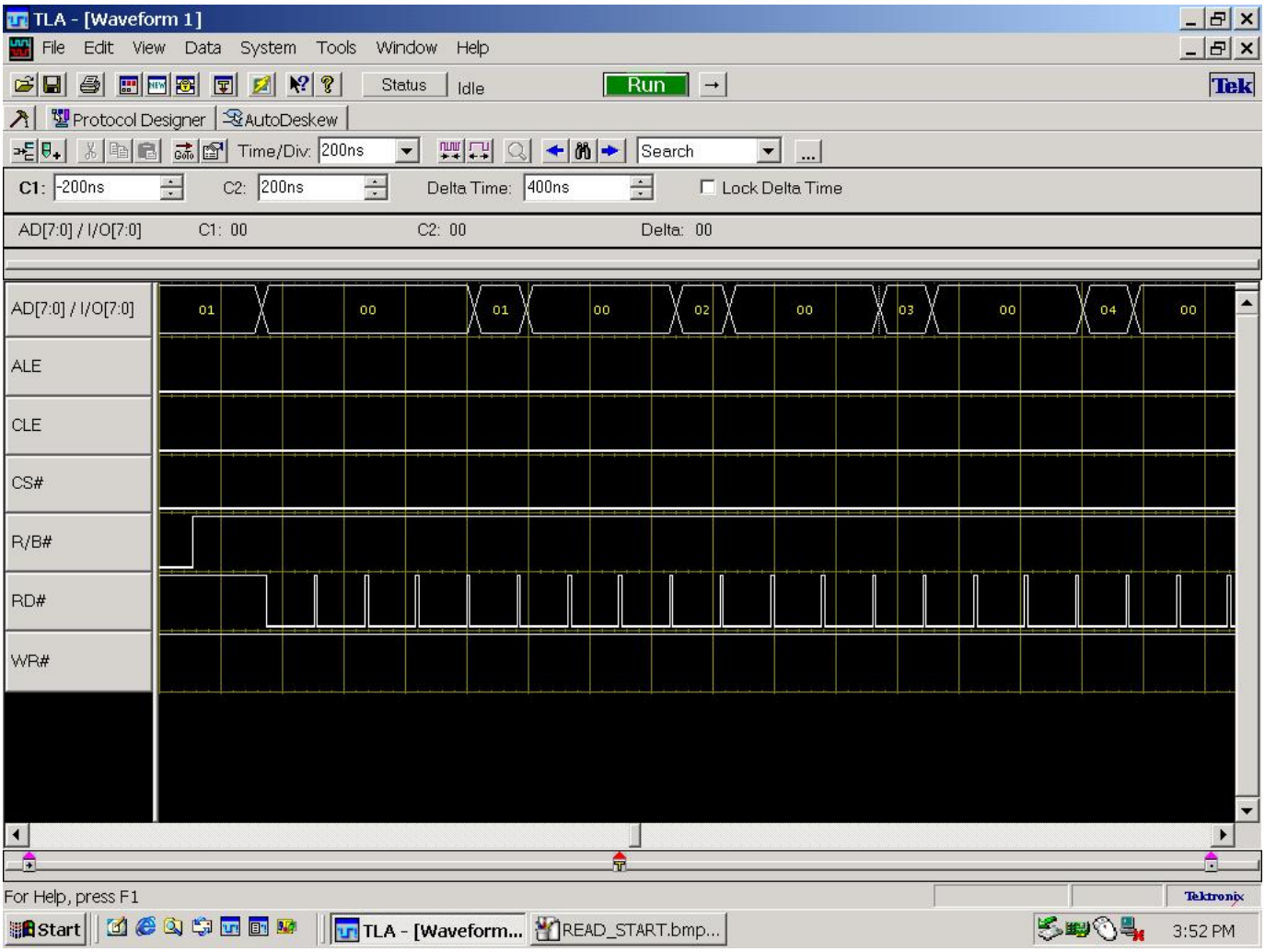

Figure 17. Page Read Operation Completion as Indicated by R/B# Signal Followed by Data Read Operation from Flash Data Register

References

[1] *ADSP-21262 EZ-KIT Lite schematics.* Analog Devices, Inc.

[2] *Preliminary data sheet of K9F1208U0B 64M X 8-bit NAND Flash memory*, Samsung Electronics.

[3] *NAND Flash Applications Design Guide*, Revision 1.0 April 2003, System Solutions from Toshiba America Electronic Components, Inc.

[4] *ADSP-2126x SHARC® DSP Peripheral Manual*, Revision 2.0 January 2004. Analog Devices, Inc.

Document History

Revision	Description
Rev 1 – November 17, 2005 *by Aseem Vasudev Prabhugaonkar*	Initial Release