INTERFACE & CONTROL BLOCK

The Interface & Control Block includes a SPI interface, four control pins, a non-volatile memory array in which the device stores default configuration data, and an array of device registers implemented in Static RAM. This RAM, also called the device registers, configures all hardware within the CDCE62005/CDCE62002.

SPI (Serial Peripheral Interface)

The serial interface of CDCE62005/CDCE62002 is a simple bidirectional SPI interface for writing and reading to and from the device registers. It implements a low speed serial communications link in a master/slave topology in which the CDCE62005/CDCE62002 is a slave. The SPI consists of four signals:

- **SPI_CLK**: Serial Clock (Output from Master) – the CDCE62005/CDCE62002 and the master host clock data in and out on the rising edge of SPI_CLK. Data transitions therefore occur on the falling edge of the clock. (LVCMOS Input Buffer)
- **SPI_MOSI**: Master Output Slave Input (LVCMOS Input Buffer).
- **SPI_MISO**: Master Input Slave Output (Open Drain LVCMOS Buffer)
- **SPI_LE**: Latch Enable (Output from Master). The falling edge of SPI_LE initiates a transfer. If SPI_LE is high, no data transfer can take place. (LVCMOS Input Buffer).

SPI interface Master

The Interface master can be designed using a FPGA or a micro controller. The CDCE62005/CDCE62002 acts as a slave to the SPI master and only supports nonconsecutive read and write command. The SPI clock should start and stop with respect to the SPI_LE signal as shown in Figure 1. SPI_MOSI, SPI_CLK and SPI_LE are generated by the SPI Master. SPI_MISO is generated by the SPI slave the CDCE62005/CDCE62002.

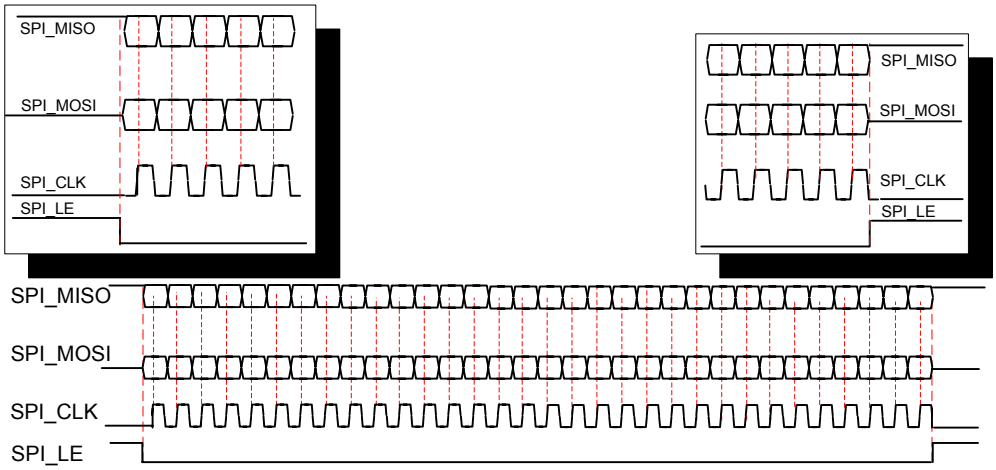

Figure 1 CDCE62005/CDCE62002 SPI Read/Write Command

SPI Consecutive Read/Write Cycles to the CDCE62005/CDCE62002

Figure 2 Illustrates how two consecutive SPI cycles are performed between a SPI Master and the CDCE62005/CDCE62002 SPI Slave.

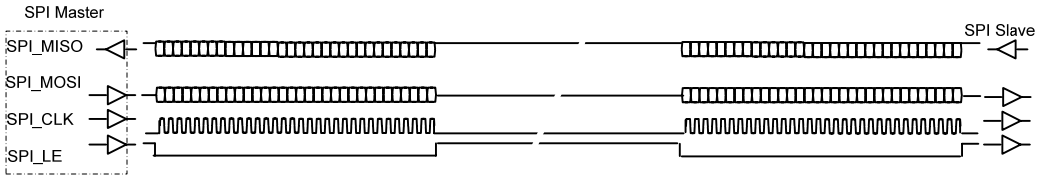

Figure 2 Consecutive Read/Write Cycles.

Writing to the CDCE62005/CDCE62002

Figure 3 illustrates a Write to RAM operation. Notice that the latching of the first data bit in the data stream (Bit 0) occurs on the first rising edge of SPI_CLK after SPI_LE transitions from a high to a low. For the CDCE62005/CDCE62002, data transitions occur on the falling edge of SPI_CLK. A rising edge on SPI_LE signals to the CDCE62005/CDCE62002 that the transmission of the last bit in the stream (Bit 31) has occurred.

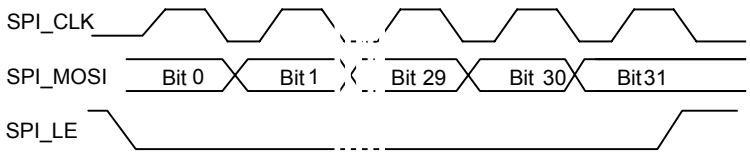

Figure 3 CDCE62005/CDCE62002 SPI Write Operation

Reading from the CDCE62005/CDCE62002

Figure 4 shows how the CDCE62005/CDCE62002 executes a Read Command. The SPI master first issues a Read Command to initiate a data transfer from the CDCE62005/CDCE62002 back to the host (see Table 5)This command specifies the address of the register of interest. By transitioning SPI_LE from a low to a high, the CDCE62005/CDCE62002 resolves the address specified in the appropriate bits of the data field. The host drives SPI_LE low and the CDCE62005/CDCE62002 presents the data present in the register specified in the Read Command on SPI_MISO.

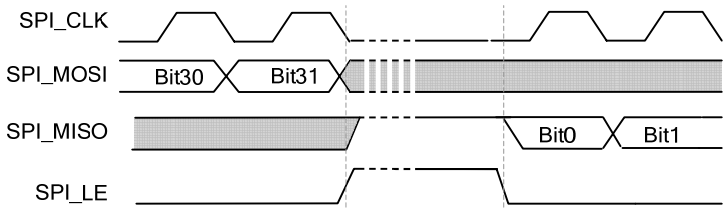

Figure 4 CDCE62005/CDCE62002 SPI Read Operation

Writing to **EEPROM**

After the CDCE62005/CDCE62002 detects a power-up and completes a reset cycle, the device copies the contents of the on-board EEPROM into the Device Registers. (SPI_LE signal has to be HIGH in order for the EEPROM to load correctly during the rising edge of Power_Down signal).

The host issues one of two special commands shown in Table 5 to copy the contents of Device Registers 0 through 7 (a total of 224 bits) into EERPOM. They include:

- Copy RAM to EEPROM – Unlock, Execution of this command can happen many times.
- Copy RAM to EEPROM – Lock: Execution of this command can happen only once; after which the EEPROM is **permanently locked**.

After either command is initiated, power must remain stable and the host must not access the CDCE62005/CDCE62002 for at least 50 ms to allow the EEPROM to complete the write cycle and to avoid the possibility of EEPROM corruption.

SPI CONTROL INTERFACE TIMING

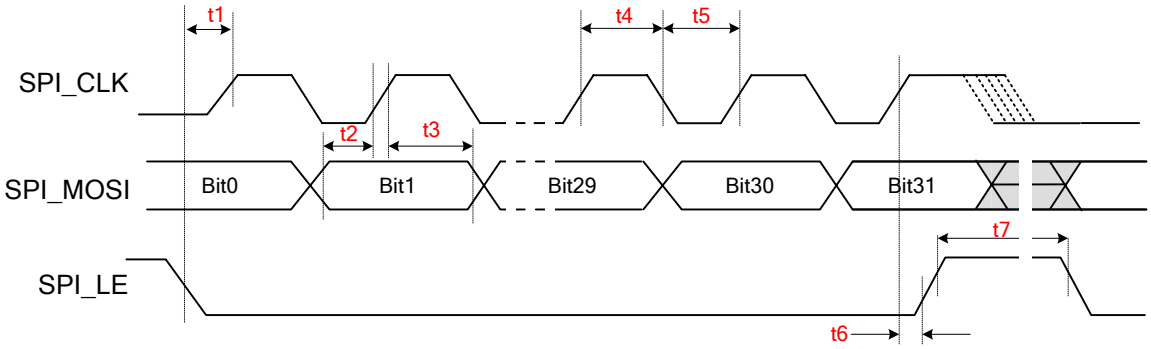

Figure 5 Timing Diagram for SPI Write Command

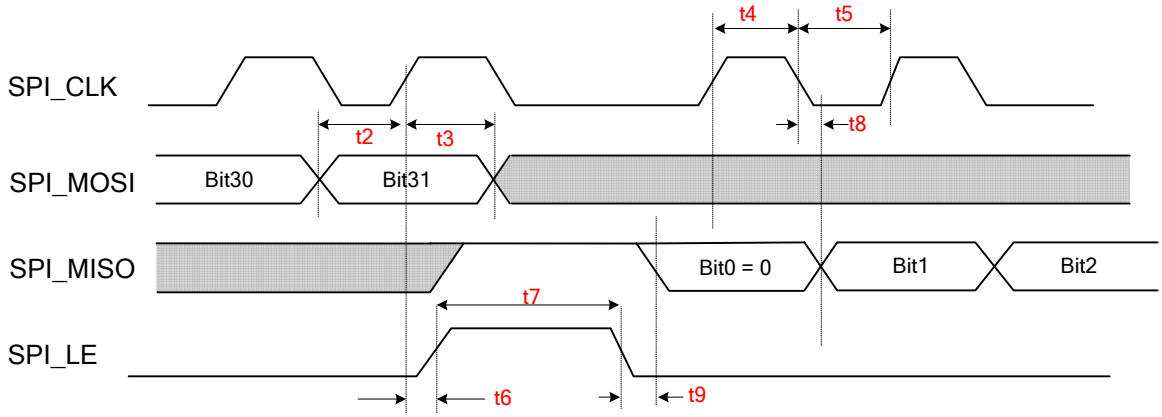

Figure 6 Timing Diagram for SPI Read Command

Table 1 SPI Bus Timing Characteristics

SPI Bus Timings					
	Parameter	**MIN**	**TYP**	**MAX**	**UNIT**
f_{Clock}	Clock Frequency for the SPI_CLK			20	MHz
t_1	SPI_LE to SPI_CLK setup time	10			ns
t_2	SPI_MOSI to SPI_CLK setup time	10			ns
t_3	SPI_MOSI to SPI_CLK hold time	10			ns
t_4	SPI_CLK high duration	25			ns
t_5	SPI_CLK low duration	25			ns
t_6	SPI_CLK to SPI_LE Hold time	10			ns
t_7	SPI_LE Pulse Width	20			ns
t_8	SPI_CLK to MISO data valid			10	ns
t_9	SPI_LE to SPI_MISO Data Valid			10	ns

Timing Capture of SPI transactions on the CDCE62005/CDCE62002 EVM

This section shows captured transactions between a µC and the CDCE62005/CDCE62002 on the TI CDCE62005/CDCE62002 EVM using the CDCE62005/CDCE62002 GUI.

Read-Write Transaction: The transaction will first of all read Register 6, which is at the time holding the value 01h. The register is then written with the value 06h and through a second read transaction it will be confirmed whether or not the writing was successful.

EEPROM write transaction: writing to EEPROM locked and unlocked will be demonstrated furthermore.

Reading a SPI register

The SPI Registers are read by a read command from µC to the CDCE62005/CDCE62002 followed by a response from the CDCE62005/CDCE62002 back to the µC. The example here shows how to read from SPI register 6.

Read command from µC on MOSI: `0x0000006E`

Response from CDCE62005/CDCE62002 on MISO: `0x00000106`

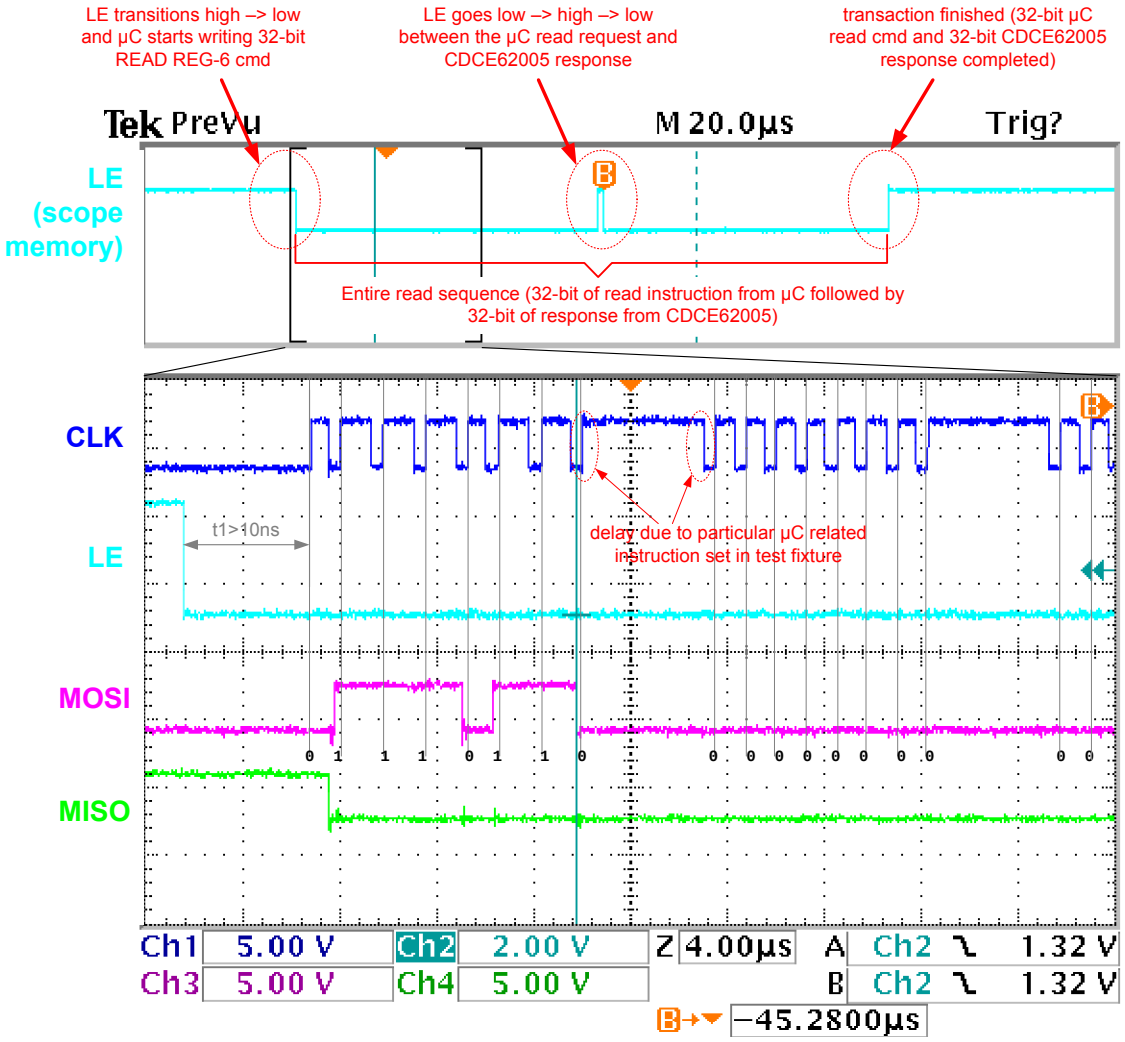

Figure 7: µC generates Read command (first 18 bits shown)

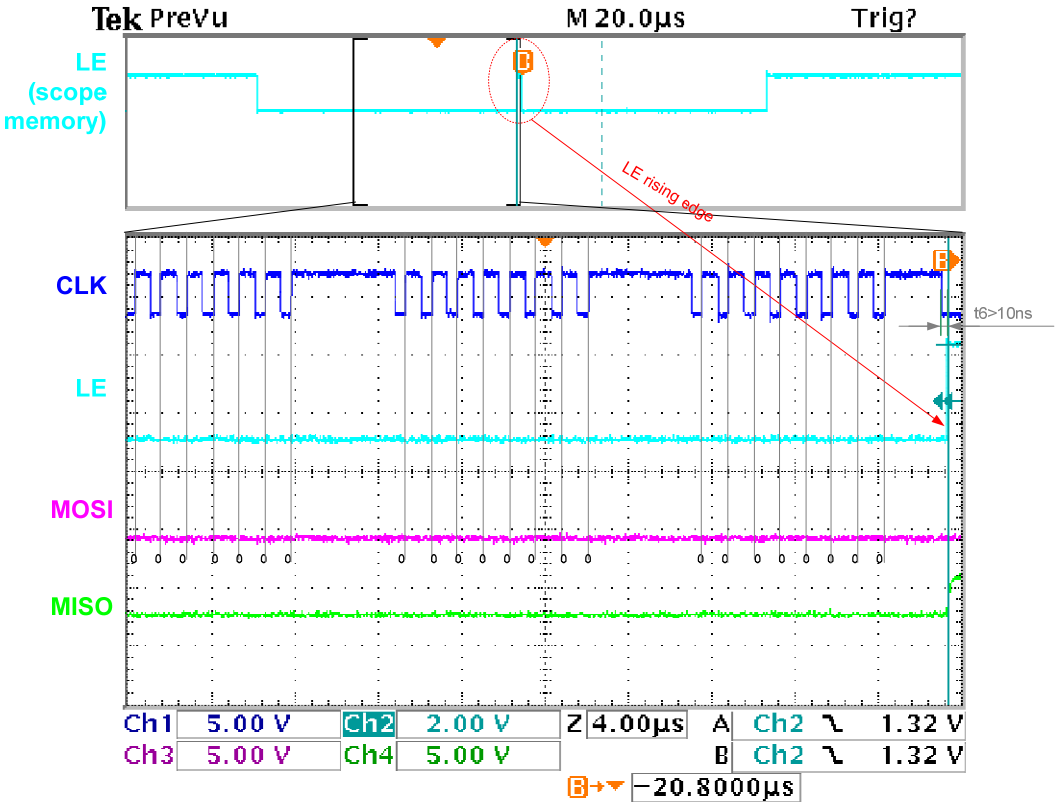

Figure 8: read continued (part 2, showing LE going back low after first 32 bit from µC to CDCE62005/CDCE62002)

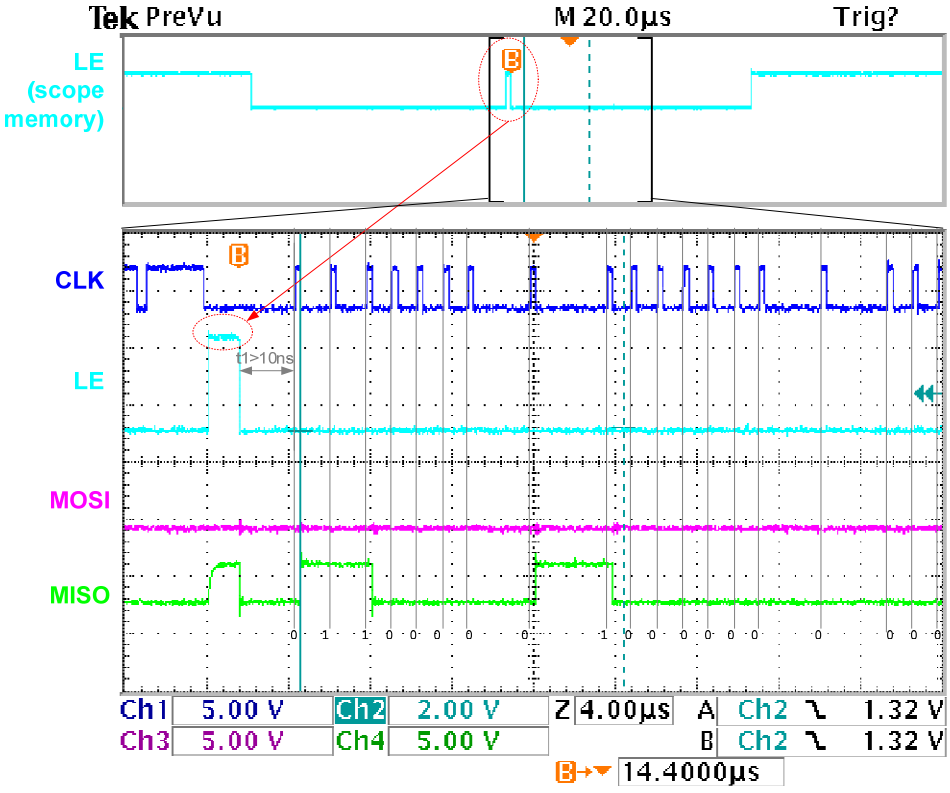

Figure 9: CDCE62005/CDCE62002 responds to REG-6 reading command with the REG-6 content, which is 00000106h

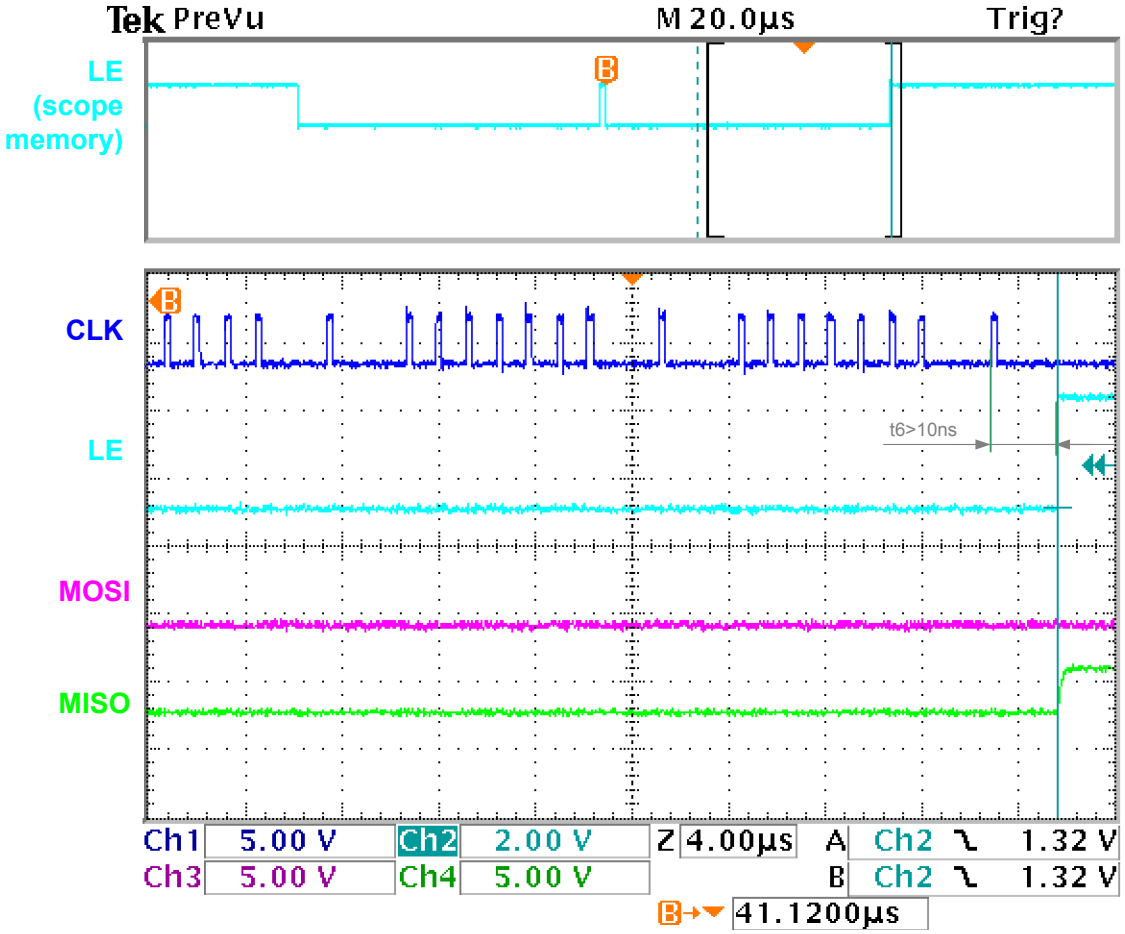

Figure 10: Read response continued (from previous figure); the LE going high completes the response by the CDCE62005/CDCE62002

Write a RAM register

The following figure shows how to write the a RAM register. A consecutive read of the same RAM register verifies that the value was indeed written properly.

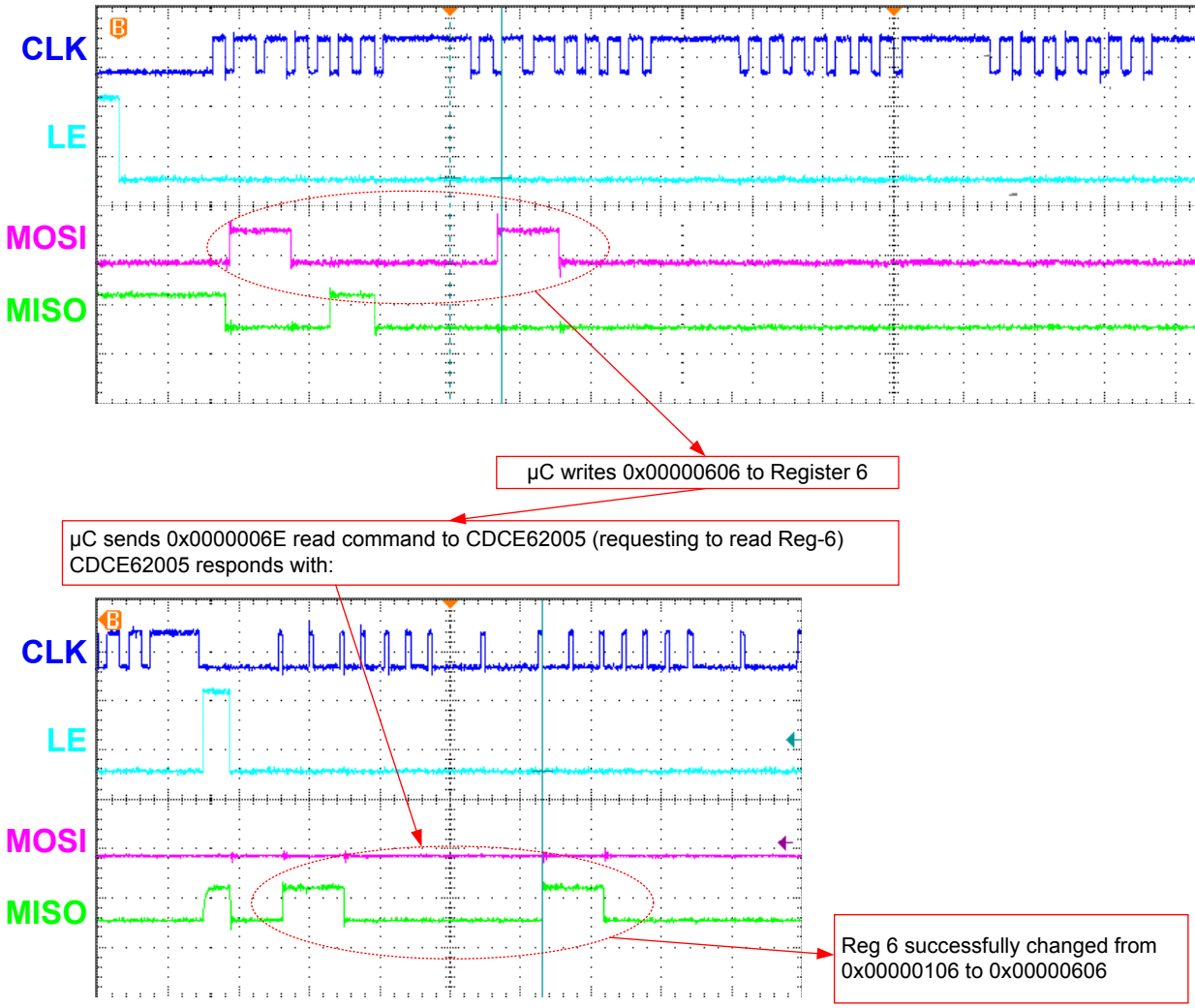

Figure 11: Example of writing a new register value to Register 6

EEPROM write

The following transaction shows how to transfer the register space in the RAM into the EEPROM, so that it is permanently available, even after the device becomes powered down. Two options exist: The EEPROM can either be written in a locked or in a unlocked mode. Writing the EEPROM with the RAM EEPROM LOCK command will lock the EEPROM permanently. It can not be overwritten again. Writing to the EEPROM with the RAM EEPROM UNLOCKED command keeps the EEPROM unlocked, so that in the future the EEPROM can be changed again.

```
RAM EEPROM LOCK command:       0x0000A03F
RAM EEPROM UNLOCK command:     0x0000001F
```

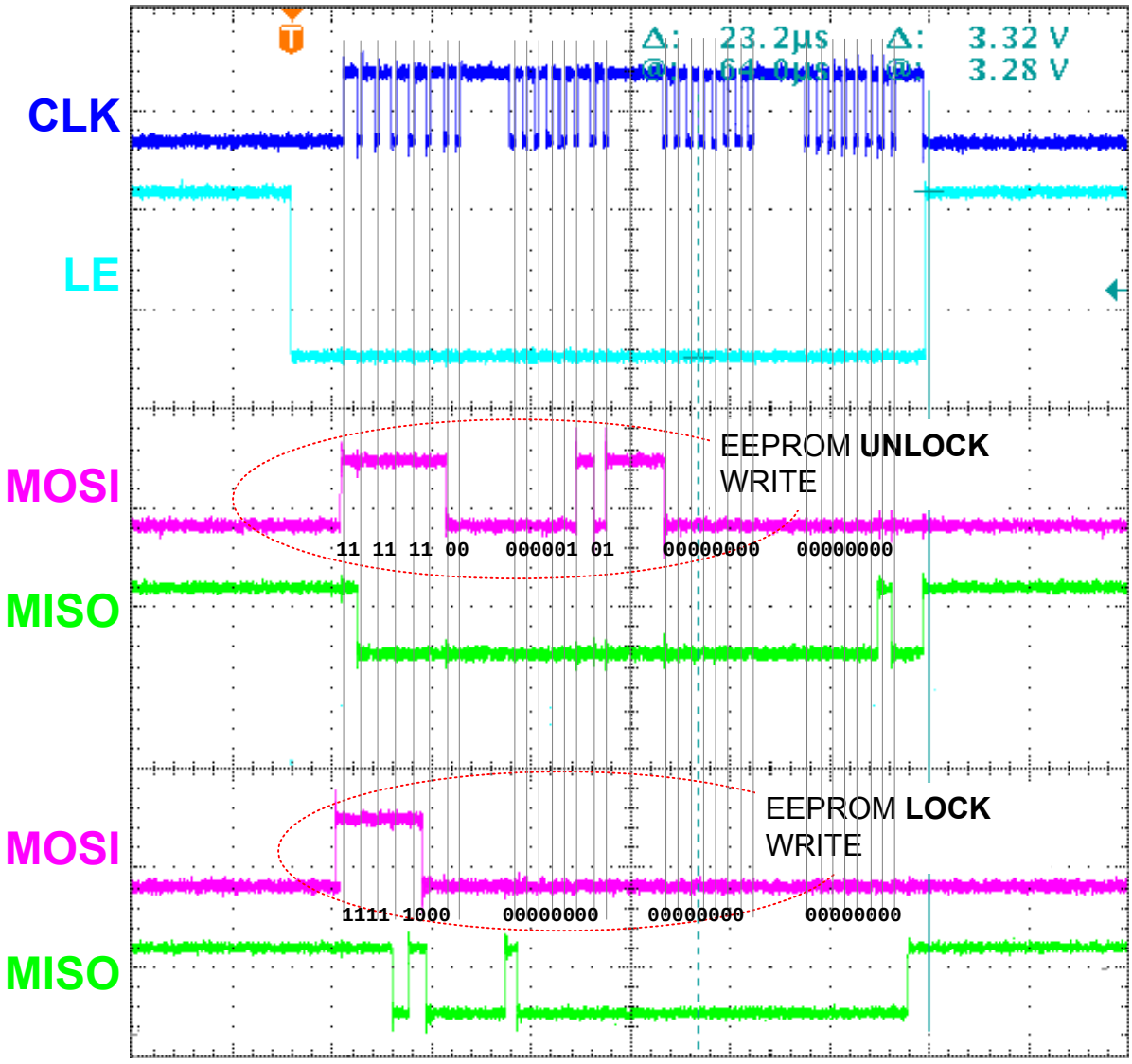

Figure 12: EEPROM Write command: the top MOSI command shows how to transfer the current register space into EEPROM and lock the EEPROM; the bottom MISO command shows the same transaction but leaving the EEPROM unlocked