Programming Multiple Devices Using a Single SPI Bus

Application Note 87

Summary

Many pSemi devices can be programmed using a three-wire serial peripheral interface (SPI). This application note documents three main ways in which multiple devices can be programmed using the same SPI interface:

- Multiple latch enable (LE) lines
- Addressable devices
- Cascaded data lines

SPI Operation

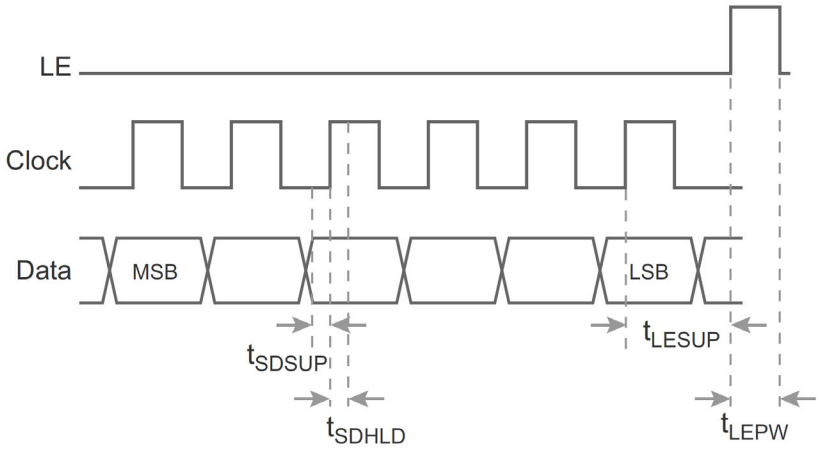
The three SPI lines are:

- CLK (clock; also called CLK, SCL)—Provides clock edges for the device's shift register.
- DATA (also called SI, SDA)—This is the data information. Refer to the device's datasheet for the data content. Depending on the device, the data may contain address information.
- LE (latch enable; also called SEN, LE, SS (slave select))—This is pulsed after the CLK and DATA have finished loading into the shift-register. It applies the contents of the shift-register to the hardware.

Figure 1 shows some example minimum timings:

- t_{LESUP}—LE set-up time after last clock rising edge
- t_{LEPW}—LE minimum pulse width
- t_{SDSUP}—Serial data set-up time before clock rising edge
- t_{SDHLD}—Serial data hold time after clock rising edge

Figure 1 ▪ SPI Timing Example Using PE4314



Buffering

Some devices (for example, the PE44820) have buffered outputs.

- CLK => CLKO
- DATA => SDO1 (serial data out 1)
- LE => LEO

These are to improve with fan-out current issues, but check the propagation delays as they might reduce the maximum CLK rate.

Examples

This paper provides three examples for communicating with multiple devices, using three pSemi devices.

- **Example 1**—Multiple LE lines programming method, using PE4314
- **Example 2**—Addressable programming method, using PE43712
- **Example 3**—Cascaded programming method, using PE44820

Example 1: Multiple LE Lines Programming Method, Using PE4314

PE4314 is a DSA with a 6-bit shift-register. It has no address field and no output for the last bit in the shift register. For programming multiple devices, the only option is to use multiple LE lines.

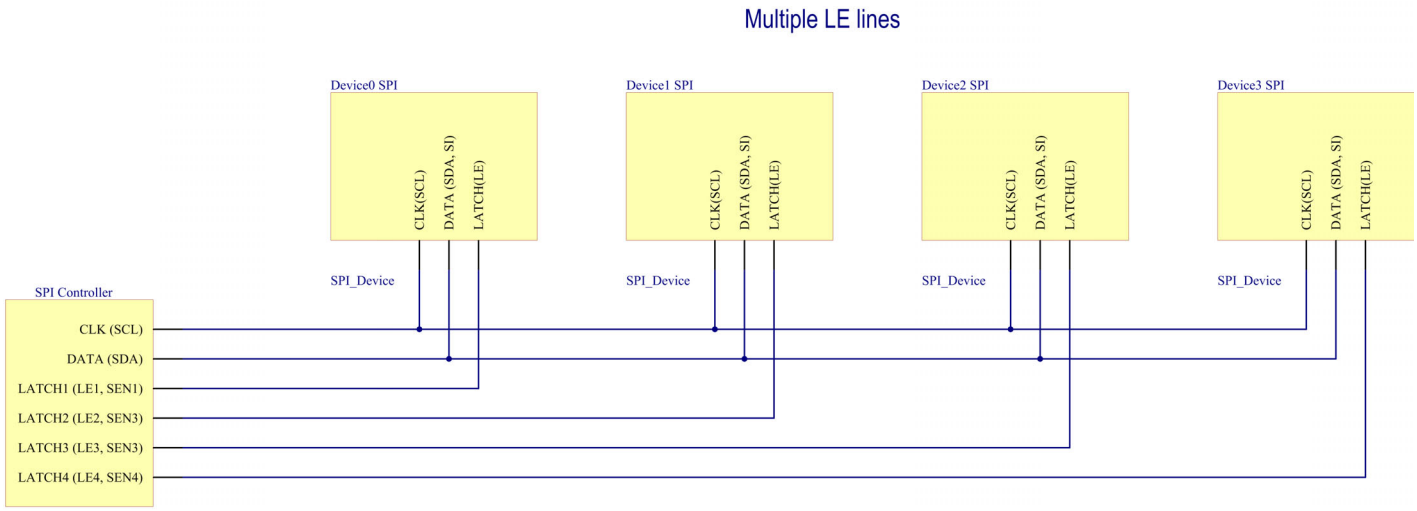
Note: If a common clock is used, the data will be clocked into the shift register of all the devices, not only the device being latched. It is important to hold the latch of the other devices low so that erroneous data does not get latched into the other devices.

Details—PE4314 Multiple LE Lines

• For use where addressing is unavailable or undesirable.

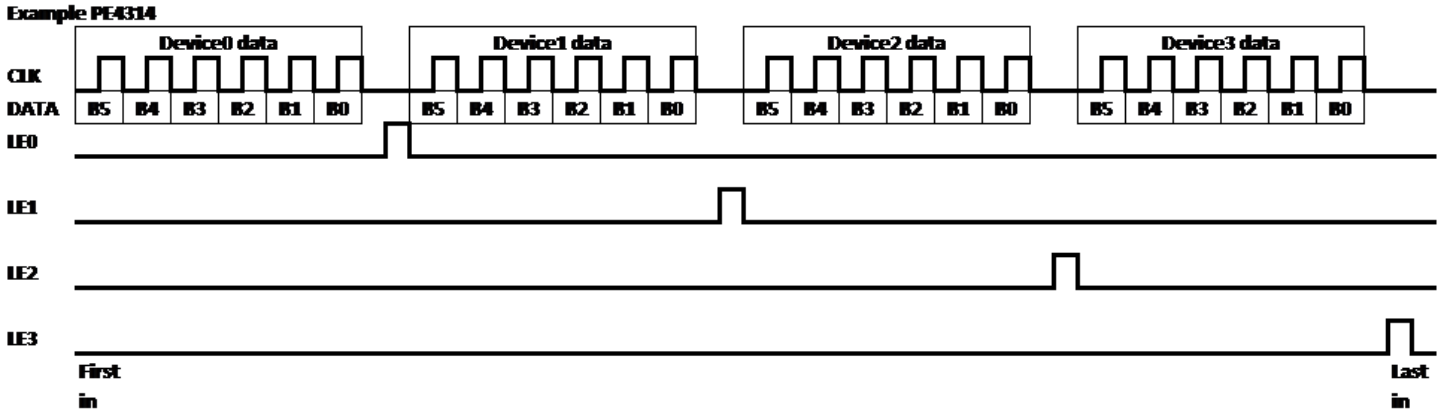
• Requires one LE (EN, SEN) per device.

Figure 2 ▪ PE4314 Multiple LE Lines Method



- Data contains no address data, or addresses data set to zero.
- Optional robust approach would be to include addressing.
- Devices update asynchronously on each LE event.

Figure 3 ▪ Timing Example Using PE4314



Example 2: Addressable Programming Method, Using PE43712

PE43712 is a DSA with a 16-bit shift-register. It supports addressing but has no output for the last bit in the shift register. For programming multiple devices, the user may choose one of the following:

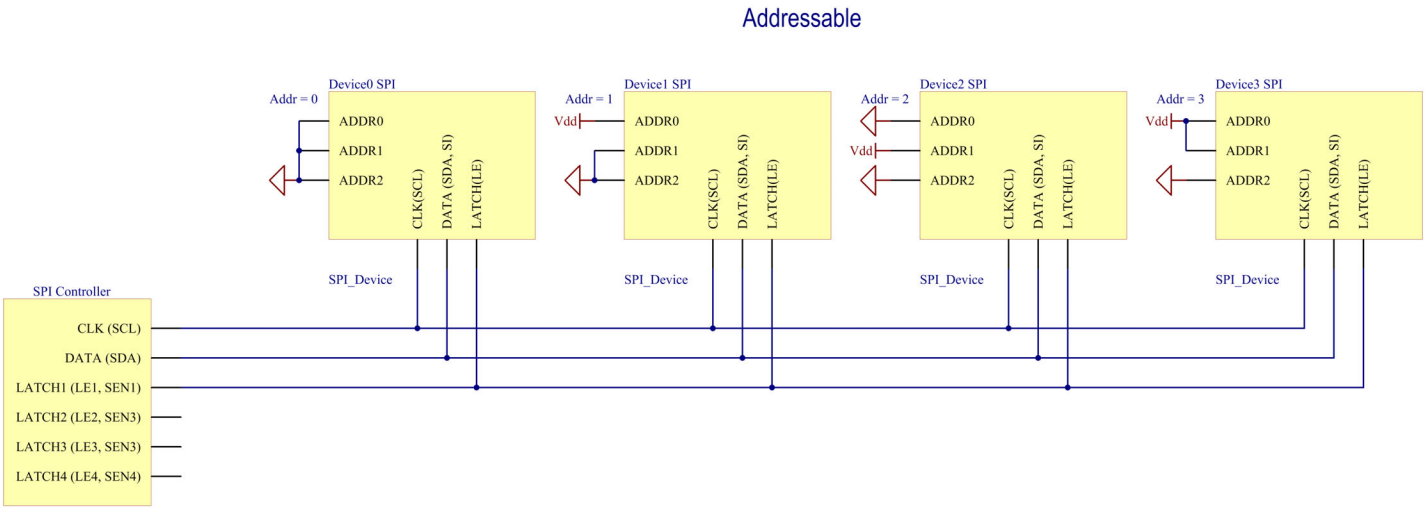
- Multiple LE lines with setting addr=0
- Single LE line with addressing. (**This is the example used below**.)
- Multiple LE lines with addressing. (This is the most robust case, but requires multiple LE lines.)

With the common clock, the shift register of each device will be updated; but with the addressing, the LE signal will be ignored if the addresses do not match.

Details—PE43712 Addressable

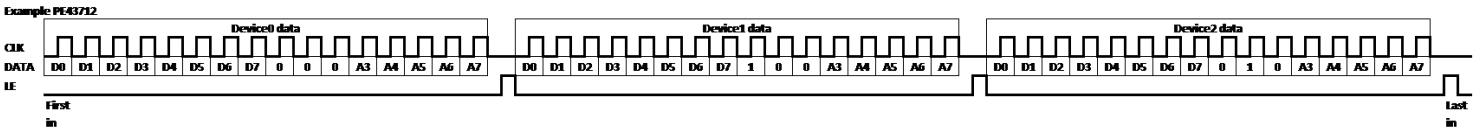
- Where addressing is available
- Requires only one LE (EN, SEN) line connected to all devices

Figure 4 ▪ PE43712 Addressable Method



- Data contains address information.
- LE event ignored unless the address contained in the data corresponds to the device's set address.

Figure 5 ▪ Timing Example Using PE43712



Example 3: Cascade Programming Method, Using PE44820

PE44820 is a digital phase shifter with a 13-bit shift-register. It supports addressing and has two pins: SDO1 and SDO2. SDO1 is a buffered output of the serial data input. SDO2 is a buffered output of the last bit of the internal shift register. For programming multiple devices, multiple options are available:

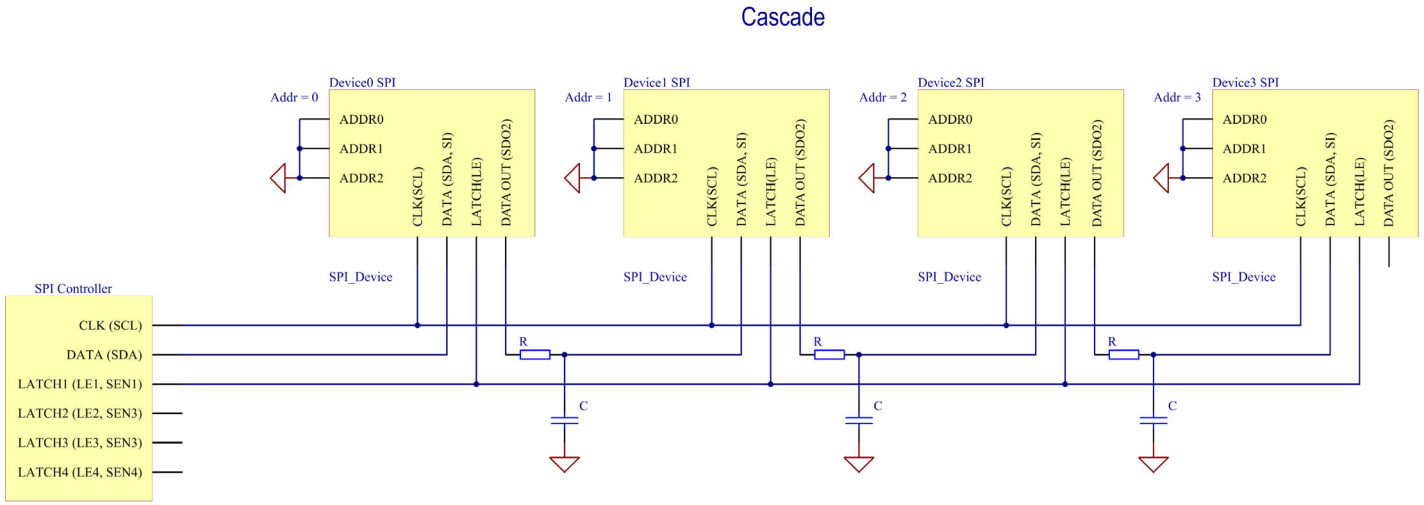
- Multiple LE lines, setting addr=0
- Single LE line with addressing
- Multiple LE lines with addressing (robust)
- Cascaded programming, setting addr=0. (**This is the example used below**.)
- Cascaded programming with addressing. (This is the most robust case, but it is harder to program.)
- Can also include parallel and cascaded programming

Details—PE44820 Cascaded

- For devices where SDO2 reflects the last bit in the shift register. SDO2 updates on a rising CLK edge.
- Addressing is optional—can use addr=0 for all devices.
- Requires only one LE (EN, SEN) line connected to all devices.

- RC time delay required on each SDO2 line to 'hold' the last bit out of the preceding device so it is passed to the next device.

Figure 6 ▪ PE44820 Cascaded Method



- R should be chosen carefully. If R is too large, it will not provide enough drive for the data input. If R is too small, it will cause current spikes in the supply. Recommend R should be around ~1k Ohms.
- Suggested RC time-constant should be greater than hold time (~10ns) but less than CLK period (e.g. R = 1k Ohms, C = 15 pF, Tau = R*C = 15 ns).
- Data contains address information.
- Devices update synchronously on a single LE event.

Note: For a cascaded scheme, it might be tempting to delay or invert the CLK on alternate devices to ensure reliable data transfer. Do not do this because, although data arrives reliably, it is impossible to program each device with unique data. This is because with a delayed CLK, for some devices in the cascade, the first data bit will always equal the last data bit of the previous device.

Figure 7 ▪ Timing Example Using PE44820 Cascaded, with Common Address of 000 Set

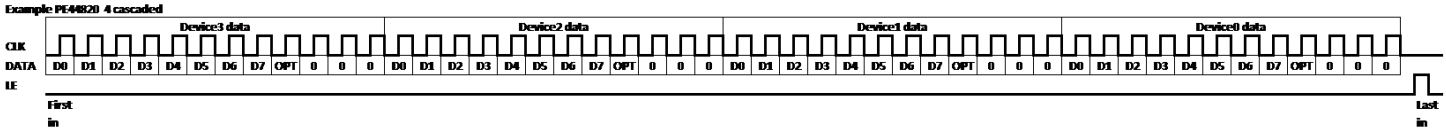
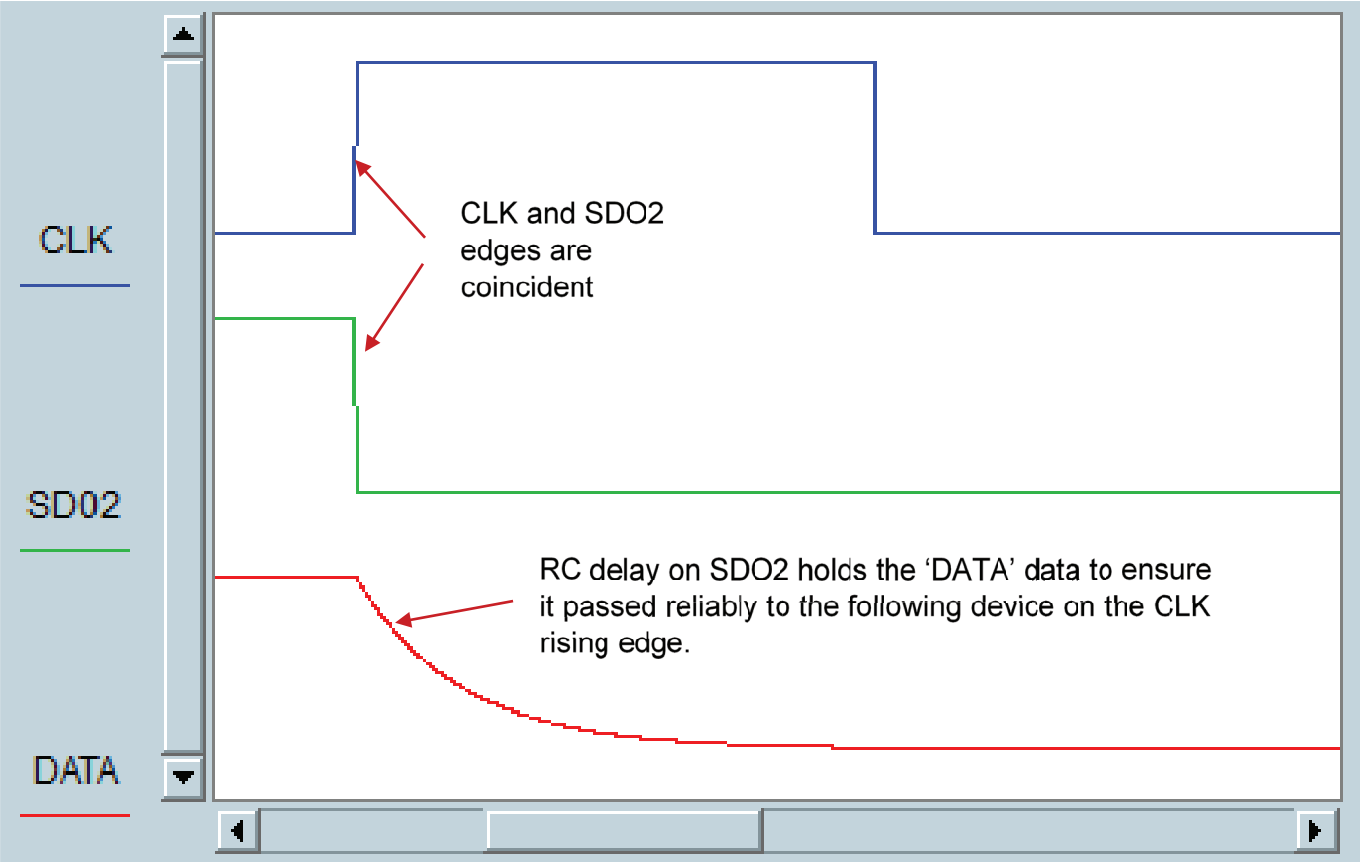


Figure 8 ▪ PE44820 Timing Plot Showing SDO2 and CLK Timings



Using the PE44820 with 16-Bit Register Programming

In normal operation, the PE44820 requires 13-bit register programming consisting of eight data bits, one OPT bit, and four address bits.

Figure 9 ▪ 13-bit Register

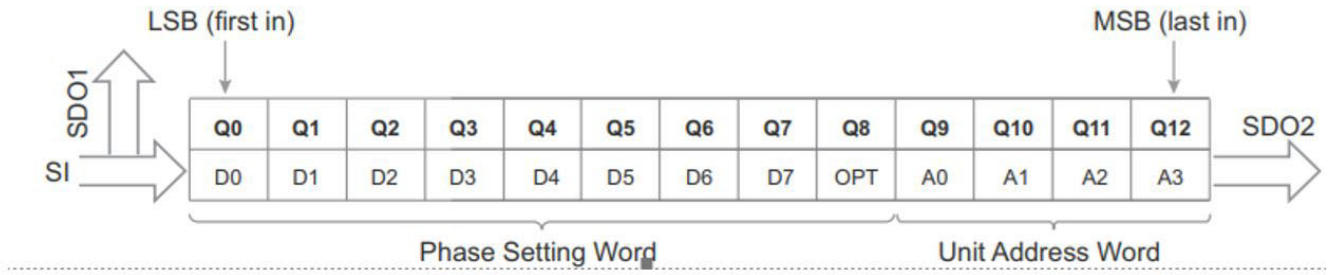
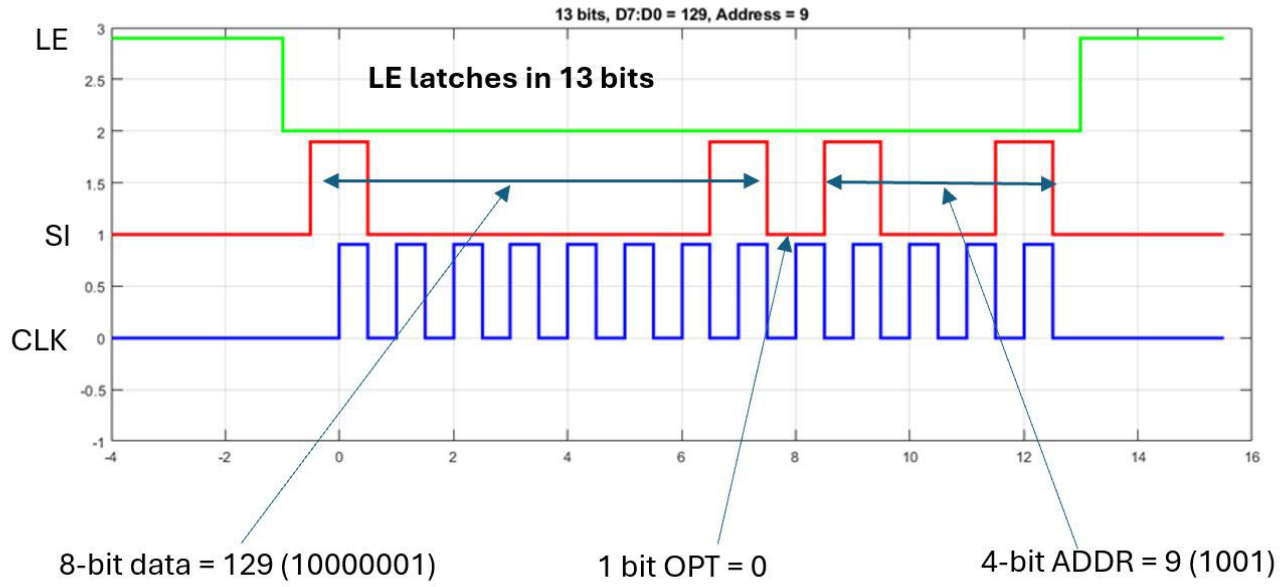
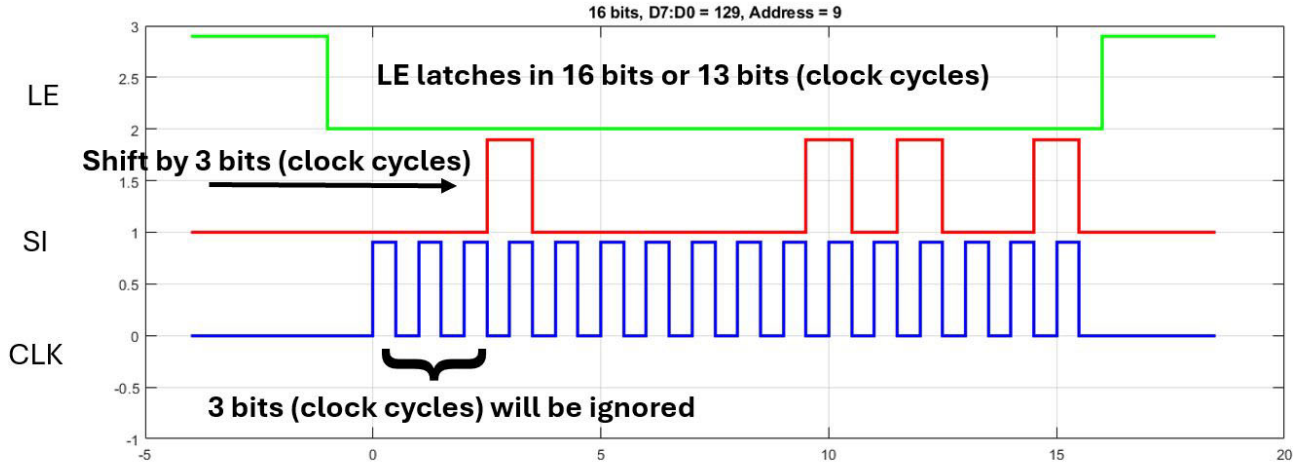


Figure 10 ▪ Programming the PE44820 13-bit Register



For 16-bit programming of the 13-bit PE44820, shift the 13-bit programming word by three bits, adding the three bits to the LSB (first in) side of the programming word as shown in **Figure 11**, and the first three bits will be ignored.

Figure 11 ▪ Programming the PE44820 Using a 16-bit Register



Conclusion

This application note has demonstrated the three main methods for communicating with multiple devices using a single SPI bus. For each main method, circuits and timing examples are provided.

Sales Contact

For additional information, contact Sales at sales@psemi.com.

Disclaimers

The information in this document is believed to be reliable. However, pSemi assumes no liability for the use of this information. Use shall be entirely at the user's own risk. No patent rights or licenses to any circuits described in this document are implied or granted to any third party. pSemi's products are not designed or intended for use in devices or systems intended for surgical implant, or in other applications intended to support or sustain life, or in any application in which the failure of the pSemi product could create a situation in which personal injury or death might occur. pSemi assumes no liability for damages, including consequential or incidental damages, arising out of the use of its products in such applications.

Patent Statement

pSemi products are protected under one or more of the following U.S. patents: patents.psemi.com

Copyright and Trademark