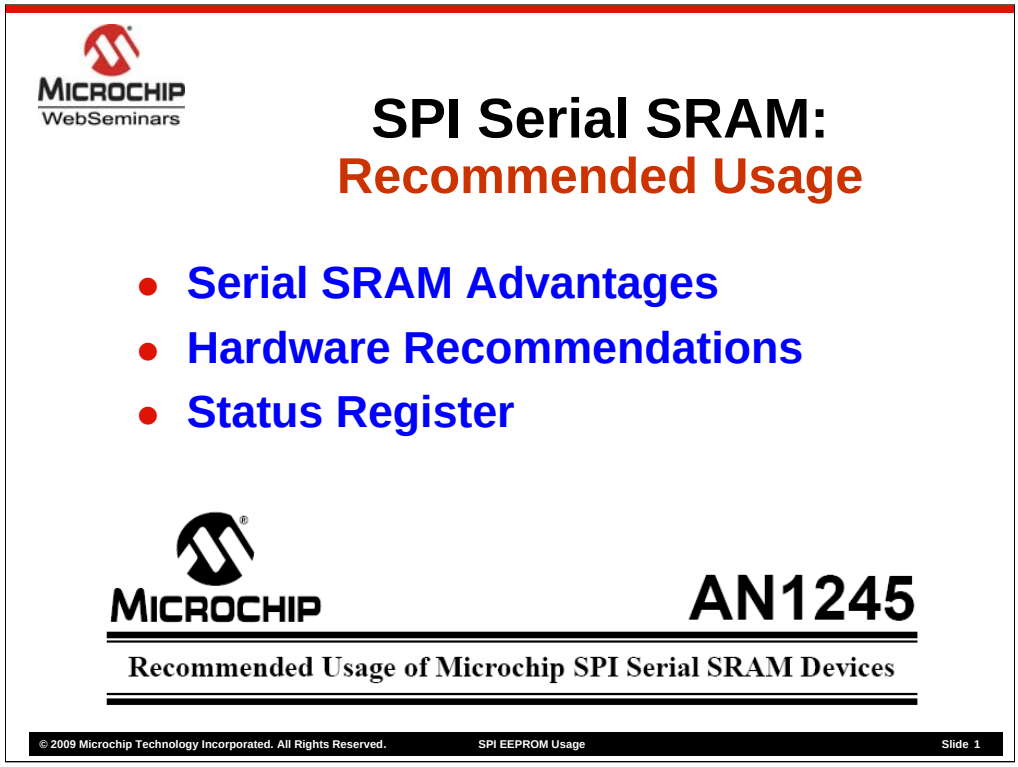


Hi, my name is Barry Blixt, marketing manager for Microchip memory products. Welcome to this short web seminar in which I will discuss some recommended usage practices for serial SRAM.

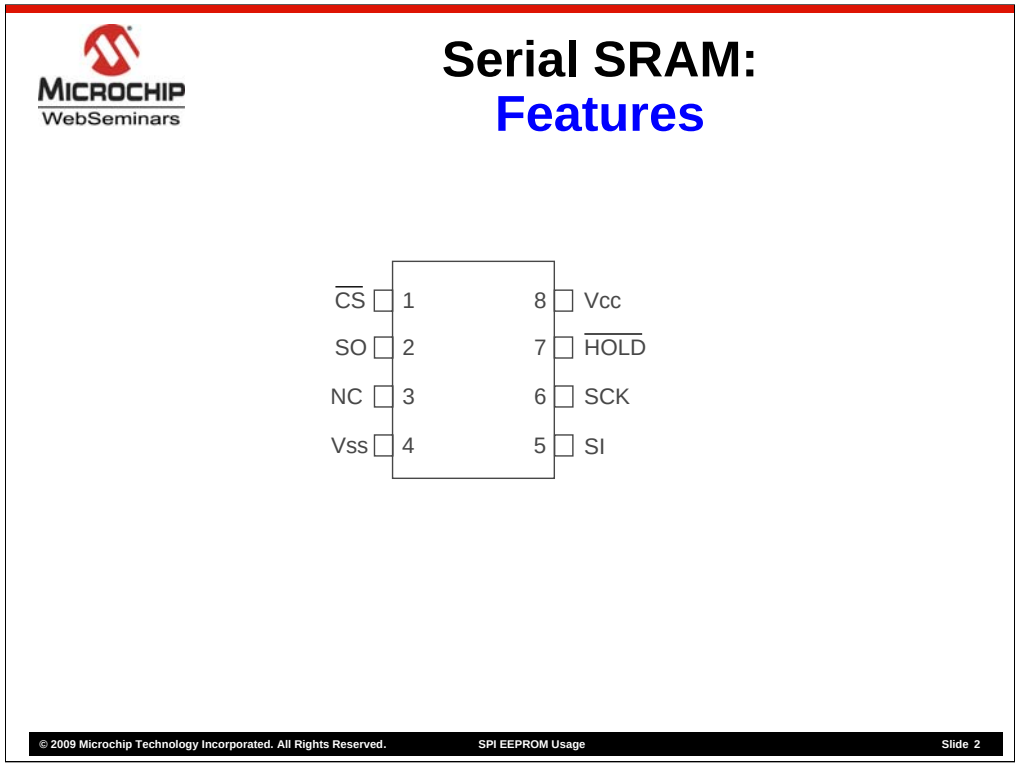
These are some issues that I'll address in this presentation:

•First, we'll take a quick look at the advantages of serial SRAM products.

•Next, we'll go over some hardware and connection recommendations.

•Lastly, we'll take a quick look at the status register and its operation.

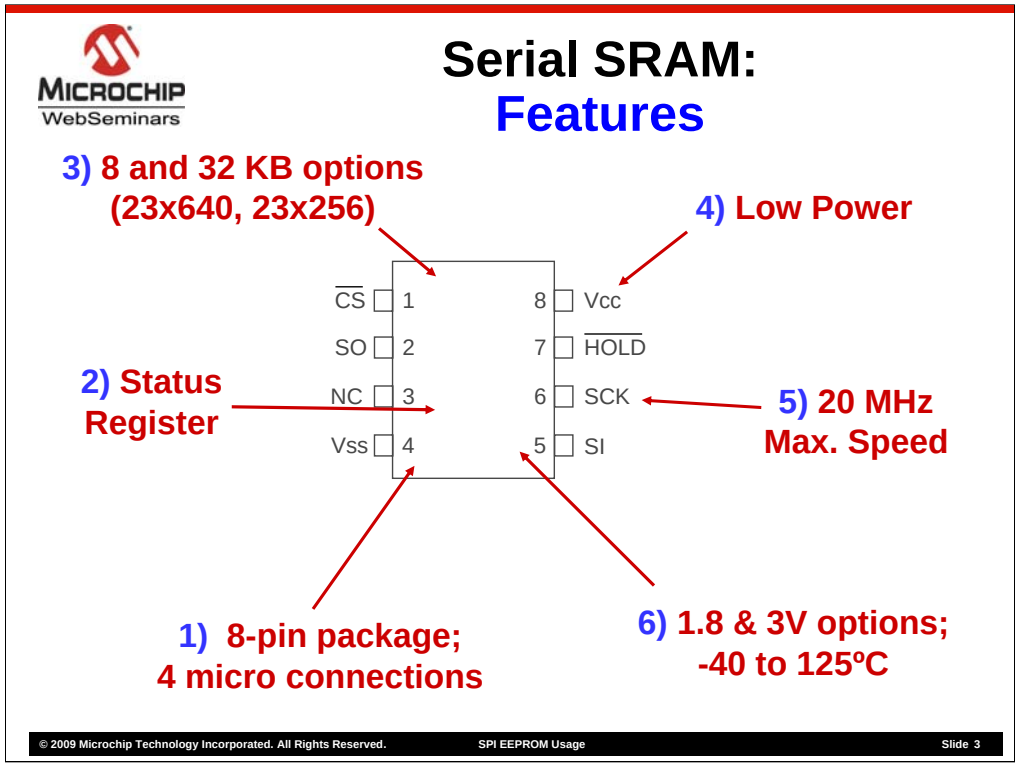
Most of the information in this seminar is taken from Microchip data sheets as well as our application note AN1245 titled "Recommended Usage of Microchip SPI Serial SRAM devices."



Serial SRAM devices use the SPI interface, which is a very common and easy-to-use protocol. Many microcontrollers have SPI hardware ports that make it very easy to add an external memory device. These features help to shorten the design cycle, especially if you are adding SRAM to an existing application.

Here you can see the device pinout. Pin 1 is chip select, which controls communication to the chip. Pin 2 is data out. Pin 3 is not connected. Pin 4 is ground. Pin 5 is data in. Pin 6 is the clock. Pin 7 is hold, and Pin 8 is supply voltage. Note that the Chip Select and HOLD pins are both active low.

Let’s look at some advantages of serial SRAM products.



First, the devices come in small, 8-lead packages which only require 4 connections to a microcontroller: Chip Select, Clock, Data Out and Data In. This contrasts with parallel memory technologies that typically come in 28 to 44-pin packages and require many more connections.

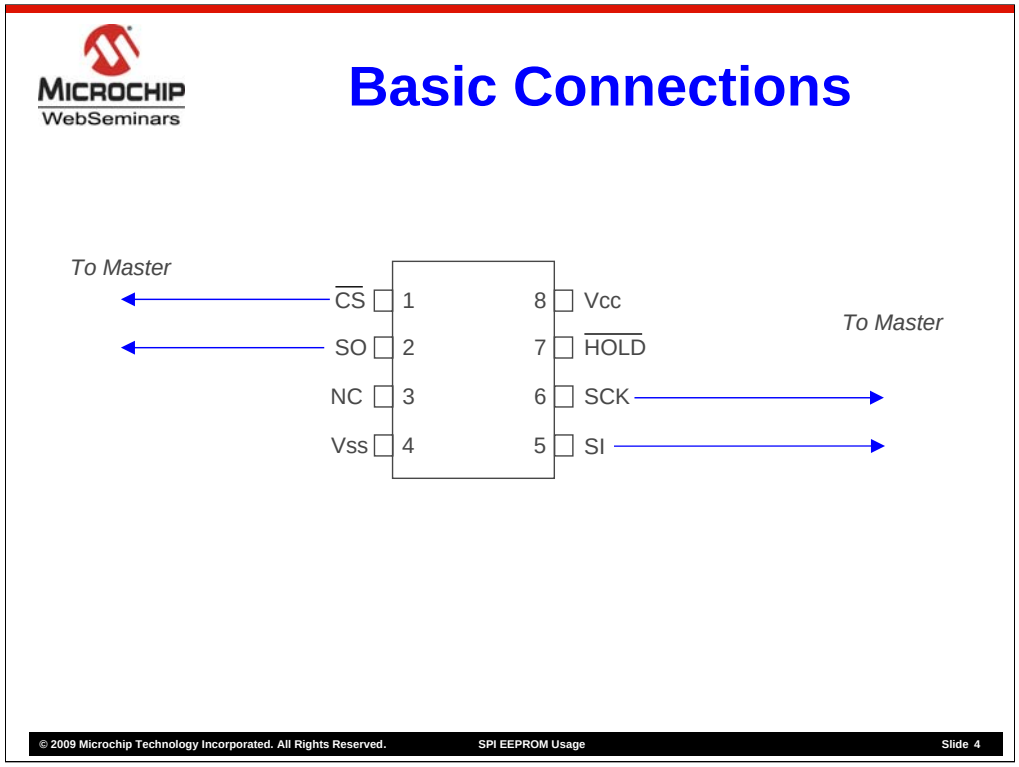
Second, the devices' Status register allows engineers to use the parts in byte, page, or full-chip sequential mode. We will talk more about these modes in the Status register section of this seminar.

Third, the devices come in two density options: 8 and 32 Kbytes. This gives designers a simple way to add extra RAM to an application. We've labeled these parts the 23x640 and 23x256.

4th, the devices have very low power consumption. Typical standby current is 1 uA at 3 volts. Max read current is just 10 mA at the maximum speed of 20 MHz. This makes the serial SRAM devices excellent for low-power embedded applications.

Next, the bus allows data rates of up to 20 MHz for very fast data transfer.

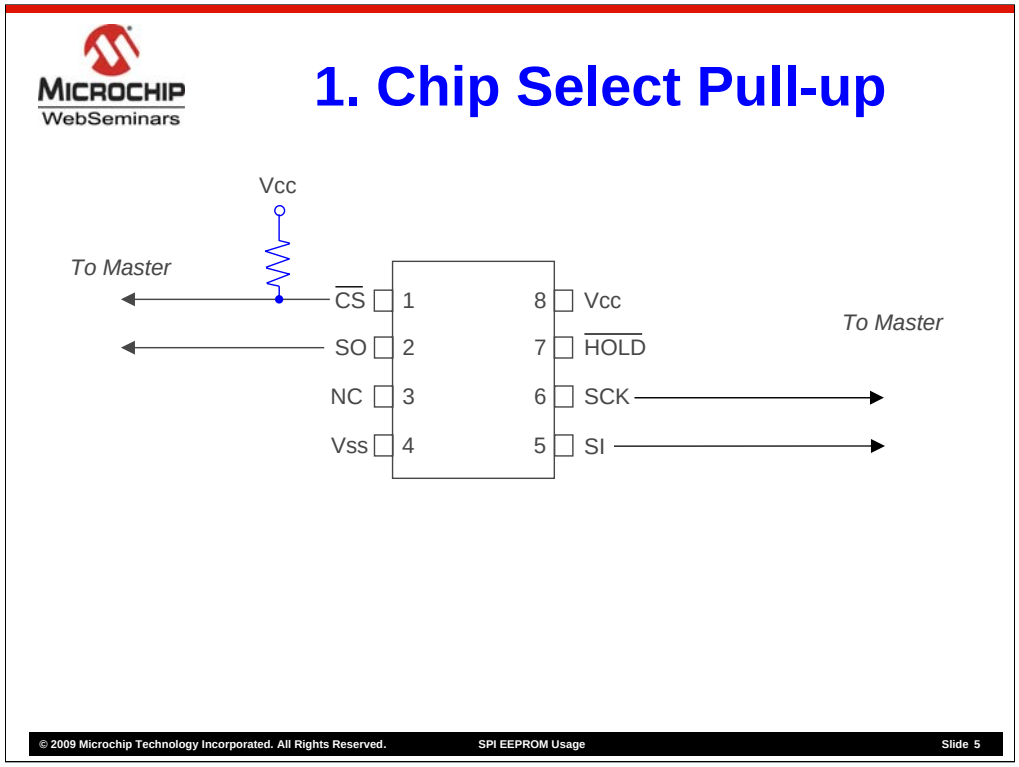
Finally, they come in 1.8 and 3V options and can operate up to 125 degrees C.



In the next few slides, I'll show you the connection details of serial SRAM products. We'll start in this slide which shows the 4 basic connections of an SPI interface: chip select, data out, data in and clock are all connected to the master. Remember that one of the advantages of serial SRAM over parallel RAM is that you only need 4 I/Os to connect a serial SRAM device.

These connections are often simplified by microcontrollers that have built-in SPI ports. Our web site has application notes with code to assist in making these connections.

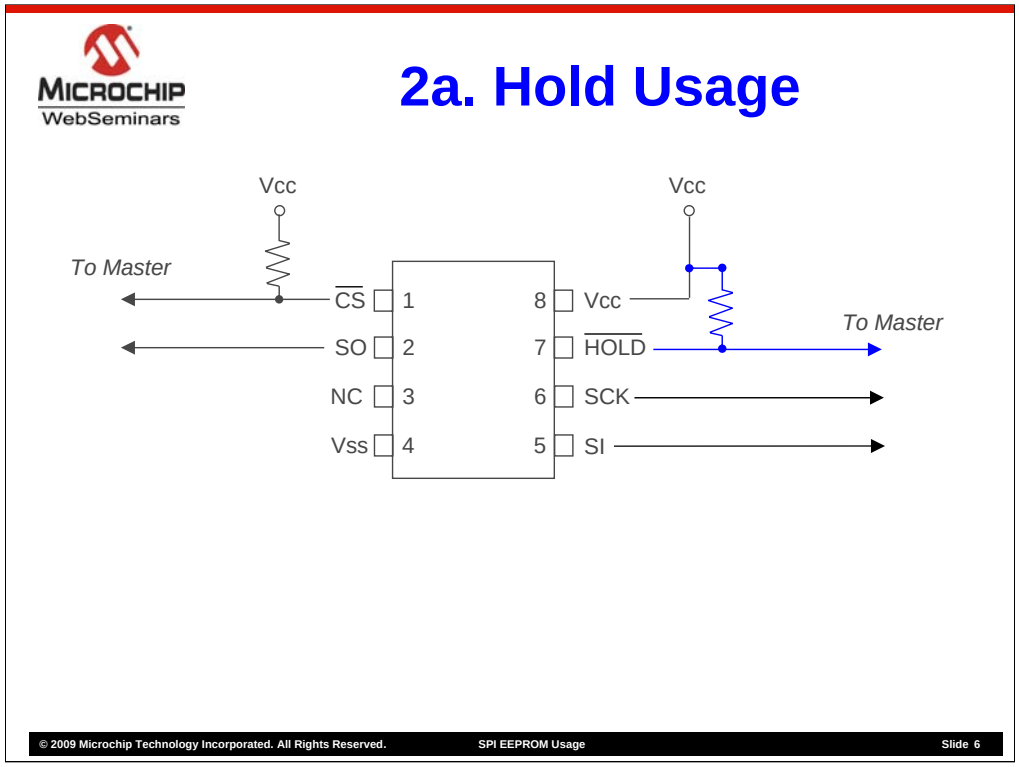
Now that we've seen the basic connections, we'll start our recommendations.



The first recommendation is to use a pull-up resistor on pin 1, chip select. This prevents the pin from floating in power up and power down situations.

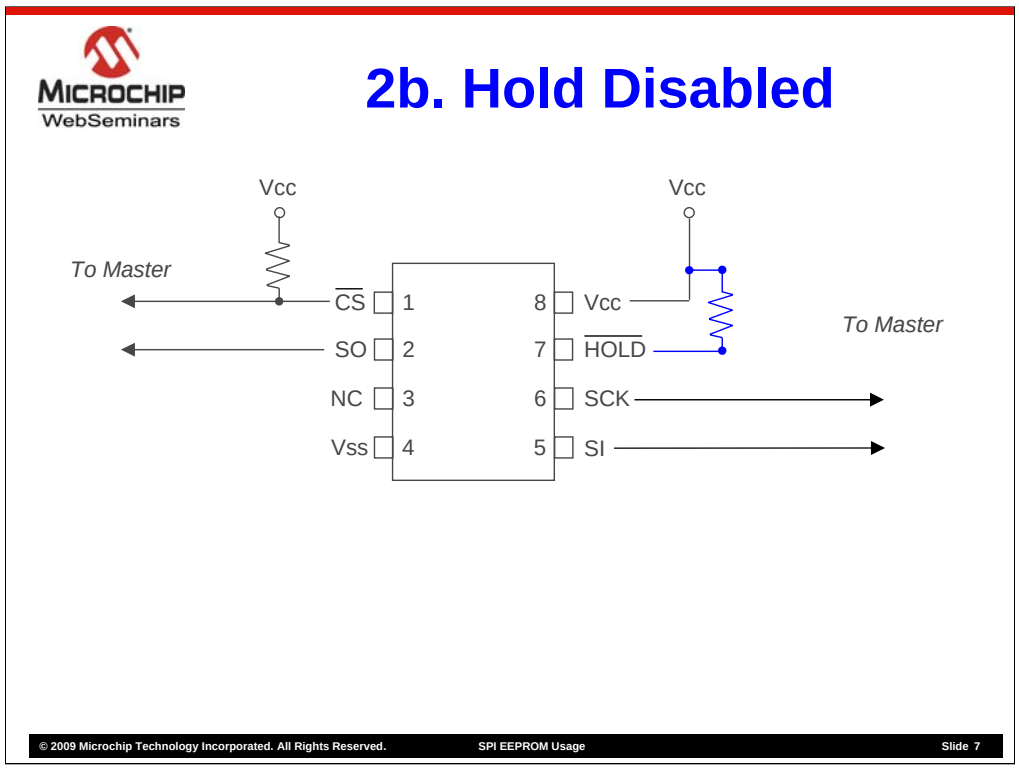
In normal operating conditions, the master controls chip select. If the master pulls the chip select pin low, the device is selected and will accept commands.

But during power up and down situations, pins can float high or low, depending on system conditions. If chip select is allowed to float, it could end up floating low. In this case, the SRAM device would be selected, and random inputs could be interpreted as a write command – which could corrupt data. A pull up resistor solves this issue by keeping Chip Select high, and thus deselected, during these indeterminate power levels.

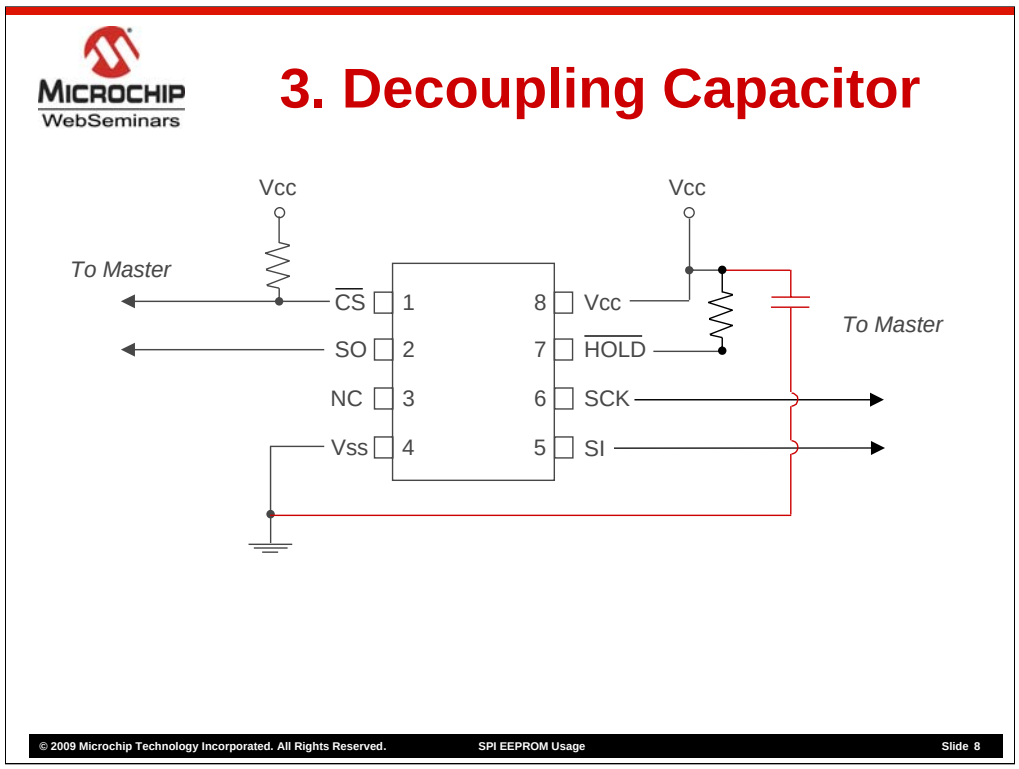


Next, let's look at the HOLD pin's operation. Pin 7, HOLD, is used to suspend a bus transmission to the SRAM device in the middle of a sequence allowing it to be completed later without the need to re-send the entire sequence.

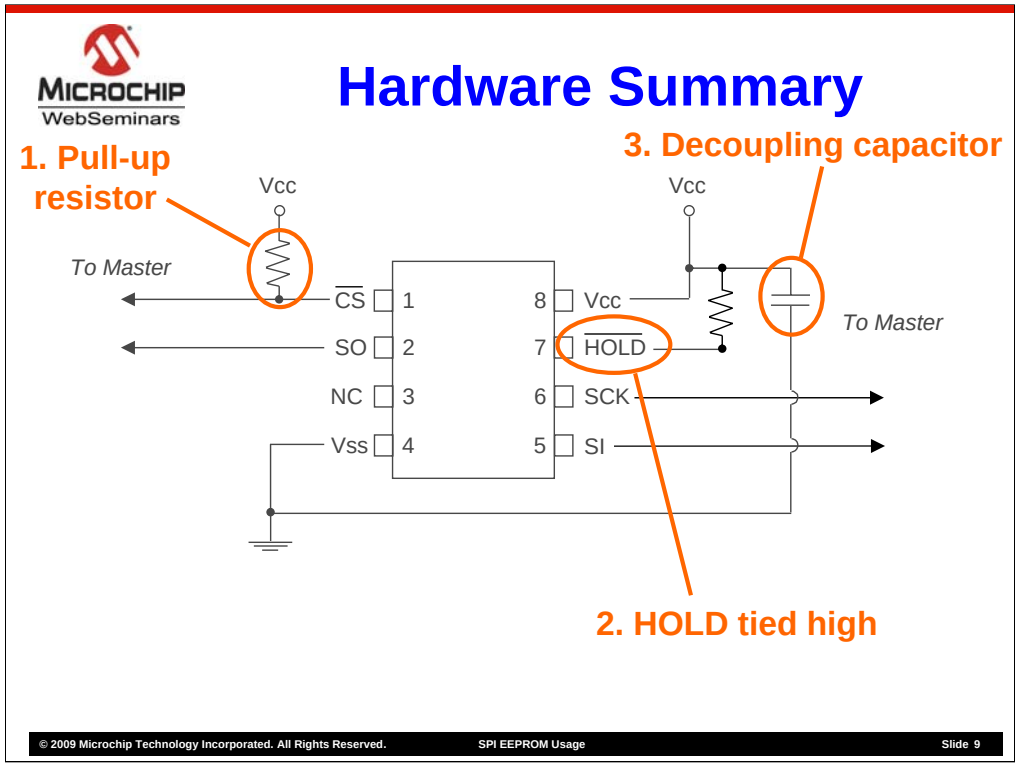
If the HOLD pin is used in an application, it must be controlled by the master. HOLD is active low, so it is held high during normal operation. It too should have a pull-up resistor to prevent floating that could put the chip into an unexpected Hold condition.



Most applications don't use the HOLD pin's functionality. If this is the case, the pin should be tied high through a resistor. Note that this set-up frees up a microcontroller pin. This is the most common implementation of the HOLD pin.



Our 3rd recommendation is to make sure to use a decoupling capacitor of approximately 0.1 uF. It should be as close to the device as possible to help filter high-frequency noise from the power supply.



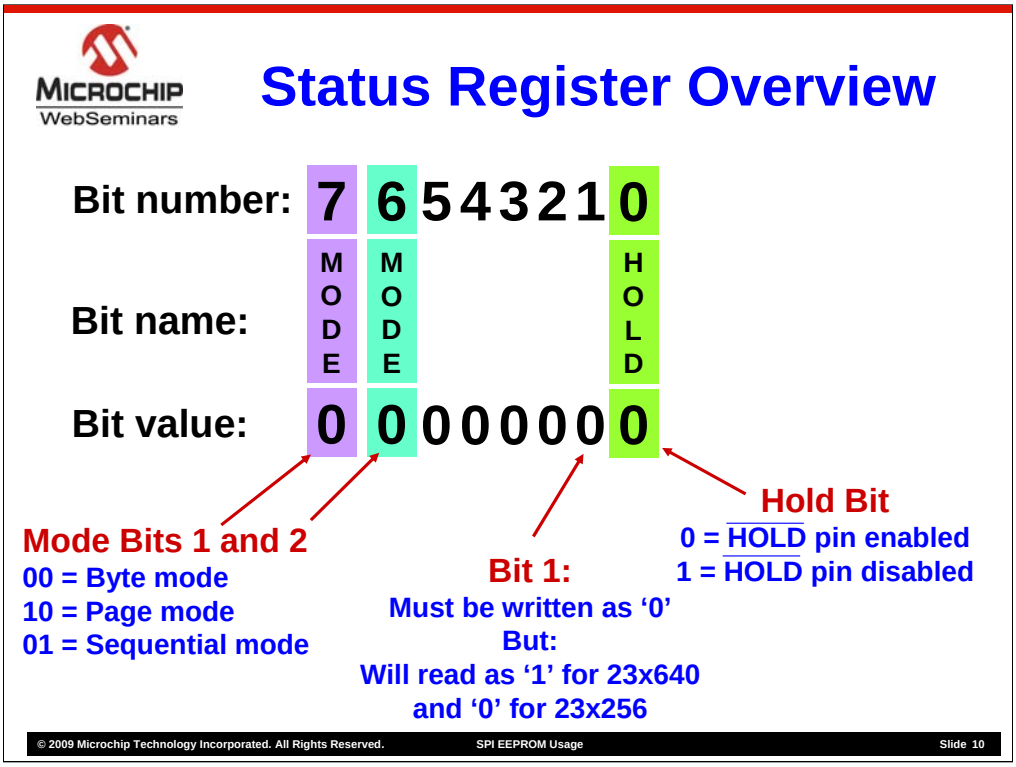
Here is a quick review of our hardware recommendations:

First, use a pull-up resistor on the chip select pin.

Second, use a pull-up resistor on the HOLD pin whether it is connected to the master or has been disabled by being tied high.

Third, use a decoupling capacitor.

For our next recommendation, we'll take a look at serial SRAM's Status register.



These SRAM devices use the SPI protocol which has an 8-bit Status register as I'm showing on this slide. The top row of the table shows the bits numbered 0 through 7. The next row down shows the corresponding name – hold and mode. Notice that serial SRAM only uses 3 of the 8 bits – 0, 6 and 7. These are all read/write bits and are changed with a Write Status Register command. They can be read by a Read Status Register command.

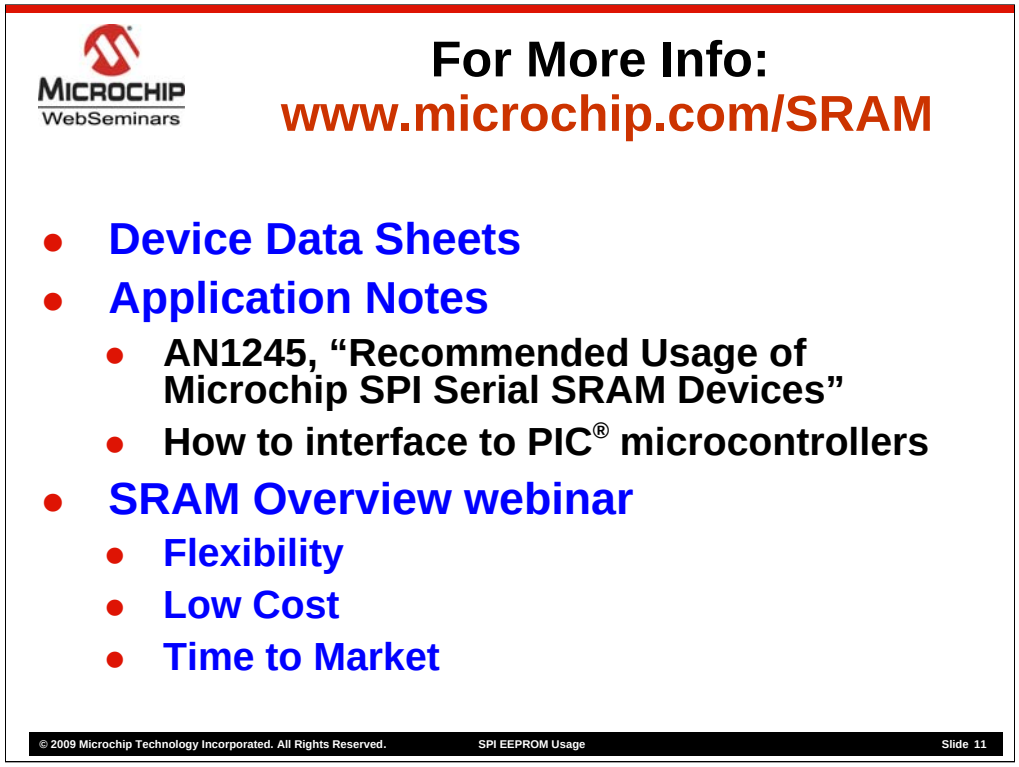
Now I'll describe the usage of each bit. Bit 0 is the Hold bit and enables and disables the HOLD pin. If this bit is set to 0, the HOLD pin is enabled. Then if the HOLD pin is toggled low, the transmission is suspended until the pin is toggled back high. If the Hold bit is set to 1, then the HOLD pin's functionality is disabled.

Bits 6 and 7 control the operating mode of the device. If these bits are set to '00', the part is in byte mode. In this mode, reads and writes are limited to the byte that was most recently addressed. If the bits are set to '10', the part is in page mode, and reads and writes are limited to within the current 32-byte page.

If the bits are set to '01', the chip is in sequential mode, and the entire chip can be read or written in a single command.

Bits 2 through 5 must always be 0s.

Bit 1 requires a special comment: when writing to the Status register with a Write-Status-Register command, it must always be written as a 0. But when reading the status register, bit 1 will read as a 1 for 23x640 devices and a 0 for 23x256 devices. So be sure to write bit 1 as a 0 for all parts.

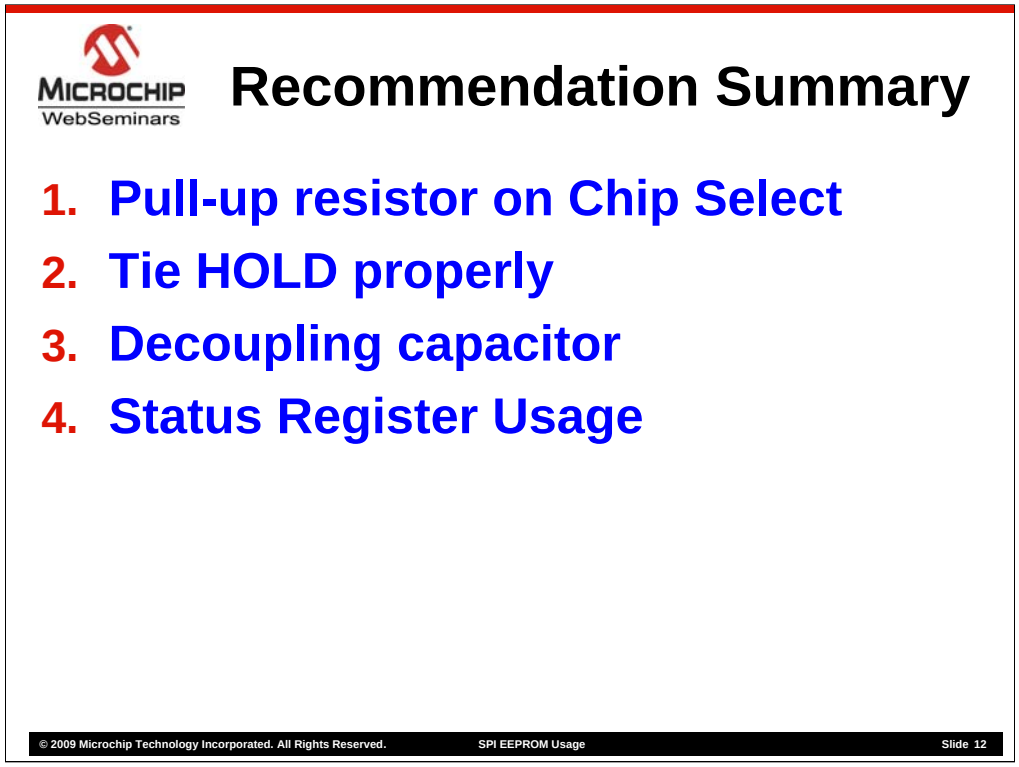


There is more information about Microchip's serial SRAM product line on our web page.

The product data sheet goes into more depth about how the devices work and how to use them – including a lot of information about how to use SPI.

We have several application notes posted, including AN1245, which accompanies this web seminar. There are also several application notes describing how to connect SRAM devices to microcontrollers. Many of these application notes also have code.

There is also a 2nd webinar on serial SRAM that is a product overview and describes how serial SRAM offers design flexibility, low costs, and improved time-to-market.



And that completes this web seminar about recommendations for serial SRAM designs. Let's quickly review them now.

•First, make sure the chip select pin has a pull-up resistor.

•Next, make sure the HOLD pin is either disabled by being tied high or is connected to the master. In either case, use a pull-up resistor to prevent floating the pin.

•3rd, use a decoupling capacitor.

•4th, remember to correctly use the Status register. It controls the HOLD pin as well as defines which mode the part is operating under. And remember that bit 1 must always be written as a 0.

To get more details on any of these concepts, please look through our application notes and data sheets.

Thanks a lot for your time.

Trademarks