Application Note 170
Adding Nonvolatile SRAM into Embedded Systems

INTRODUCTION

Dallas Semiconductor's nonvolatile (NV) SRAMs are plastic encapsulated modules that combine an SRAM, a power control IC, and a battery to provide high performance NV memory. NV SRAMs are the only NV solution on the market that does not specify a maximum number of write cycles. Additionally, the interface remains the same as a standard SRAM, and the read and write access times remain similar to the SRAM used within the module.

NV SRAMs are particularly well suited to microprocessor circuits because of their fast access times and the SRAM interface. External memory buses available on many microprocessors are intended for use with SRAM for data memory. Thus, due to the equivalent interfaces, NV SRAMs can simply replace an SRAM to provide NV storage in many applications. Often in the past, EPROM, EEPROM, and flash have been used for program memory space; however, NV SRAMs may also be used for program memory as well. This application note will show how to interface an NV SRAM to a microprocessor based system as either program or data memory, and list the advantages that using a NV SRAM will bring to the system compared to other NV memories currently available.

DIFFERENCE BETWEEN EPROM, EEPROM, FLASH AND NV SRAM

Although EPROM, EEPROM, Flash and NV SRAM are compatible NV storage solutions to some extent, accidentally choosing one that is a poor choice for a particular application can be crippling to a design. The main challenges that a microcontroller solution will impose on memory choice are the following:

- Availability of densities appropriate for the application
- Fast access rate required when used for program memory space
- A limited number of write cycles for NV memory can cause a reliability problem as the product approaches its end-of-life cycle
- Block writes are undesirable, particularly if the blocks of memory are larger than can be cached by a microprocessor
- UV erase time is inconvenient for development, cannot generally be done in-circuit, requires an additional special piece of equipment, and parts can erase in sunlight if not properly treated

By looking at all four memory types and charting them based on performance (Table 1), it is apparent that NV SRAM is the only memory type that will score high based on all of these benchmarks.

EPROM fails to be convenient by requiring UV erase time and adding high voltage for V_{PP} and V_{CC} during programming. EPROM can be a viable solution for program memory because it does provide the proper interface and read access times for a microcontroller application. The lack of electrical erasability makes it useless as NV data storage.

EEPROM is a nice solution for program memory space, but its limited write cycles and slow write access make it less than ideal for data memory. Because of the limited number of write cycles, it is only suited for program memory space in most applications.

At first glance, flash appears to be a good solution. It offers both synchronous and asynchronous read functions, and the write cycles are fast compared EPROM or EEPROM. The problems facing flash memory are the limited number of write cycles, and partitioning of the memory into blocks (64 bytes to 64k bytes) that must be written to all at once. This means that the data must be cached by the microprocessor, and then written to the flash device a block at a time. If the microprocessor in the design does not have a large amount of internal RAM, it may not have the resources to cache the data. For some applications, such as data loggers, this causes a problem because it is possible to lose the data leading up to a power fault condition if the data is still stored in the cache of the microprocessor. Although a few flash devices that are available on the market do not require block writes, the bulk of the products at this time have this limitation. This also brings up another limitation of the products: the lack of standardization of flash devices. Although most all of the flash devices allow asynchronous reads similar to an SRAM or EPROM, not all devices do. Also, there is a very wide range of write interfaces that are available. This could potentially become a problem as devices reach their end of life if the interface is not supported with the next generation. Care also needs to be to be taken to ensure that the maximum number of write cycles is not exceeded or the memory may begin to experience random bit errors as the part reaches its end of life. Several flash memories also require a programming voltage (~13V) to enable a fast write cycle mode. The additional voltage level can add cost to a design that would otherwise not require that voltage level. It is also hard to find flash memories below 1Mb in density.

FOUR MEMORY TYPES Table 1

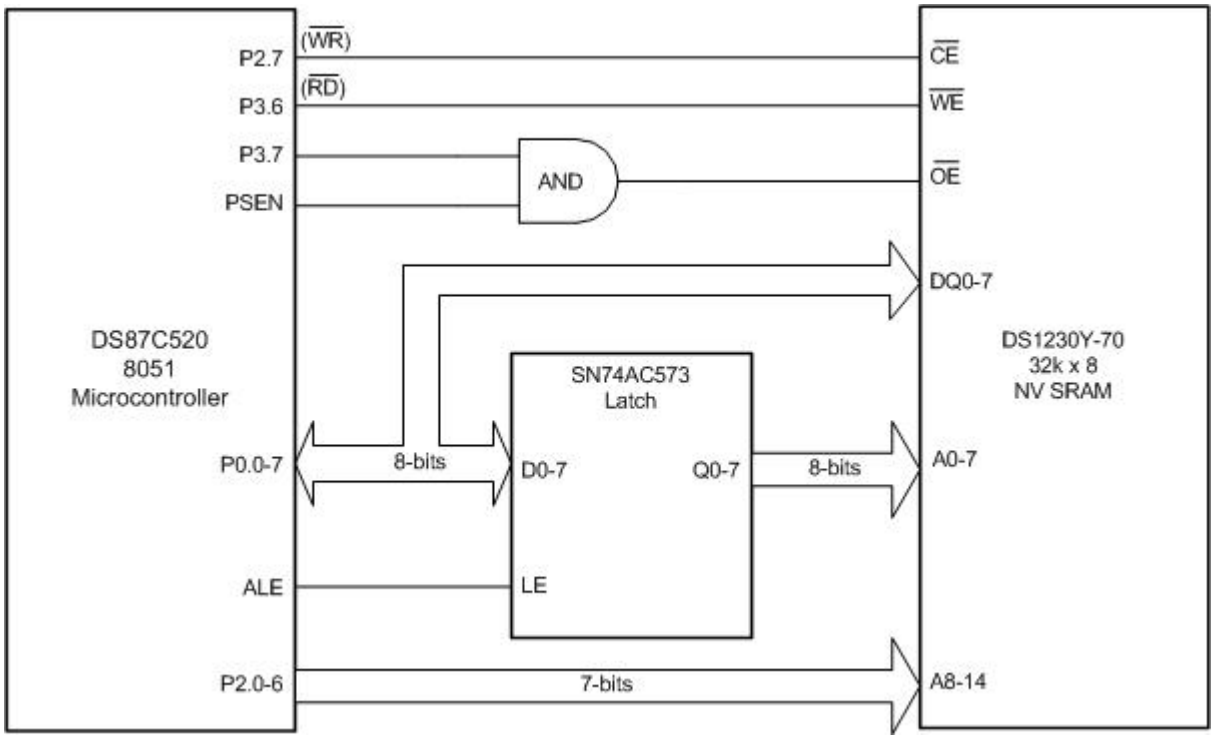
Memory	Density	Read Access (ns)	Write Access Time	Write Cycle Lifetime	Block Writes	UV Erase
EPROM	2k to 4M x 8	45 to 200	100µs / byte After part erased V_{CC} = 6.5V, V_{PP} = 13V	100 write cycles	No	Yes
EEPROM	2k to 512k x 8	70 to 200	200µs to 1ms / byte	10k to 100k write cycles	No	No
Flash	128k to 8M x 8 partitioned	45 to 200	Nonstandard write interface, most devices require block writes, fastest devices write at Around 10µs / byte with V_{PP} = 13V	~10k write cycles	Y/N	No
NV SRAM	2k to 2M x 8	70 to 200	70 to 200ns / byte	Unlimited cycles for up to 10 years	No	No

NV SRAMs are available in densities from 16kb up to 16Mb, and offer access times as fast as 70ns for read and write operations. The interface remains the same as SRAM for both reads and writes. During nominal V_{CC} conditions, the NV SRAM will operate exactly like an SRAM of equivalent speed. There are no block boundaries to worry about, and the data will be stored in the same location as it would with a normal SRAM. As the power fails (V_{CC} drops below its minimal level for operation), it will write protect the data by internally setting the chip enable ($\overline{CE}$) signal, and it will switch to battery power to retain the data. As the power is brought back above the trip point, the NV SRAM will return control of the $\overline{CE}$ signal to the microprocessor and switch the SRAM's power back to V_{CC}. As long as the microprocessor's control signals are inactive when control of the $\overline{CE}$ signal is returned to it ($\overline{CE}$ is high) no data will be lost.

Since NV SRAMs are SRAM-compatible during operation, it can simply be used to replace an SRAM of an equivalent speed grade in any microprocessor system. The only limitation that this places on the system is a need for the microprocessor to be aware of the current V_{CC} level. Dallas Semiconductor recommends the use of a CPU supervisor that will hold the processor in reset until the NV SRAM is ready. Most microcontrollers will default the bus control signals ($\overline{CE}$, $\overline{WE}$, $\overline{RD}$) to inactive during reset, and this will prevent data from being lost at any point in time by guaranteeing that the NV SRAM resource is always available while the microprocessor is operating. Dallas Semiconductor makes a wide variety of CPU supervisors that can accomplish this task, and several other functions that are related to the reset signal could be added to make the system even more reliable.

Figure 1 shows one versatile setup that can be used to make a single NV SRAM work for both program and data memory space. The primary advantage of this is it keeps the component countdown, but another hidden advantage is that it can make all of the memory space useful. If the microcontroller is only using 1kB of 32kB for program memory space, the other 31kB can be used for NV data storage. In many instances, a total of 32kB of space may be all that is required, but segmenting the program and data space into two types of memory can waste overlapping space and force the system designer to use larger memories.

USING A SINGLE NV SRAM FOR PROGRAM AND DATA MEMORY Figure 1



USING AN NV SRAM FOR DATA SPACE

NV SRAMs are great resources in microcontroller systems for storing calibration information, data logging applications, and any other application where settings need to be stored through power cycles. As mentioned before, NV SRAMs operate the same as an SRAM of the same speed while powered to a nominal V_{CC} level. Because of the speed of NV SRAMs, it is generally not required to use clock cycle stretching while reading or writing to them unless they are attached to a fast DSP or advanced microcontroller.

Most microcontrollers, including 8051 derivatives, provide control signals to access program memory and external RAM individually. Figure 1 is an example showing how to use the existing control signals to

generate a new set of control signals where one NV SRAM can serve as both program and data memory. Figure 2 shows how to address two separate memories in a 3V system using the existing control signals. The main difference is that PSEN is used to enable the output of the program memory, and the $\overline{RD}$ signal (P3.7) of the 8051 is used to enable the output of the data memory. This could cause bus contention if a condition existed where both memories were driving the bus. The other possible bus contentions that could occur are when the microcontroller is driving data to the NV SRAM while an external memory was instructed to drive data out at the same time. This particular design was checked to ensure that bus contentions will not occur, and it should be checked any time that a new microprocessor is being used with external memory.

Two things that should be pointed out about this design are that the program memory is permanently write-protected, and it is possible to address more than the microprocessors natural addressing space for data memory.

When using a NV SRAM with the $\overline{WE}$ pin tied high, nothing in the circuit will be able to write data to it, so a memory programmer will have to be used to program it. BP Microsystems and Data I/O programmers support almost the entire Dallas Semiconductor NV SRAM product portfolio. When using a programmer to program the parts, care must be taken to avoid subjecting the parts to ESD before they are placed in the circuit. If an ESD hit does occur, it can cause the parts to lose data before they are safely in-circuit. Once the part makes it to the board with the correct data, the NV SRAM will protect the data's integrity from that point on.

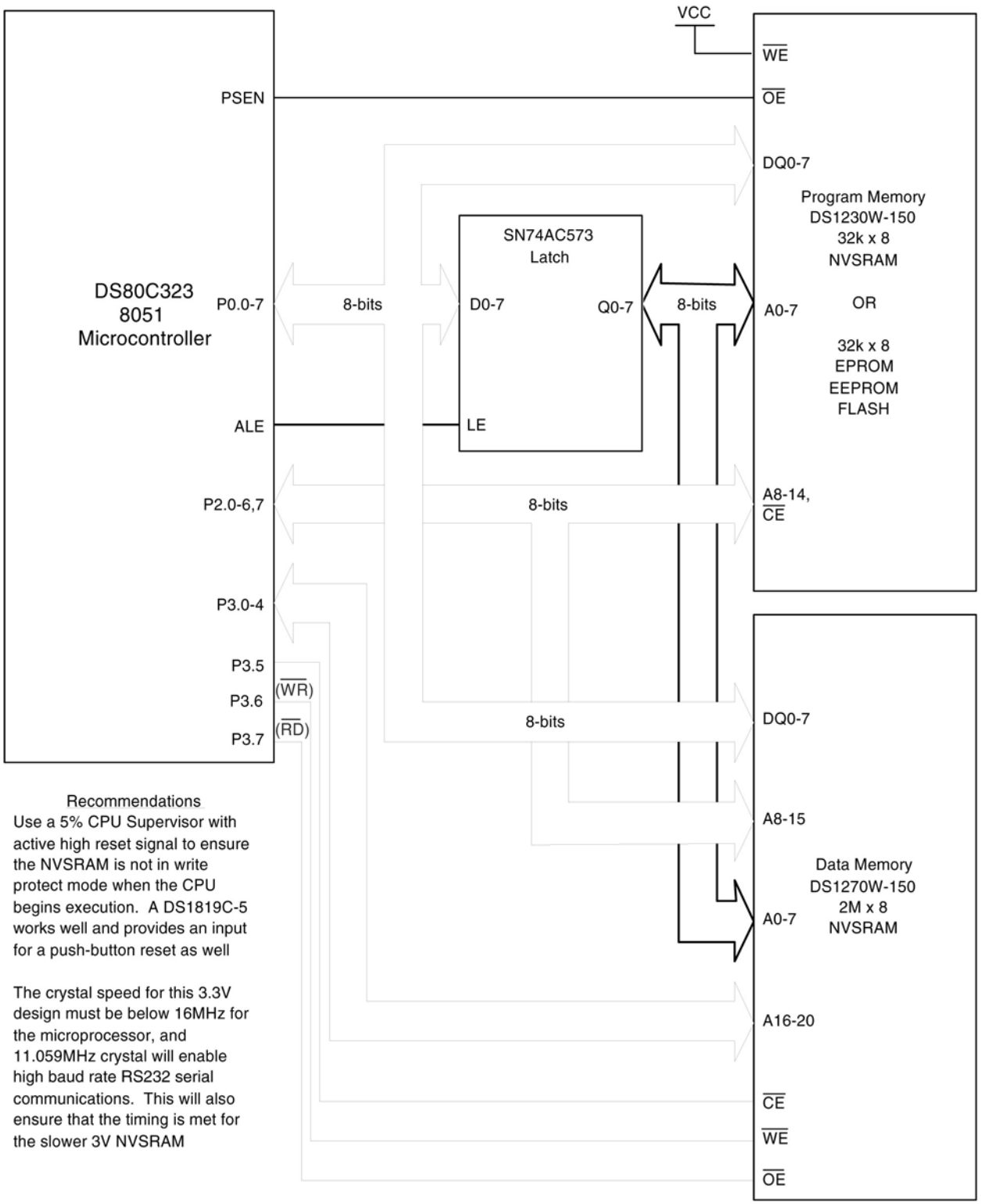
As far as addressing more data memory goes, it is as simple as adding the upper address lines to a digital port on the microprocessor. The 8051 shown on the diagram natively supports 64kB (A0 to A15) of addressing space. The last five address lines (A16 to A20) and the chip select pin can be used to select 64kB memory pages, and with couple of read and write subroutines addressing the larger memory becomes transparent to the programmer. This does not work for program memory, because the microprocessor is not aware of the pages when it is fetching code, so it will fetch an instruction at the proper 16-bit address, but possibly in the wrong page. Thus, if a DS1245 were used in Figure 1, then only one-half of that memory would be addressable. Attempts to use A16 to create a second page of memory would cause the microcontroller to fetch instructions in the second page whenever accessing data in the second page of memory.

Other things that should be considered when performing a design like this include:

- Examination of the bus timing. Are the control signals present long enough for read and write cycles to occur without error? This examination should be specific to the system clock rate since the timing of the signals is always clock rate dependent. If the bus timing is too fast for the data memory, most processors have a provision for inserting entire clock cycle stretches to make the data memory accesses slower.
- The bus control signals need to be examined during the power-up and brown-out conditions. If $\overline{CE}$ = $\overline{WE}$ = 0V while V_{CC} is above the trip point, then the data at address present will be corrupted if it is not an intentional write attempt. If the system is only addressing the native addressable space, then verification that the bus control signals are inactive during reset is adequate to ensure data will not be corrupted. When using extended addressing, as is shown in the example below, it must be verified that the I/O signals controlling either $\overline{WE}$ or $\overline{CE}$ are high during both power-up and brown-out conditions. Additionally, verify that no two chips are will be driving the bus while the processor is in reset. The use of a CPU supervisor may be required to guarantee that the microprocessor does not try to access the NV SRAM while the NV SRAM is in battery backup mode. When in this mode the chip-enable signals internal to the module are high regardless of the $\overline{CE}$ signal external to the module, and thus the SRAMs

in the module will ignore any attempts to access them. This is particularly bad when an NV SRAM is used as program memory because it can cause the processor to execute junk instructions.

USING SEPARATE MEMORIES FOR PROGRAM AND DATA SPACE IN A 3V SYSTEM Figure 2



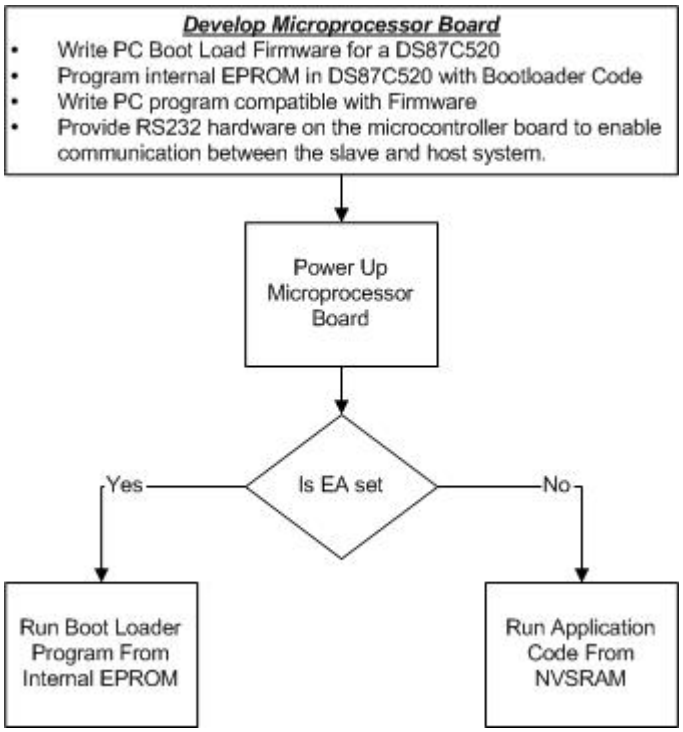
MICROCONTROLLER REQUIREMENTS TO USE AN NV SRAM FOR PROGRAM MEMORY SPACE

The two special concerns when using an NV SRAM for program memory are 1) that the NV SRAM must have a sufficiently fast access rate that it does not require clock cycle stretching, and 2) the NV SRAM is not accessible when V_{CC} is below the voltage monitoring trip point. A microcontroller has to fetch an instruction every machine cycle (generally there is more than one clock cycle per machine cycle). If the program memory is slow, the clock rate may have to be slowed down to accommodate the memory access time. This will obviously affect the systems performance. The fastest 5V NV SRAMs have a 70ns access time, and 3.3V NV SRAMs can be accessed as quickly as 100ns. This is sufficient for most applications.

The use of a CPU supervisor with a trip limit closer to V_{CC} than the NV SRAM eliminates the problem of a microcontroller attempting to access a NV SRAM before it is ready. One other related possibility is that V_{CC} noise can cause the instantaneous voltage to dip below the V_{CC} trip limit. This will also cause the part to become inaccessible for a short period of time after the trip limit is violated. For this reason, it is strongly recommended that a dedicated decoupling capacitor is placed near the V_{CC} pin of the module. Because 5% CPU supervisors are readily available, it is easiest to use a NV SRAM with a 10% trip limit. Both 3.3V and 5V parts are available that meet this criteria.

As mentioned before, it is not a recommended practice to use an NV SRAM as a portable data carrier. The chance of data loss when the part is not in a circuit is significant. The best solution is to program the parts in-circuit. An easy solution to enable in-circuit programming for an embedded system is to build a boot loader system as shown by the flow chart below.

BOOT LOADER FLOW CHART Figure 3



The boot loader system works by switching the location of the program memory from the microcontroller's internal EPROM to external NV SRAM memory. The firmware that exists in internal memory is nothing more than code to receive data via the RS232 port, and copy that data to the appropriate NV SRAM location. An easy file format to use to implement this procedure is Intel's Hex File format, because it encodes both the

data and the location of the data as a standard file format which many assemblers use. A PC can then be used to send the data via a RS232 serial port to the microcontroller whenever the $\overline{EA}$ pin is set, which selects the internal memory for execution. Once the microcontroller has written the code out to the NV SRAM, turn the power off, clear the $\overline{EA}$ pin, and the microprocessor will begin to execute the code that was placed in the NV SRAM when power is restored. Once the initial firmware and software are developed, this speeds system development time by facilitating quick programming time. One disadvantage is that the crystal speed must stay fixed at a multiple of the development speed to ensure that the serial port's baud rate can be accommodated by the PC software.

MAXIM INTEGRATED PRODUCTS / DALLAS SEMICONDUCTOR CONTACT INFORMATION

Company Addresses:

Maxim Integrated Products, Inc
120 San Gabriel Drive
Sunnyvale, CA 94086
Tel: 408-737-7600
Fax: 408-470-5841

Dallas Semiconductor
4401 S. Beltwood Parkway
Dallas, TX 75244
Tel: 972-371-4448
Fax: 972-371-4799

Product Literature / Samples Requests:
(800) 998-8800

Sales and Customer Service:
(408) 737-7600

World Wide Website:
www.maxim-ic.com

Product Information:
http://www.maxim-ic.com/MaximProducts/products.htm

Ordering Information:
http://www.maxim-ic.com/BuyMaxim/Sales.htm

FTP Site:
ftp://ftp.dalsemi.com