

# Wanted: Secure, Low-cost On-Chip Code Storage for Embedded Signal-Processing Systems

Craig Rawlings, Michael Fliesler  
Kilopass Technology, Inc., Santa Clara CA

## Abstract

Consumer demand for appliances such as PDAs, digital cameras, cell phones, and MP3 players is forcing developers of such systems to continuously add new functionality to their products while offering them at lower price points. In addition, product vendors have become increasingly aware of and concerned about theft of product IP represented as code stored in silicon in these appliances. These market forces are changing the way such systems need to be designed.

Embedded signal-processing systems require storage of a security ID, application code, DSP coefficient parameters, and firmware running on the processing engines. Traditionally, this has been done with a combination of off-chip non-volatile memory (NVM) storage, either Flash or EPROM, and SRAM on the processing chip. A second technique eliminates the second chip and puts Flash memory directly on the processing chip. Either solution raises the cost of the system—either with the need for a second chip or in the increased cost of adding Flash fabrication to the processing chip—and opens the door for reverse-engineering theft of the code stored in Flash memory.

A better solution for embedded code storage is to add one-time programmable (OTP) NVM to the processing chip that does not require additional processing steps and, thus, does not add to the chip's fabrication cost. Other requirements for the OTP NVM would be high security, the ability to program the memory after chip fabrication, for enhanced product flexibility, and the ability to scale the memory to future process nodes for further system cost reduction.

This paper will review the problems with existing NVM solutions and describe a low-cost, reliable and secure OTP NVM technology for boot-up and program code storage for signal-processing applications. The technology also supports in-the-field modification of code and DSP coefficients—of critical importance to many DSP-based systems.

## Tradeoffs in Code Storage

There are several memory technologies available for storing firmware and other code in SoC-based systems, the most common being ROM, EPROM and Flash. Of these choices, EPROM use is rapidly decreasing with many applications that originally employed EPROM now using the less expensive and easier to program Flash. Both ROM and Flash represent technologies that result in tradeoffs in key design parameters—flexibility, scalability, power consumption, cost, and speed—in the systems in which they are used (**Table 1**).

	<b>ROM</b>	<b>EPROM</b>	<b>Flash</b>
<b>Overview</b>			
Process	Standard CMOS	Special	Special
Process Generation	Leading	2-3 Gen. Lag	1-2 Gen. Lag
Scalable Process	Yes	No	No
<b>Benefits Comparison</b>			
Programming	Mask	Field	Field
System Performance	High	Low	Low
Read Access	Fast	Medium	Low
Wafer Cost	100%	130%	150%
Power Dissipation	Low	Medium	Medium
Scalability and Reliability	High	Medium	Medium

**Table 1. Traditional Non-Volatile Memory Comparisons**

### ***Flexibility***

Today’s consumer-driven signal-processing systems must exhibit flexibility in the way the NVM that holds the system’s firmware is configured. Many systems are in development cycles during a time when signal-processing standards, such as those for video compression, are in a state of flux. These systems need to have “softer” firmware; in other words, firmware that can be modified either in the field or, at least, some time after completion of chip fabrication. Softer firmware also accelerates the development of derivative products and reduces the cost of this development.

A problem with mask-programmable ROM is that the processor’s firmware is “locked-in” when the hardware is finalized, during chip fabrication. Consumer-product feature sets change very frequently—sometimes every three months—and market demand often changes rapidly for every production version of the embedded masked-ROM device.

In a typical SoC system design flow, firmware is directly in the project’s critical path. Every change in the contents of a masked ROM requires an NRE charge and, in addition, a turnaround that may stretch to several months. ROM-based storage also adds cost to a system in that it increases inventory management expense.

Beyond accommodating changes in data standards and decreasing the time and cost of derivative products, processing systems that have firmware that can be configured after chip processing has other benefits as well. System ECOs (Engineering Change Orders) during system development may be more easily handled since such changes may be accomplished through software modifications. Product lifetime is increased, since firmware can be changed to deal with market and technology changes after the system is

deployed. Finally, firmware configuration during system development enhances the co-development of hardware and software, accelerating time to market for the system.

### ***Security***

With flash and ROM it is relatively easy to steal the information they store. Application software is very valuable to the company that has developed it and, therefore, also to that company's competitors. Downloading firmware from an external Flash chip makes the download process susceptible to undesirable interception by a third party. On-chip Flash, through reverse engineering, can be read to reveal the contents of the Flash memory, as can ROM.

### ***Cost***

Since many signal-processing systems are targeted for consumer applications, cost is a very critical design parameter. While ROM is small and processed in a normal logic process, flash is not. Flash memory is expensive. Off-chip, Flash adds the cost of an additional chip (along with higher power and lower access speed). On the same chip as the processor, Flash increased wafer cost by as much as 50%. For cost-sensitive consumer appliances such as cell phones, cameras, MP3 players, and PDAs, this is not viable.

### ***Scalability***

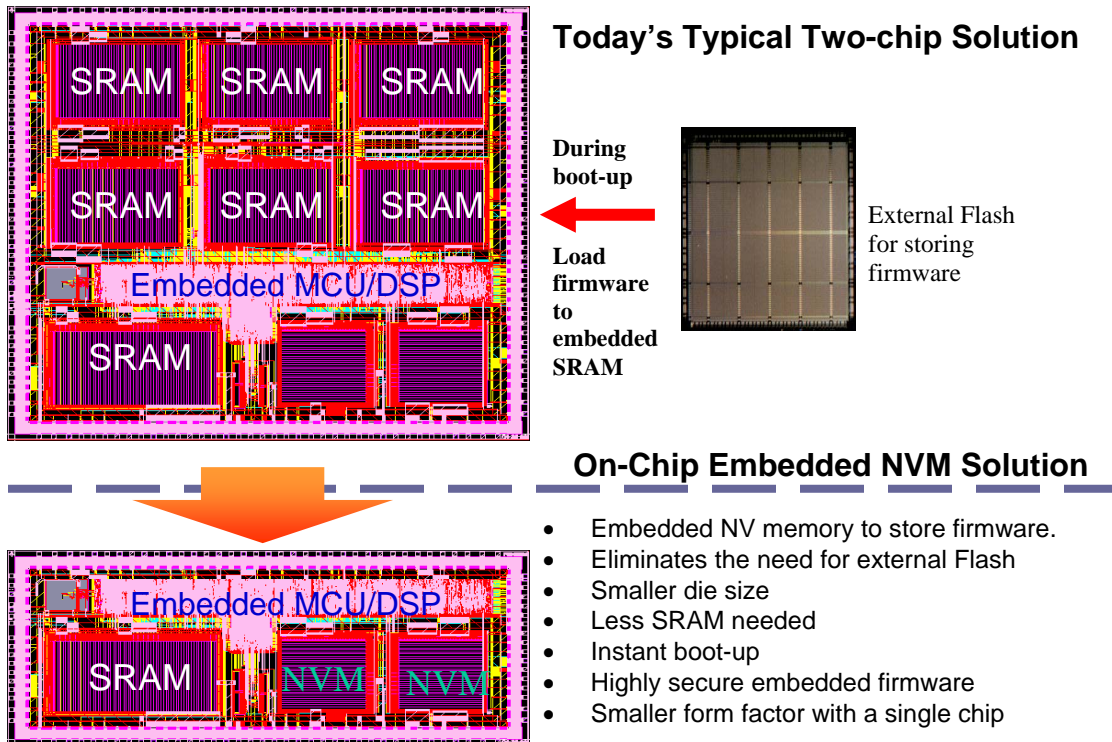
Flash and other charge types of floating-gate NVM technologies are limited in their ability to store charge— as the logic oxides get thinner, direct tunneling occurs and the charge drains through the tunnel. This limits the scalability of flash charge-storage technologies to around 80 to 85 Angstroms for the tunnel oxide thickness—the minimum thickness of the tunnel oxide flash required to reliably hold the charge once it is stored. Mainstream CMOS technology gate oxides are much thinner, in the range of 30 Angstroms. Another drawback is that flash technology, when implemented on-chip, is larger than other types of NVM, which can further increase chip cost.

### ***Power Consumption***

A big disadvantage of firmware stored in Flash on a second chip is power. Accessing the firmware, even if it is then downloaded to SRAM on the processor chip, is power-intensive compared to storing the firmware on the processor chip. This is undesirable for battery operated devices such as cell phones and digital cameras.

### ***Speed***

For many signal-processing applications, flash read-access times are too slow for the processor to execute code directly from the flash memory. To circumvent this problem, many systems use a separate flash chip to store code and then use SRAM on the same chip as the processor as a buffer to store the code downloaded from flash memory (**Figure 1**). While read-access of embedded NVM is slower than that of ROM, system cost and power are reduced by going from a two-chip to a one-chip solution.



**Figure 1. A two-chip solution for firmware storage uses an external flash chip to store code, which is then downloaded to embedded SRAM (top). With low-cost and secure embedded non-volatile memory, the cost is reduced and code security is assured (bottom).**

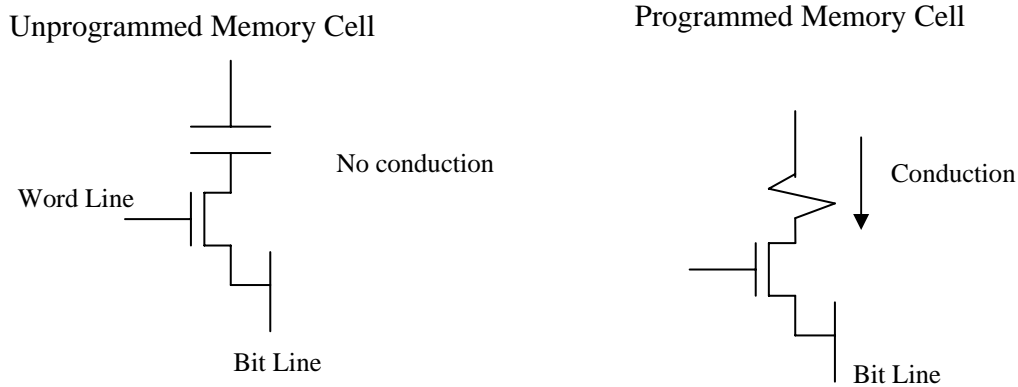
### Signal-Processing Memory Code-Storage Requirements

A typical signal-processing application, a handset chip-set for cell phone use, has a memory footprint requirement for program-code storage of around 32 Kbytes and a read-access requirement of 20ns. Embedded ROM meets these requirements, but is not optimal since ROM is programmed during wafer processing, resulting in system firmware development being in the critical path of system development. For cost-sensitive applications such as cell phones, both on- and off-chip flash add significant cost to an already low-margin market. What is needed is an embedded NVM that is low cost, secure, and configurable after completion of wafer fabrication.

### XPM for Signal-Processing Applications

Kilopass Technology has developed a one-time programmable (OTP) NVM technology that has many of the attributes required for signal-processor code storage. The embedded memory is processed in the same CMOS logic process as is the processor and thus adds no additional mask steps or processor layers and, hence no additional processing cost. The technology is scalable to shrinking process nodes and has been verified down to 90nm.

Floating-gate technologies such as flash and EPROM have data retention and scalability problems. XPM does not depend on injection and tunneling of charge to and from a floating gate to store ones and zeros. Instead, the XPM memory cell uses a very short channel MOS transistor which looks like a capacitor (an open circuit) in the unprogrammed state. When programmed by a short voltage pulse, the gate oxide undergoes a permanent breakdown and the transistor conducts like a resistor (a closed circuit). However, the memory cell does not add any additional leakage current during normal operation, keeping power consumption low.

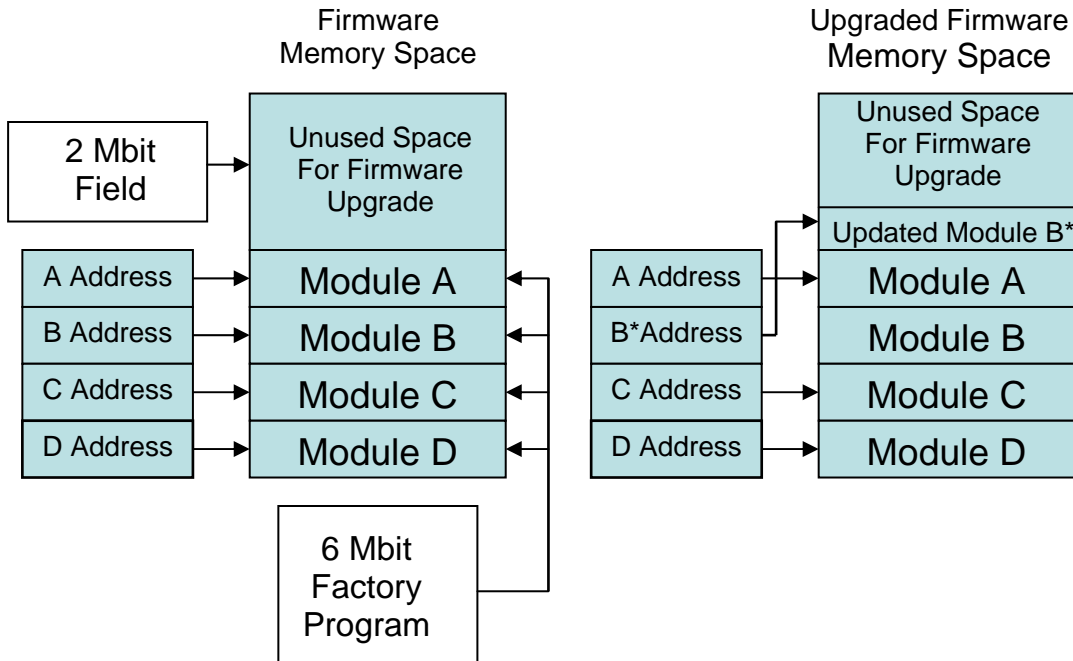


**Figure 2. With XPM, a short programming voltage pulse breaks down the oxide in a short-channel transistor, converting an open circuit path into a closed one.**

The current needed to program an XPM memory block is low and a 1-Mbit memory can be programmed in just a few seconds. XPM is dense, at  $1 \frac{1}{2}$  transistors per bit, and inherently secure since the programming of a bit is within the gate oxide and cannot be read by voltage contrast scanning techniques.

Read access for current XPM technology is around 50ns but should be improved early next year to around 20ns. This is adequate for many current signal-processing applications, but others push this access time down to 10ns or less.

NVM reconfigurability as offered by flash comes at a price. The system developer can modify the system in the field to account for software upgrades or other modifications, but the technology has larger memory cell size, additional processing cost, and a lack of scalability to leading-edge CMOS processes. In many cases, a one-time programmable NVM such as XPM can be used in place of traditional reconfigurable flash NVM. The trick is to allow some “patch” space: include one or more uncommitted sectors in the OTP memory along with the sectors that store existing program code. To upgrade one of the program-code modules, the upgraded module is programmed into an unused memory sector and control logic is switched to point to the updated module. Thus the memory, which is OTP on a cell-by-cell basis, is actually “few times programmable” at the system level. Since XPM does not consume a lot of chip area, having unprogrammed sectors in a memory instance does not have a significant impact on chip size and yield.



**Figure 3. By including some unprogrammed sectors of XPM memory, system developers can update firmware by placing the upgrade, in memory, in the originally unprogrammed sector.**

### Looking Ahead

Currently both ROM and flash program storage have limitations with respect to one or more of the following parameters: cost, access time, security, scalability and time-to-market risk. A new one-time programmable NVM, using gate oxide breakdown to program a bit, shows promise due to several attributes, including small size, low processing cost, secure storage, and scalability to new process nodes to lower product cost and/or increase a system's feature set. Although current XPM access times are slower than that of embedded ROM, XPM is still a promising solution for embedded boot-up and firmware storage in many current signal-processing systems. Furthermore, read-access times should significantly improve over time with technology and process-node advancements.