MODULAR OPEN SYSTEMS APPROACH

Challenges and Mitigations for Data Remanence in FPGA Based Systems

Kevin Paar, Senior Research Engineer, Graf Research
Dr. Scott Harper, CTO, Graf Research

This material is based upon work supported by the Government under Contract No.W56HZV-21-C-0020. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Government.

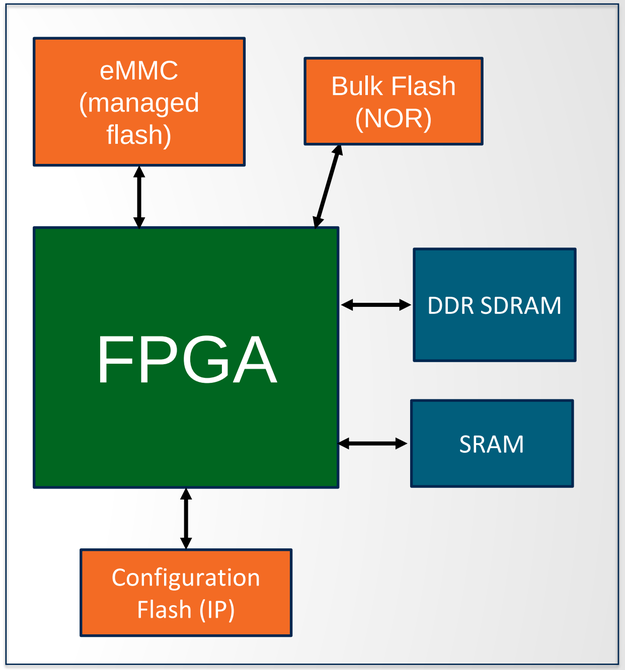
Information in Electronic Systems

MODULAR OPEN SYSTEMS APPROACH

- Systems contain sensitive information
  - Intellectual property (ASICs, FW, SW)
  - Configuration
  - Transient operational data
- Recoverable information can be exploited
  - Systems need to be cleansed
- Typical approach is kinetic physical destruction
  - Potentially dangerous, conspicuous, questionable completeness
- FPGA based systems support non-destructive approaches
  - Unlike ASICs, FPGA IP can be *zeroized*

FPGA Based Electronic Systems

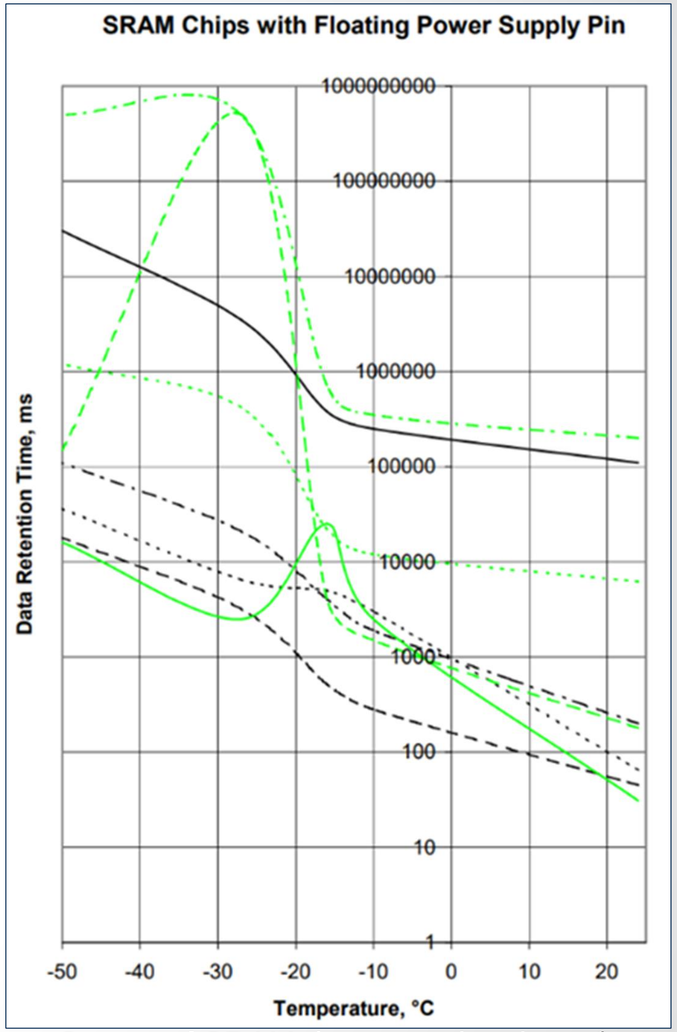
- An FPGA is at the center of many complex systems
  - The FPGA configuration data is zeroizable IP
- An FPGA system contains many forms of storage media
  - Volatile media
    - SRAM, DRAM, etc.
  - Non-Volatile
    - Flash (bulk, managed)
- Any media can contain sensitive information
  - Proper *sanitization* requires knowledge of media characteristics
- Media can be susceptible to data *remanence*
  - Data retention beyond intention
  - Data may be retained as physical changes



Typical FPGA system

SRAM Data Retention

- SRAM is volatile by design
  - Loss of power should eliminate data
  - Loss of data is random over time
- Power-off data decay can be longer than expected
  - Restoring power can allow for recovery
- SRAM decay dependencies:
  - Temperature
    - Low temperatures may push retention to hours or days
  - Silicon technology
  - System design

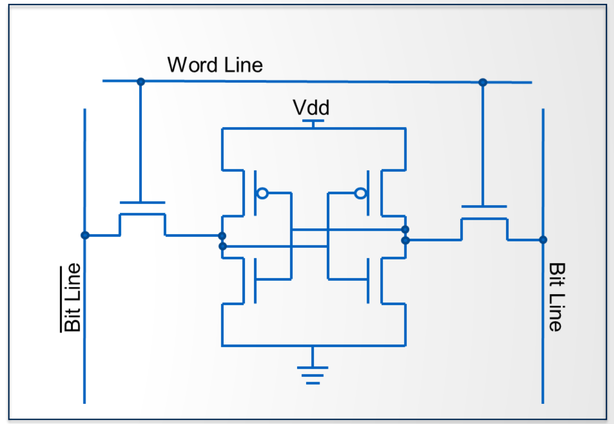


Survey of SRAM data retention times.[1] (20% data loss)

[1]S. Skorobogatov, “Low temperature data remanence in static RAM,” University of Cambridge Computer Laboratory, Cambridge, UK, Tech. Rep. UCAM-CL-TR-536, June 2002.

SRAM Imprinting

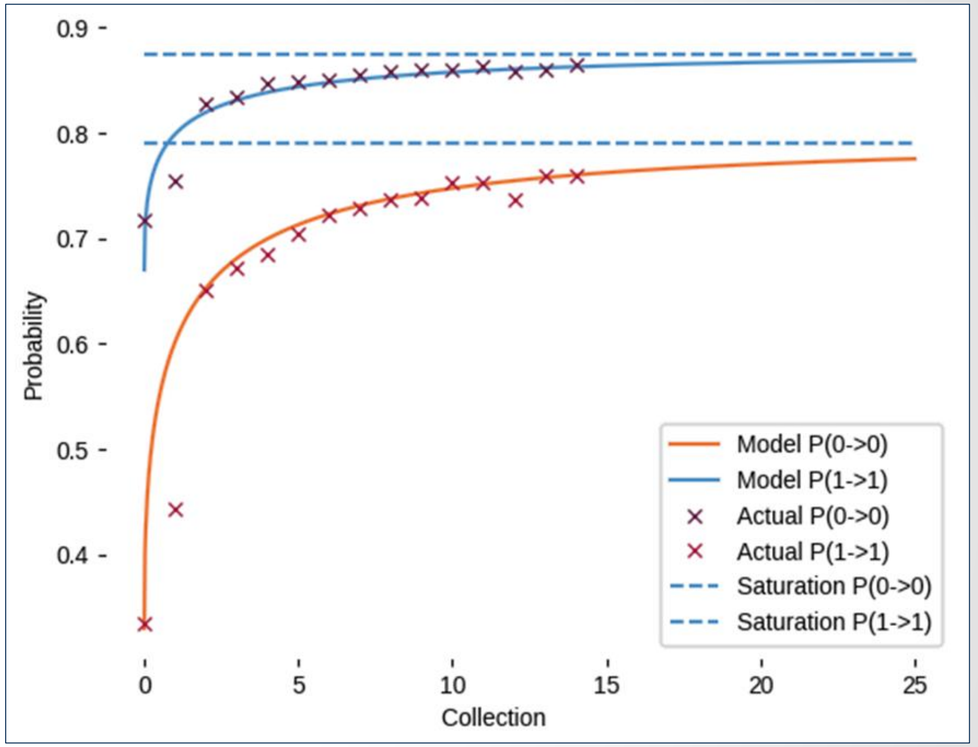
- Typical SRAM cells are painstakingly balanced
  - Initial power-on state is statistically random
  - Operational wear can tip the balance (adds bias)
- Static data retention causes physical changes
  - Negative Bias Thermal Instability (NBTI)
    - Increases PFET threshold voltage
  - Effects dependent on silicon technology
- Bias makes power-on state less random
  - Recovery can reveal previously held static data
- Reducing imprinting
  - Dynamic values avoid changes caused by NBTI
  - Elimination of bias requires leveling of the transistor wear



Typical balanced SRAM cell

FPGA Configuration Remanence

- FPGA internal configuration is SRAM
  - Operational configuration cannot be encrypted
- Configuration SRAM will power-up randomly
  - Normal operation clears configuration state
- Recovery of power-up state can reveal prior contents (imprinting)
- FPGA remanence has been characterized
  - Physical testing backed by models
  - Empirical data aligns to model
  - Modeling indicates theoretical saturation
- Mitigations also being studied

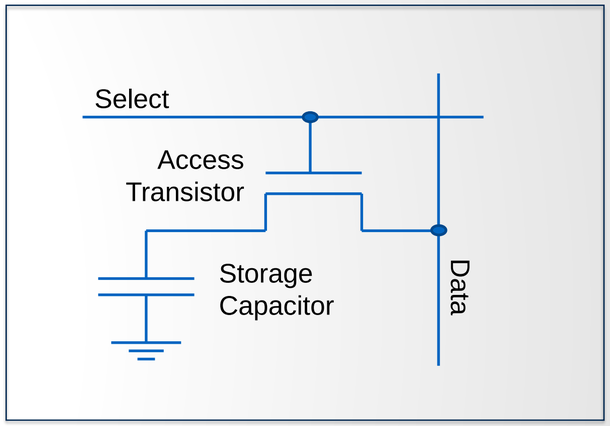


Probability of FPGA configuration data recovery vs aging in a common 90nm FPGA.

DRAM Remanence

- DRAM data is always decaying
  - Circuit has no active hold after write
  - Data values must be “refreshed” periodically
  - Optimized to reduce decay

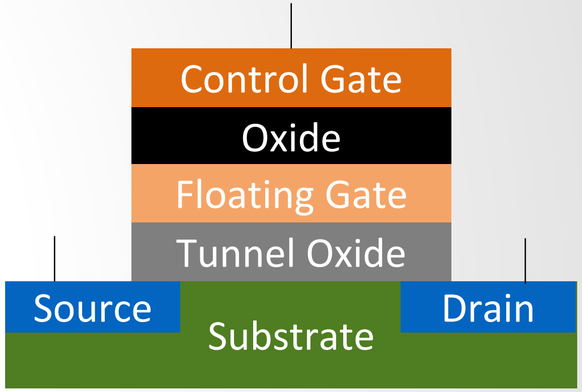
- Low temperatures can greatly increase decay time
  - Enables the “cold boot attack” (typically < -25C)
  - Successful recovery demonstrated with DDR1-4
    - One experiment achieved <0.17% data loss after one hour



Typical DRAM storage cell

Flash Memory Imprinting

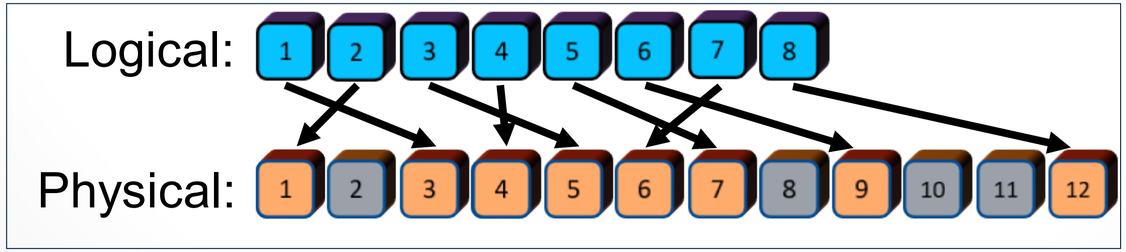
- Flash memory is based on floating gate transistor cells
  - Charge forced into or out of floating gate
  - Trapped charge encodes the data value
- Once programmed it cannot be completely erased
  - Some charge becomes *permanently* trapped
- Difference between unused and erased cells is significant
  - Careful read operations can reveal latent charge (imprint)
- Flash should be *pre-conditioned* to avoid remanence
  - Repeated program/erase cycles on all cells
- Multi-Level Cell (MLC) flash makes recovery more difficult



Floating gate transistor

Managed Flash Considerations

- Managed flash (eMMC, NVMe) combines a controller with bulk flash
  - Supports increased performance and reliability
- Controller adds abstraction between logical and physical flash locations
  - Erase commands remap physical flash locations
    - Physical erasure done opportunistically
  - Bad blocks can be mapped out
- Complete erasure (zeroization) requires controller to cooperate
  - More flash to erase than what is advertised (spare cells)
  - Must break through the abstraction layer



Abstraction of logical to physical flash locations in managed flash

Remanence Mitigations

- Do not rely on power-off decay of volatile memory
  - Low temperatures are a threat
- Remember that SRAM can imprint
  - Prevent long-term static data if possible
- Pre-condition flash media
  - Repeated R/W cycles can reduce imprinting effects
- Prevent access to FPGA configuration state
  - Lock down JTAG and other interfaces
- Encrypt all external media
  - Any potentially sensitive information
- FPGA and all media should be zeroized
  - More complex than writing zeros

FPGA System Zeroization

MODULAR OPEN SYSTEMS APPROACH

- Platform agnostic zeroization for FPGA systems
  - Shim between application and media
    - Leverages existing media interface
  - Direct access to FPGA configuration
- Parallelized zeroization of all media
  - Operations customed to media type/size
- Selective zeroization of application
  - Zeroization infrastructure remains resident
- Goal: Drop-in integration
  - Configuration wizards
  - Encapsulation of application design
  - Integration with FPGA vendor toolchain
- Current SBIR Phase II program with DEVCOM GVSC is developing this solution