# Secure Interface Architecture for Charge Trap Transistor (CTT) Based EEPROM

Vishal Reddy Banala
School of Electrical and Computer Engineering
Oklahoma State University
Stillwater, USA
vbanala@okstate.edu

Cheng Hao
School of Electrical and Computer Engineering
Oklahoma State University
Stillwater, USA
cheng.hao@okstate.edu

Chris Hutchens
School of Electrical and Computer Engineering
Oklahoma State University
Stillwater, USA
c.hutchens@okstate.edu

***Abstract*—Cryptographic attacks and memory observability among DRAM or PROM have drawn researcher's attention. C. Kothandaraman et al. [1] have proposed a secure memory element using charge-trap based transistor for a potential memory application. Our work presents an on-chip interface architecture between CPU and secure EEPROM for control and data communication. This architecture allows the secure EEPROM to be embedded with the processing unit preventing interface eavesdropping so that encryption keys can be accessed locally and securely. It is also designed to overcome the cold-boot attacks and side channel attacks employing on-chip implementation and parallel data communication. The design is implemented in VHDL and validated with secure EEPROM model. The synthesis and simulation results of the design are presented.**

***Keywords—EEPROM; encryption; charge trapping; HfO$_2$; high-k dielectric; cyber security; programming.***

## I. INTRODUCTION

Encryption helps improving data security during transferring data. DRAM and PROM are commonly used storage means for encryption keys. Post power removal, their content will not be lost instantly due to a finite amount of discharge time [2]. The finite decay time allows retrieval of the secure content using simple techniques when memory is physically accessible. J. Halderman et al. [2] have demonstrated that DRAM loses its content over a period of seconds after cutting power or being removed from the motherboard at normal operating temperatures. O. Lo et al. [3] have successfully demonstrated side channel attacks (SCA) on serial data bus using power analysis. C. Kothandaraman et al. [1] have proposed a secure memory cell structure employing charge-trap based transistors. In addition to the secure cell structure, embedding memory core alongside the processor secures the content from interfacial physical attacks considering modern memory sizes [4]. This work developed an interface architecture for control and data communication between CPU and secure EEPROM that employs the charge-trap memory cells. Section II explains the functionality of CTT based differential storage cell. Section III introduces the overall architecture of the interface. Section IV discusses the write/read/erase operations of the interface. Simulation results are presented in Section V. Section VI is the conclusion.

This work is sponsored by Air Force Research Laboratory (AFRL), Rome, New York.

## II. EEPROM STORAGE CELL

CTT traps charge in the high-k dielectric layer when a high positive gate voltage, nominally 2V, is applied [1] [5] [6]. Whereas, it de-traps charge when a negative voltage, nominally -2V, is applied [1] [5] [6]. Each memory cell consists of a differential NMOS transistor pair, T1 and T2, whose gates are tied together forming a word line as shown in figure 1 [6]. Write operation is conducted by applying high gate voltage along with input data so that carriers trap into the dielectric of one side of the differential pair resulting in a threshold difference. This threshold difference results in current difference in Bit line and Bit line bar which is sensed by Sense Amplifier (SA) for a logic output. Erase operation is conducted by applying a negative gate voltage that de-traps the carriers from high-k dielectric layer [6] [7]. After carriers are removed from the dielectric layer, the threshold voltage of the programmed transistor returns to a value close to its native state. The erase residual is negligible compare to the programming shift [1].

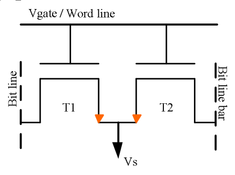


Figure 1. CTT based storage cell of the secure EEPROM [6].

## III. EEPROM INTERFACE ARCHITECTURE

The top level block diagram of the interface architecture is shown in figure 2. The interface consists of three input registers, finite state machine (FSM) controller, counters, clock divider, necessary combinational logic and command user interface (CUI) logic. The interface connects the EEPROM and CPU with four bus signal groups, namely data bus, R/W signal, address and clock. The R/W signal indicates CPU read or write and the address specifies input register. ECC functional block is reserved for future versions [6]. The 80-bit data is appended with 8 bits at the MSB side at the output of data register to support the ECC function. Transporting data parallelly into the interface and moving memory from off-chip to on-chip degrades signal-to-noise ratio (SNR) by 12 to 15 bits preventing SCA.

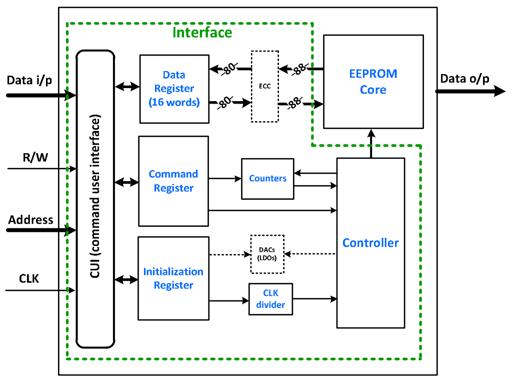


Figure 2. Interface architecture block diagram.

### A) *Input Registers*

The 80-bit initialization register consists of the CPU clock divider and programmable write/read/erase (W/R/E) voltages. Bit allocation of the initialization register is shown in figure 3. The remaining twenty bits are reserved for future use.

**Initialization Register**

| MSB | TBD 20 bits | Vm0p8 12 bits | VRead 12 bits | VErs 12 bits | VDD2 12 bits | CLK_scale 12 bits | LSB |
|---|---|---|---|---|---|---|---|
| | [79:60] | [59:48] | [47:36] | [35:24] | [23:12] | [11:0] | |

Figure 3. Initialization register bit allocation.

The 80-bit command register contains the EEPROM address, counter values for programming voltage stabilization time and application time for the write, read, and erase operations respectively, quenching time, a BUSY bit and command bits. Figure 4 shows the bit allocation of command register.

The BUSY bit permits either CPU or EEPROM to access the data register. The BUSY bit prohibits the CPU from writing the data register until a command execution is completed by the EEPROM. The address bits CMD[0:9] specify the word and bank address of the EEPPROM location(s) to be written/read/erased. Bits CMD[10:11] are reserved for future use. Voltage stabilization time bits, CMD[12:19], [28:35] and [44:51], are the time delays ensuring programming voltages are stable at the supply pins before their application to the EEPROM core. Voltage application time bits, CMD[20:27], [36:43] and [52:59], sets the time period that the programming W/R/E voltage is applied. The timeout bits, CMD[60:67], are provided for high voltage supply quenching during transitioning back to the idle state to avoid excess voltage application. Bits [68:70] are reserved for future use. CMD[72:79] specify the memory operation to be carried out. CMD[76:79] specify a block (16 words) when equal to xXf and a single word when equal to xX0. The type of operation is specified by CMD[72:75]. Write (W) operation is x1X, Erase (E) operation is x2X and Read (R) operation is x3X, respectively. All other combinations of CMD[72:79] are invalid for this interface and reserved for future use.

**Command Register**

| MSB | CMD 8 bits | BUSY 1 bits | TBD 3 bits | timeout 8 bits | R_Vapp 8 bits | V_stab_R 8 bits | E_Vapp 8 bits | V_stab_E 8 bits | W_Vapp 8 bits | V_stab_W 8 bits | TBD 2 bits | Address | | LSB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | Bnk 2 bits | Row 8 bits | |
| | word [76:79] CMD [72:75] | [71] | [68:70] | [60:67] | [52:59] | [44:51] | [36:43] | [28:35] | [20:27] | [12:19] | [10:11] | [0:9] | | |

Figure 4. Command register bit allocation.

The sixteen 80-bit words data register provides data buffering between CPU and EEPROM which is accessed by them in a mutually exclusive manner under the control of the BUSY bit. The R/W signal determines write or read access to the data register. The data register serves as an intermediate buffer between CPU and EEPROM to minimize the overhead time as the EEPROM operates much slower than the CPU.

### B) *Controller:*

The EEPROM interface operation is controlled by a finite state machine (FSM) whose state diagram is shown in figure 5. Write, read and erase operations are carried out in the similar way. For each operation, there are three critical states including programming voltage stabilization state, programming voltage application state and quenching state. The time period of each state is defined in the command register discussed in section II(A).

This controller also generates intermediate control signals W_in, E_in, R_in, WR, ER, RD and RD_DONE for s state definition and state transition. With respect to each operation, signal W_in, E_in, and R_in are active from voltage application state (W/E/R_Vapp) to quenching state (W/E/R_done). Signal WR, ER, and RD is true only during the “W/E/R_Vapp” state of the corresponding operation. RD_DONE is active upon Read completion and used to set the BUSY bit to a logic low so that the CUI returns access to CPU and interface control enters idle state.

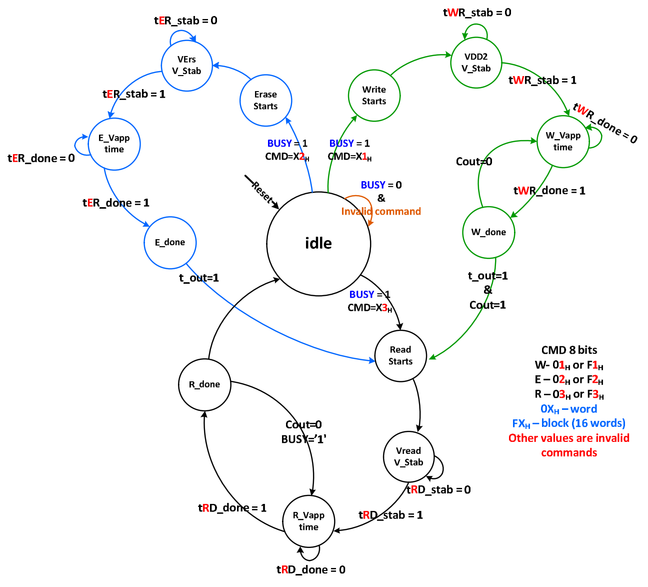


Figure 5. State diagram of the controller.

### C) *Command User Interface (CUI):*

The CUI controls the data traffic between CPU and input registers. Figure 6 presents the CUI circuit diagram. Registers can be accessed by applying the corresponding address specified by the two least significant bits of the CPU address, CPU_addr[0:1]. The CPU R/W bit indicates that CPU writes to the input registers when it is logic one and reads from the registers when it is logic zero. CPU and EEPROM access to the 16-word data register is mutually exclusive and depends on the status of the BUSY bit. This ensures CPU and EEPROM can never access the data registers simultaneously. Once the CPU completes access of the input registers, the BUSY bit is set and control is passed to the EEPROM. After EEPROM completes a W/R/E operation, the wrapper clears the BUSY bit returning the control back to the CPU. The CPU may read initialization and command registers independent of the BUSY bit status.

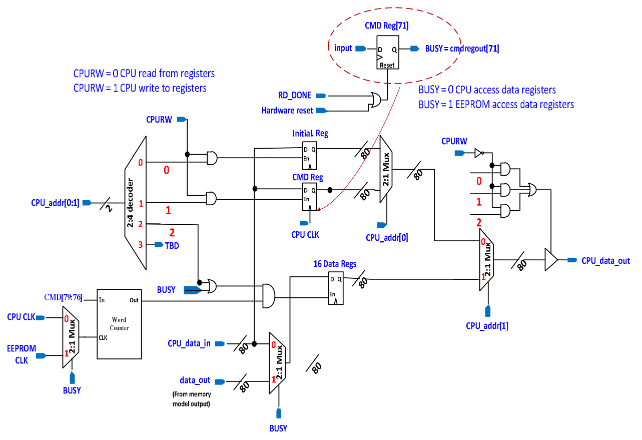


Figure 6. Command user interface schematic.

A 4-bits word counter tracks operation progress through the 16-word data register. For a single word, the word counter is loaded with value x0 via the command register bits CMD[76:79]. For a block of data, the word counter is loaded with value xF. The word counter is decremented as execution of data proceeds in the data register. The word counter is clocked by the CPU when CPU has access to the data registers or by the memory clock when EEPROM has access to the data registers. Memory clock is derived by dividing the CPU clock by the value in the initialization register bits [0:11].

## IV. Operation: Write, Erase and Read

Command register bits CMD[72:79] are x10 or x1F indicating write one word or a block (i.e.16) of words respectively. When an invalid command is issued, EEPROM interface returns to idle state and the CUI returns data register access to the CPU. The Command register address bit CMD[0:9] is decoded to allow access to the target address and bank. When writing a block of words, the initial address is decoded. The word counter in CUI is loaded with xF for a block write and x1 for a word write. For a block write, the EEPROM address counter increments to the next addresses sequentially after each word written. The word counter is decremented and the next word in the data register is written to the memory. This process repeats until the block of words are written to memory. This design only supports consecutive block access with 16-word boundaries. This means the starting block address must be an integer multiple of 16.

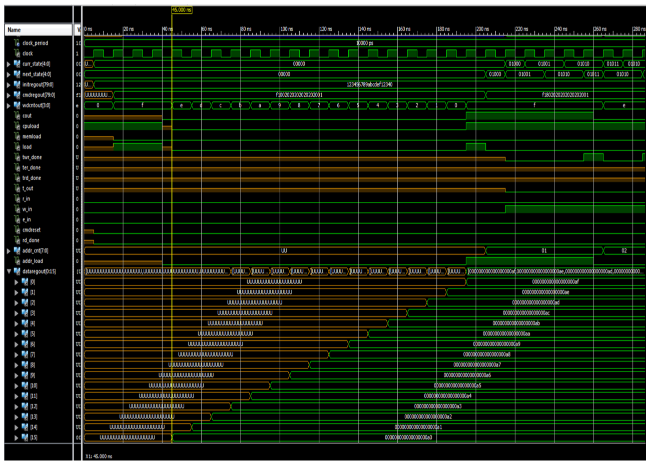
Erase operation is carried out in a similar manner as write but only on a block of words [8]. Erase operation begins by CPU loading the command register CMD[72:79] bits with x2F, BUSY bit with 0, correct timer values and address. Then command register is reloaded with BUSY bit equals 1 to start the erase operation and the word counter while other bits remained the same from previous load. To erase another block, CPU needs to reload the command register with new block starting address and erase command.

After the write or erase operation is complete, read operation automatically starts to confirm the write or erase result. Read operation begins by CPU loading the command register CMD[72:79] bits with x30 or x3F. The command bits CMD[0:9] are decoded for target address of the word and bank. The read result is written back to the data register for CPU retrieval and confirmation [6]. After read operation is completed, BUSY bit is set to 0 returning data register access back to CPU. The EEPROM interface then enters idle state. The CPU can retrieve the data from the data register or overwrite with new values.

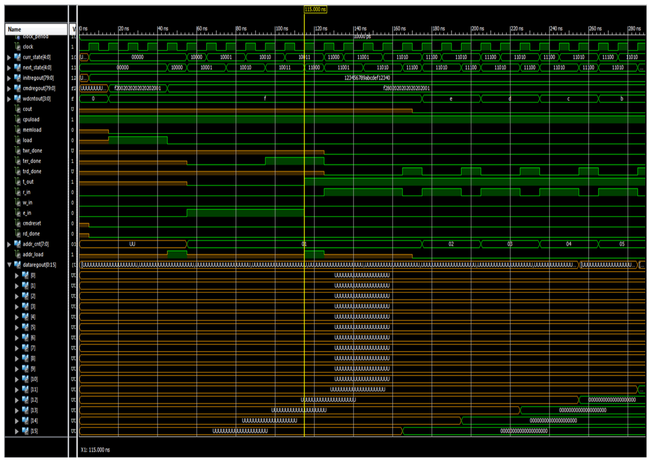
## V. Simulation Results

The EEPROM interface is logically checked against with an EEPROM memory model that we developed based on charge-trap memory cell operations. The model consists of four 256 by 88 arrays. Each array represents one EEPROM bank. Write/Read/Erase operation of the array is performed based on the control signals generated in the interface. A Xilinx simulator platform is used to execute VHDL code. The cursor guides the beginning of the operation in respective waveforms. Fig. 7(a) shows the waveform of writing 16 words from the data register to the EEPROM array. Fig. 7(b) shows the erase operation of a block (16 words) of words. Fig. 7(c) shows the waveform of reading 16 words to data registers for validation from CPU after performing write operation on EEPROM memory.

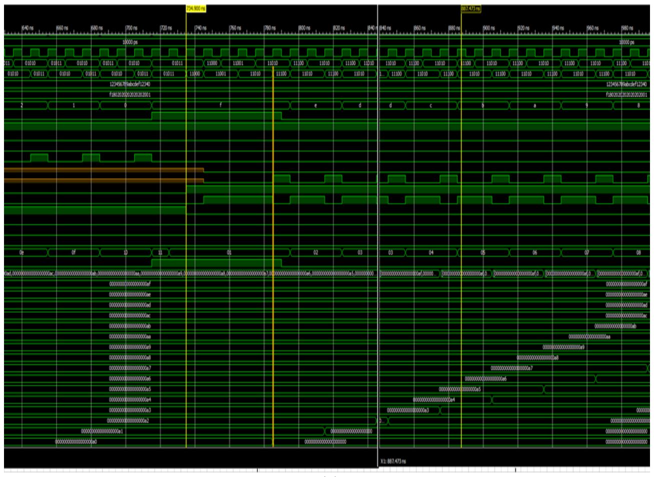
Synthesis report states that the interface architecture has a gate count of 90816 D-type flip-flops, 32 comparators and 352 multiplexers on macro level. Total synthetic cell area is noted as 255 square micrometers with total dynamic power consumption of 350.99nW. When compared with EEPROM memory area, the interface architecture area is negligible.



(a)



(b)



(c)

Fig. 7. (a) Simulation of writing 16 words from data register to EERPOM bank. (b) Simulation of erasing a block (16 words) of words. (c) Simulation of reading 16 words to data registers for validation from CPU after performing write operation on EEPROM memory.

## VI. Conclusion

This work presents the interface architecture that enables the charge-trap based secured EEPROM to be embedded with the processing unit. The embedment protects the secure content stored in EEPROM from physical cryptographic attacks and memory observability. The custom designed on-chip CPU-EEPROM interface provides programming control and data communication. In turn these guarantee secure communication with the outside world or full CPU data security. The on-chip implementation and parallel data bus secures the interface from SCA and eavesdropping. The interface is implemented and verified in VHDL with a gate count of 90816 registers and an area of 255 square micrometers after synthesis. Interface area is insignificant compared to the 65,000 square micrometers for the EEPROM.

## Acknowledgement

Author thanks Dr. Chris Hutchens for the technical guidance throughout the work. Also, thanks Mr. Cheng Hao for his assistance on analog circuit design. Special thanks to AFRL for funding this project.

## References

[1] C. Kothandaraman et al, “Oxygen vacancy traps in Hi-K/Metal gate technologies and their potential for embedded memory applications,” 2015 IEEE Internatioinal Reliability Physics Symposium, Monterey, CA, pp. MY.2.1-MY.2.4., 2015.

[2] J. Halderman et al., “Lest we remember: cold-boot attacks on encryption keys,” Communications of the ACM, vol. 52(5), pp. 91–98, May 2009.

[3] Owen Lo et al., “Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power naalysis (CPA),” Journal of Cyber Security Technology 2017, 1:2, 88-107.

[4] Saeed Mohsenifar, M. H. Shahrokhabadi., "Gate Stack High-κ Materials for Si-Based MOSFETs Past, Present, and Futures," in Microelectronics and Solid State Electronics 2015, 4(1): 12-24.

[5] F. Khan, E. Cartier, J. C. S. Woo, and S. S. Iyer, “Charge trap transistor (CTT): an embedded fully logic-compatible multiple-time programmable non-volatile memory element for high-k-metal-gate CMOS technologies,” IEEE Electron Device Letters, vol. 38, no. 1, pp. 44-47, Jan. 2017.

[6] B. Jayaraman et al., “80-kb logic embedded high-k charge trap transistor-based multi-time-programmable memory with no added process compliexity,” IEEE Journal of Solid-State Circuits, vol. 53, no. 3, pp. 949-960, March 2018.

[7] Chun Zhao *et al*., "Review on Non-Volatile memory with High-k dielectrics: Flash for generation beyond 32nm,” in High-k materials and devices, Materials 2014, MDPI.

[8] Sergei skorobogatov, “Local heating attacks on flash memory devices,” 2009 IEEE International Workshop on Hardware-Oriented Security and Trust (HOST).