Insights into the Potential Usage of the Initial Values of DRAM Arrays of Commercial Off-the-Shelf Devices for Security Applications

Nikolaos Athanasios Anagnostopoulos[*], André Schaller[*], Yufan Fan[*], Wenjie Xiong[†], Fatemeh Tehranipoor[‡], Tolga Arul[*], Sebastian Gabmeyer[*], Jakub Szefer[†], John A. Chandy[‡] and Stefan Katzenbeisser[*]

[*] Security Engineering Group (CRISP-CYSEC), Technische Universität Darmstadt
[†] Computer Architecture and Security Lab, Yale University
[‡] Department of Electrical and Computer Engineering, University of Connecticut

Several cryptographic applications entail the availability of a secure storage on a device, for instance, to store secret keys. Physical Unclonable Functions (PUFs) can be used to provide such key storage on commodity devices in a cost-efficient manner [KKR+12]. Their security is based on the existence of at least one (random but stable) output that is unique per device for some given input. Recently, a number of different PUF implementations based on DRAMs have been proposed [SXA+17, TKXC15, TKYC17, XSA+16]. We draw motivation from these recent publications in order to investigate the potential of the initial values of DRAMs found in commercial off-the-shelf devices to be used for the implementation of a PUF.

For this purpose, we test the DRAM arrays of two evaluation platforms, i.e. Intel Galileo Gen. 2 and PandaBoard ES Rev. B3. The Intel Galileo platform features a 256 (2×128) MB DDR3 SDRAM, with a row size of 16 KB, whereas the PandaBoard contains a 1 GB Low Power (LP)DDR2 SDRAM with a row size of 32 KB. To access the values of the DRAM cells, we modify the Quark EDKII firmware on the Galileo board and the U-Boot bootloader on the PandaBoard. For enabling better insights, we obtain the initial values of the DRAM at two different positions in the progression of the boot process of each evaluation platform.

On the Intel Galileo board, code position *GP1* marks the state during boot when the DRAM has been completely set up, but has not yet been written to by any user code. At this state, however, only the refresh function has been set up, while the error correction function has not yet been set

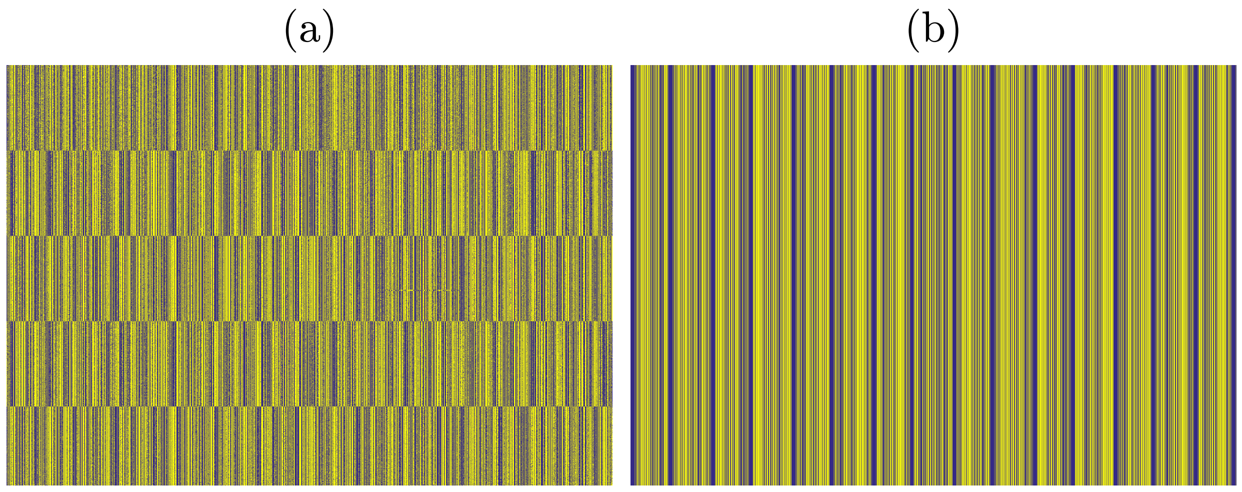

Figure 1: Patterns observed in the initial values of the DRAM of the Galileo.

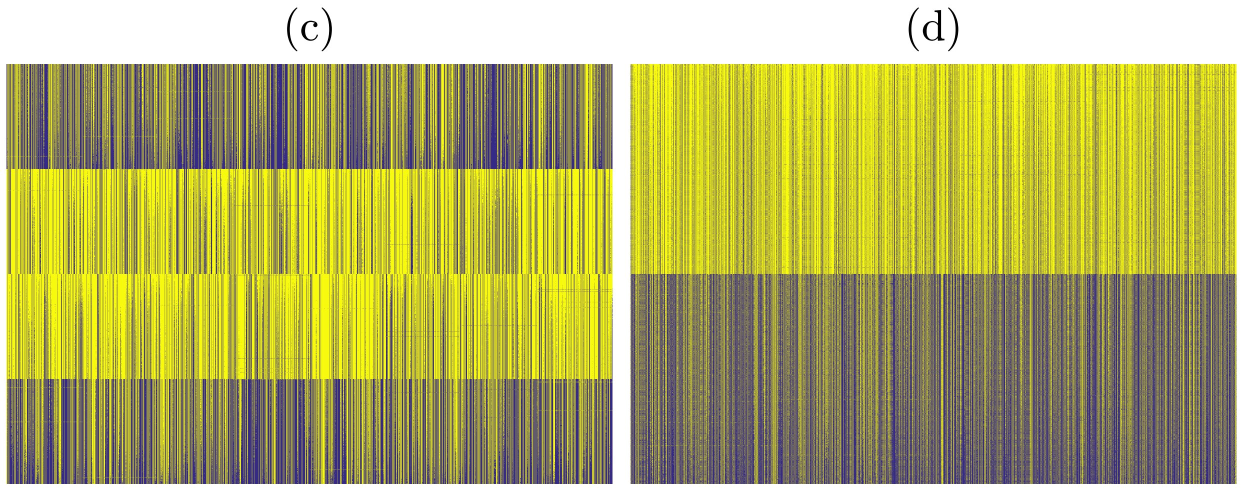

Figure 2: Patterns observed in the initial values of the DRAM of the PandaBoard.

up. Code position *GP2* refers to a prior state when the system has just enabled access to the DRAM, after setting up and initialising its addressing system. At code position *GP1*, we observe distinct pattern segments, one after the other, as shown in Figure 1a. The first four segments exhibiting a pattern have a length of 416 rows, whereas the fifth one is only 384 rows long. This behaviour is repeated in the following segments. At code position *GP2*, all of the memory follows a single pattern as shown in Figure 1b. Values at both code positions exhibit different patterns and noise between measurements, whereas changes in the firmware alter observed patterns even more radically.

On the PandaBoard, code position *PP1* marks the state during boot when the system has almost finished setting up the DRAM and all its functions, but the DRAM has not yet been written to by any user code, while, code position *PP2* refers to a prior state when the system has just enabled serial communication for transferring the initial values and only a low-level setup of the DRAM has occured, i.e. system access to the DRAM array has been enabled and its addressing system has been set up and initialised. In both cases, we observe distinct pattern segments, one after the other. However, slight modifications of the PandaBoard's U-Boot bootloader can change the segment length, e.g., from 512 rows (Figure 2c) to 1024 rows (Figure 2d).

Further analysis using relevant metrics, such as Hamming weight and intra- and inter-Hamming distances, confirms that the observed patterns prevent the usage of the obtained boot-up values of these commercial DRAMs as a PUF. Nevertheless, these patterns may provide insights into the physical layout of the DRAM arrays and into the relation between physical and logical addresses. Future research may enable access to the values of fully uninitialised cells of commercial DRAMs, which may then prove useful for security applications.

References

[KKR$^+$12] Stefan Katzenbeisser, Ünal Kocabaş, Vladimir Rožić, Ahmad-Reza Sadeghi, Ingrid Verbauwhede, and Christian Wachsmann. PUFs: Myth, fact or busted? A security evaluation of Physically Unclonable Functions (PUFs) cast in silicon. In *Cryptographic Hardware and Embedded Systems–CHES 2012*, pages 283–301. Springer, 2012.

[SXA$^+$17] André Schaller, Wenjie Xiong, Nikolaos A. Anagnostopoulos, Muhammad Umair Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer. Intrinsic rowhammer PUFs: Leveraging the rowhammer effect for improved security. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2017.

[TKXC15] Fatemeh Tehranipoor, Nima Karimian, Kan Xiao, and John Chandy. DRAM-based intrinsic physical unclonable functions for system level security. In *25th Great Lakes Symposium on VLSI*, pages 15–20. ACM, 2015.

[TKYC17] Fatemeh Tehranipoor, Nima Karimian, Wei Yan, and John A. Chandy. DRAM-based intrinsic physically unclonable functions for system-level security and authentication. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(3):1085–1097, 2017.

[XSA$^+$16] Wenjie Xiong, André Schaller, Nikolaos A. Anagnostopoulos, Muhammad Umair Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer. Run-time accessible DRAM PUFs in commodity devices. In *Cryptographic Hardware and Embedded Systems–CHES 2016*, volume 9813 of *Lecture Notes in Computer Science (LNCS)*, pages 432–453. Springer, 2016.