Driving Against the Memory Wall: The Role of Memory for Autonomous Driving

Matthias Jung
Fraunhofer Institute for Experimental Software Engineering (IESE)
Kaiserslautern, Germany
Email: matthias.jung@iese.fraunhofer.de

Norbert Wehn
Microelectronic Systems Design Research Group
University of Kaiserslautern
Kaiserslautern, Germany
Email: wehn@eit.uni-kl.de

***Abstract*—Autonomous driving is disrupting the conventional automotive development. In fact, autonomous driving kicks off the consolidation of control units, i.e. the transition from distributed *Electronic Control Units* (ECUs) to centralized domain controllers. Platforms like Audi's zFAS demonstrate this very clearly, where GPUs, Custom SoCs, Microcontrollers, and FPGAs are integrated on a single domain controller in order to perform sensor fusion, processing and decision making on a single *Printed Circuit Board* (PCB). The communication between these heterogeneous components and the algorithms for *Advanced Driving Assistant Systems* (ADAS) itself requires a huge amount of memory bandwidth, which will bring the *Memory Wall* from *High Performance Computing* (HPC) and data-centers directly in our cars. In this paper we highlight the roles and issues of *Dynamic Random Access Memories* (DRAMs) for future autonomous driving architectures.**

I. Introduction

Automotive industry is currently undergoing major changes with respect to E/E architectures for enabling autonomy. First, the current approach of hundred distributed ECUs which are connected e.g. by CAN does not scale and it is not cost efficient anymore. Therefore, there is a trend from distributed ECUs to a dozen of consolidated domain controllers, which are connected e.g. by Ethernet. This transition strongly disrupts the current automotive development. Second, there is a convergence of mainstream and mission-critical markets [1]. Due to cost and performance reasons, processing elements like *Graphic Processing Units* (GPUs), which were originally developed for the consumer sector, are now also considered for safety critical applications. Third, more and more data coming from various perception sensors has to be processed in real-time in order to guarantee the functionality of advanced driving assistant systems.

As a consequence, we will also see heterogeneous computing platforms similar to embedded consumer devices in future automotive applications. For instance, Audi presented its A8 car recently, which features an advanced autonomy level-3 driver assistance platform called *"zentrales Fahrerassistenzsteuergerät"* (zFAS) developed by TTtech. This platform is Audi's entry point for autonomy and clearly highlights the major changes above. Compared to classical automotive solutions, which consist of distributed ECUs, Audi has decided to develop their zFAS system as a consolidated platform on a single PCB. The reason for that is that the main task of this system is sensor fusion and in many cases the computation on raw sensor data is more accurate compared to the processing of meta data (like object lists), which are computed from preprocessing ECUs [2].

In particular, zFAS features [2]:

- An NVIDA K1 GPU for 360° image processing
- An Intel Mobile Eye *System on Chip* (SoC) for detection of traffic signs, pedestrians, collisions, light and lanes
- An Intel Altera Cyclone *Field Programmable Gate Array* (FPGA) for object and map fusion, parking pilot and data pre-processing
- An Infineon Aurix CPU for assistance systems like the level-3 traffic jam pilot

A major task of future ADAS platforms with autonomy level-4 and level-5 is the inference of *Neuronal Networks* (NN). A look in today's HPC data-centers shows that instead of GPUs even *Application Specific Integrated Circuits* (ASICs) are applied for the efficient processing of NNs. This leads to the conclusion that ASICs and heterogeneous SoCs with custom NN hardware accelerators will be used in future automotive platforms as well.

However, in all the discussions on automotive electronics, the aspect of memory was not yet addressed sufficiently [3]. The ever increasing amount of large data-sets that have to be processed in real-time by the heterogeneous compute platforms must be buffered in large, fast and high endurance consumer memories like DRAMs, which leads to several challenges, which are discussed in the following sections of this paper.

II. Challenges Related to DRAM Memory

A. Bandwidth Challenge

A very prominent example for a custom NN ASIC from the HPC world is Google's *Tensor Processing Unit* (TPU). Google showed, that four out of six studied neural networks are bandwidth-limited by the DRAM memory [4]. Due to slow memory accesses the computational units are idling. This shows that a fast execution of NNs is not only guaranteed by performant application specific computational units but also the memory subsystem must be designed wisely in an application specific way in order to avoid hitting the so called *Memory Wall* [5]. While automotive is mainly focusing on perception

with *Sensors*, *Processing* and car to car *Communication* the problem of the memory wall for artificial intelligence is currently not addressed and probably totally underestimated. Current automotive applications require a DRAM bandwidth of less than 60 GB/s, which can be provided easily with standard DDR and LPDDR solutions. However, advanced ADAS applications for autonomy level-4 and level-5 will require up to 1024 GB/s memory bandwidth [3], which is hard to realize with standard DDR and LPDDR technologies. Therefore, automotive must concentrate on other DRAM solutions like *Graphic DDR* (GDDR) and *High Bandwith Memory* (HBM) in order to cope with these high bandwidth requirements. For example, four parallel HBM2 memories can provide 1024 GB/s – however, this bandwidth is just a theoretical maximum. The available sustainable memory bandwidth is often much less and strongly depends on how the data is stored in the memories, i.e. the memory access pattern. Therefore, *Application Specific Memory Controllers* are required in order to keep up the sustainable bandwidth [6].

B. Non-Deterministic Timing Behavior of DRAMs

DRAMs have a non-deterministic timing behavior [7], which makes it difficult to provide predictable performance, and thus guarantee the timing predictability of tasks [8]. Therefore, the automotive community has so far largely waived the usage of DRAM for safety critical and real-time applications. For example, Infineon's Aurix controller, which is widely used for safety-critical applications, does not provide a DRAM memory controller. Because of the requirement for large memories and the mentioned convergence of mainstream and mission-critical markets, automotive is forced to focus on this type of memory.

C. Reliability vs. Termperature

DRAMs are very sensitive to high temperature, which increases the leakage at the cells. In order to avoid data corruption by retention errors, the refresh frequency needs to be increased. However, the operating temperature for automotive applications is specified between - 40°C up to 125°C. Therefore, the refresh period must be decreased from 64 ms to $4-8$ ms in the worst case, which leads to a serious collapse of the sustainable bandwidth [9].

D. Security Challenge

As DRAM process technology scales down, the electrical interference between the memory cells increases, which leads to disturbance errors. Recently, the *Row Hammer* problem [10] and its exploit [11] have caused a lot of attention in research and newspapers. By repeatedly opening and closing a DRAM row, called *Hammering*, bits in adjacent rows can flip. This effect can be exploited to write on memory locations with prohibited access rights to e.g. gain kernel privileges or escape a sandbox or hypervisor. Because automotive transitions to open environments for *Car2Car* and *Car2X* communication, the vulnerability of DRAM must be considered.

E. Safety Critical Applications

Furthermore, the communication between computational units within a domain controller will be realized with shared memory. The challenge is to establish a clear segregation between safety-critical e.g. real-time applications and non-safety-critical applications running on the same platform. Existing off-the-shelf real-time DRAM controllers, however, do not support mixed criticality and therefore allow interference between applications [12].

References

[1] S. Girbal, M. Moret, A. Grasset, J. Abella, E. Quiones, F. J. Cazorla, and S. Yehia. On the convergence of mainstream and mission-critical markets. In *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–10, May 2013.

[2] Ingo Kuss. Audi zFAS - Enorme Datenmengen bewältigen. http://www.elektroniknet.de/elektronik-automotive/assistenzsysteme/enorme-datenmengen-bewaeltigen-131797.html, July 2016.

[3] Sven Evers. Cinco-play: Memory is that critical to autonomous driving. https://www.micron.com/about/blogs/2017/october/cinco-play-memory-is-that-critical-to-autonomous-driving, November 2017.

[4] Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Salek, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, ISCA '17, pages 1–12, New York, NY, USA, 2017. ACM.

[5] Wm. A. Wulf and Sally A. McKee. Hitting the memory wall: implications of the obvious. *SIGARCH Comput. Archit. News*, March 1995.

[6] Matthias Jung, Irene Heinrich, Marco Natale, Deepak M. Mathew, Christian Weis, Sven Krumke, and Norbert Wehn. ConGen: An Application Specific DRAM Memory Controller Generator. In *International Symposium on Memory Systems (MEMSYS 2016)*, 2016.

[7] S. Goossens, K. Chandrasekar, B. Akesson, and K. Goossens. *Memory Controllers for Mixed-Time-Criticality Systems: Architectures, Methodologies and Trade-offs*. Embedded Systems. Springer International Publishing, 2016.

[8] Ankit Agrawal and Gerhard Fohler. DRAM-related Challenges in Task Scheduling with Timing Predictability on COTS Multi-cores for Safety-critical Systems. In *Proceedings of the International Symposium on Memory Systems*, MEMSYS '17, pages 265–267, New York, NY, USA, 2017. ACM.

[9] MohammadSadegh Sadri, Matthias Jung, Christian Weis, Norbert Wehn, and Luca Benini. Energy Optimization in 3D MPSoCs with Wide-I/O DRAM Using Temperature Variation Aware Bank-Wise Refresh. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–4, March 2014.

[10] Yoongu Kim, R. Daly, J. H. Kim, C. Fallin, Ji Hye Lee, Donghyuk Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 361–372, June 2014.

[11] Mark Seaborn and Thomas Dullien. Exploiting the DRAM rowhammer bug to gain kernel privileges. http://googleprojectzero.blogspot.de/2015/03/exploiting-dram-rowhammer-bug-to-gain.html, March 2015.

[12] Leonardo Ecco, Sebastian Tobuschat, Selma Saidi, and Rolf Ernst. A mixed critical memory controller using bank privatization and fixed priority scheduling. In *RTCSA*, pages 1–10. IEEE Computer Society, 2014.