# Computer Architecture HW2

Fabian Wüthrich

October 16, 2020

## 1 Critical Paper Reviews [1000 points]

see here

## 2 RowHammer [200 points]

### 2.1 RowHammer Properties

a) True

b) False

c) True (i.e. protector cells)

d) True (i.e. aggressor or protector cells)

e) True

### 2.2 RowHammer Mitigations

a) If the refresh interval is reduced from 64ms to 8ms, each row is refreshed 8 more times. Thus, bank utilization increases to $8U$ and energy consumption increases to $8E$

b) With a doubling of the rows, the mitigation is still possible, but the bank is occupied 80% by refresh operations ($U = 0.05 \times 8 \times 2 = 0.8$).

With another doubling of the rows, the mitigation cannot be implemented because we cannot refresh every row in 8ms ($U = 0.05 \times 8 \times 4 = 1.6$).

c) No, we need to know which rows are adjacent.

d) With a 8ms refresh interval an attacker can issue a limited number of activations, constrained by $t_{RC}$. If $T$ is less than the maximum number of activations issued in 8ms, we can guarantee the same level of security. We have $t_{RC} = t_{RAS} + t_{RP} = 35ns + 13.5ns = 48,5ns$ and therefore

$$T = \frac{8ms}{48.5ns} = 164948.4536 \approx 164948$$

e) A single counter requires $log_2(164948) = 17.3317 \approx 18$bits and each row needs a counter. Thus,

$$18\,bits/row \times 2^{15}\,rows/bank \times 8\,banks \times 2\,ranks = 9Mbit$$

When the number of rows per bank and the number of banks per chip are doubled we get

$$18\,bits/row \times 2^{16}\,rows/bank \times 16\,banks \times 2\,ranks = 36Mbit$$

f) The memory controller performs unnecessary activations which is bad for performance and energy consumption.

g) Let X be a RV that describes the number of errors in a year. Each 64ms interval can be seen as a independent experiment of getting an error. Therefore, X has a binomial distribution with the parameters $p = 1.9 \cdot 10^{-22}$ and $n = \frac{365 \cdot 24 \cdot 3600s}{64ms} = 492'750'000$. Therefore,

$$\begin{aligned} P(X \geq 1) &= 1 - P(X = 0) \\ &= 1 - (1-p)^n \\ &= 9.3622 \cdot 10^{-14} \end{aligned}$$

# 3 Processing in Memory: Ambit [200 points]

## 3.1 In-DRAM Bitmap Indices I

a) All users occupy $\frac{u}{8}$bytes so we need $\frac{u}{8 \cdot 8k}$ subarrays. Including the weeks, $\frac{u \cdot w}{8 \cdot 8k}$ rows are occupied.

b) Using Ambit we copy each row into the *Operand* row and perform a bulk `and`. This requires $\frac{u \cdot w}{8 \cdot 8k} \times (t_{rc} + t_{and})$ seconds overall. Then we need to transfer $\frac{u}{8}$bytes to the CPU and count the bits. This takes $\frac{u}{8X}$ seconds. Therefore, the throughput is

$$\frac{u}{\frac{u \cdot w}{8 \cdot 8k} \times (t_{rc} + t_{and}) + \frac{u}{8X}} \, users/second$$

c) The CPU has to transfer all users and weeks from memory which takes $\frac{uv}{8X}$ seconds. Therefore, the throughput is

$$\frac{u}{\frac{uv}{8X}} = \frac{8X}{w} \, users/second$$

d) We want that the execution time on the CPU is lower than in memory. Therefore,

$$\frac{uv}{8X} < \frac{u \cdot w}{8 \cdot 8k} \times (t_{rc} + t_{and}) + \frac{u}{8X}$$

Solving the inequality for $w$ gives

$$w < \frac{1}{1 - \frac{X}{8k} \times (t_{rc} + t_{and})}$$

## 3.2 In-DRAM Bitmap Indices II

a)

# 4 In-DRAM Bit Serial Computation [200 points]

# 5 Caching vs. Processing-in-Memory [200 points]