# EECS 151/251A Homework 9

Due Sunday, April 15$^{\text{th}}$, 2018

## Problem 1: DDCA Exercise 8.12 :)

You are building an instruction cache for a MIPS processor. It has a total capacity of $4\text{C} = 2^{c+2}$. It is $N = 2^n$-way set-associative ($N \geq 8$), with a block size of $b = 2^{b'}$ bytes ($b \geq 8$). Give your answers to the following questions in terms of these parameters:

(a) Which bits of the address are used to select a word within a block?

(b) Which bits of the address are used to select the set within the cache?

(c) How many bits are in each tag?

(d) How many tag bits are in the entire cache?

TODO(aryap): Solutions.

## Problem 2: DDCA Exercise 8.13 (a) - (c)

Consider a cache with the following parameters:

- Associativity, $N = 2$
- Block size, $b = 2$ words
- Word size, $W = 32$ bits
- Cache size, $C = 32$ K words
- Address size, $A = 32$ bits

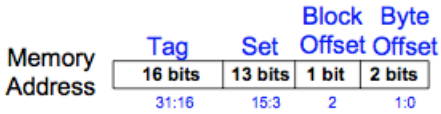
You need only consider word addresses.

(a) Show the tag, set, block offset, and byte offset bits of the address. State how many bits are needed for each field.

(b) What is the size of *all* the cache tags in bits?

(c) Suppose each cache block also has a valid bit ($V$) and a dirty bit ($D$). What is the size of each cache set, including data, tag, and status bits?

(a)




(b) Each tag is 16 bits. There are 32Kwords / (2 words / block) = 16K blocks and each block needs a tag: $16 \times 16K = 2^{18} = 256$ Kbits of tags.

(c) Each cache block requires: 2 status bits, 16 bits of tag, and 64 data bits, thus each set is 2 × 82 bits = **164 bits**.

(d) The design must use enough RAM chips to handle both the total capacity and the number of bits that must be read on each cycle. For the data, the SRAM must provide a capacity of 128 KB and must read 64 bits per cycle (one 32-bit word from each way). Thus the design needs at least 128KB / (8KB/RAM) = 16 RAMs to hold the data and 64 bits / (4 pins/RAM) = 16 RAMs to supply the number of bits. These are equal, so the design needs exactly 16 RAMs for the data.

For the tags, the total capacity is 32 KB, from which 32 bits (two 16-bit tags) must be read each cycle. Therefore, only 4 RAMs are necessary to meet the capacity, but 8 RAMs are needed to supply 32 bits per cycle. Therefore, the design will need 8 RAMs, each of which is being used at half capacity.

With 8Ksets, the status bits require another 8K × 4-bit RAM. We use a 16K × 4-bit RAM, using only half of the entries.

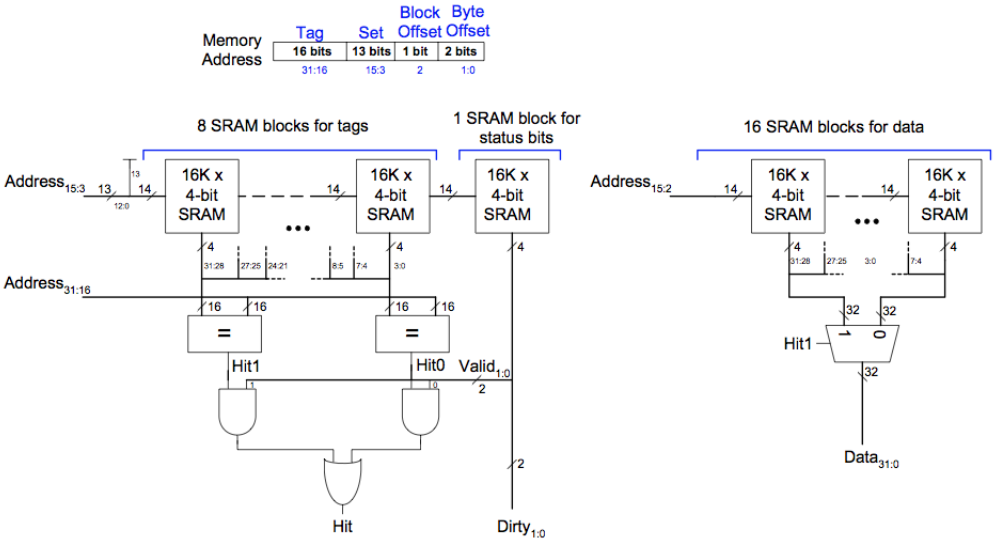



FIGURE 8.2 Cache design for Exercise 8.12

Bits 15:2 of the address select the word within a set and block. Bits 15-3 select the set. Bits 31:16 of the address are matched against the tags to find a hit in one (or none) of the two blocks with each set.

## Problem 3: True or False?

(a) The truth-table for an n-bit *carry-save adder* has exactly $2^{2n+1}$ rows.

False

(b) Using only 2 and 3-input CMOS logic gates, with the cost per transistor equal to 1, the minimum cost for the carry function in a full-adder cell is 16.

False

(c) In a 12-bit carry-select adder, grouping the bits into 3 groups of 4 bits each (4-4-4) results in lower overall delay than with any other arrangement of group sizes. Assume that the delay through a 2-input multiplexor is the same as the delay through a full-adder cell.

False

(d) Consider the design of a "2-bit-wide" serial adder used for adding 2 N-bit numbers (N is guaranteed to be even and at least 2). The 2-bit-wide adder is similar to the bit-serial adder from class, except that it adds *two* bit positions from each input within one cycle, instead of one; therefore it might need a longer clock period. Your design can only use flip-flops ($\tau_{clk-to-q} = \tau_{setup} = 1ns$), and full-adder cells ($\tau_{full-adder-cell} = 2ns$).

On N-bit addition, using the 2-bit design, it is possible to achieve an add *latency* (in $ns$) less than that for the normal serial adder.

True

(e) As a function of the number of input bits, N, the cost in terms of total transistors of a carry-lookahead adder is $\propto N + log_2(N)$.

False

(f) You are requested to perform a multiplication of 2 N-bit numbers, but your multiplication circuit only accepts N/2 bit wide inputs. You also have an adder circuit (of whatever width needed). The total number of times you will need to use the multiplier for the N * N multiplication is 3.

False

## Problem 4: The Multiplier in a Homework

(a) Draw a wallace tree for a 5 x 5 multiplier using Full Adder and Half Adder cells. What is the critical path?

Showing the steps for the dot diagram:

Note that you can use a fast (carry-bypass, carry-select, etc.) type of adder for the last stage instead of the ripple-carry.

The critical path for the above diagram is shown below, and the delay is $t_d = t_{HA} + 7 * t_{FA}$. Note that there are different implementations for this problem, so if you used e.g. a fast adder in the final stage, your critical path may be shorter.

(b) Add one pipeline stage to improve the throughput as much as possible. What is the new critical path?

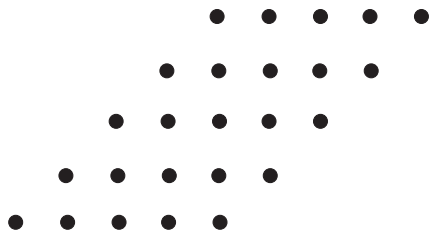


Figure 1: Original organization of partial products. Note that each dot represents a partial product.

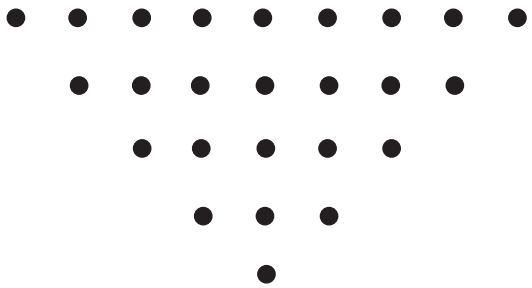


Figure 2: We rearrange the partial products in order to make grouping easier.

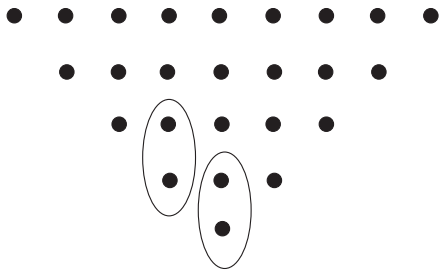


Figure 3: The circles show the first two groups, i.e. the first stage of the tree. We have two groups of two dots each and no carries so far, so we need two half adders for the first stage. After the first stage evaluates, the generated carries will pass to the left, and appear as dots on the next dot diagram.

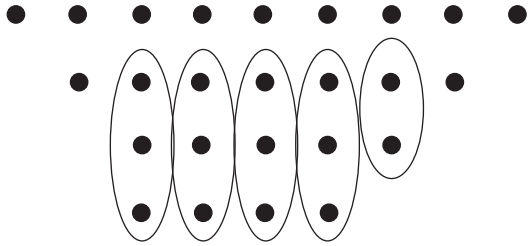


Figure 4: Now we can group the dots for the second stage, including the carries generated before. For every group of 3 we will use a FA, and we will use a HA for the group of 2. Again, the generated carries from each column will appear as a dot on the left column in the next diagram.

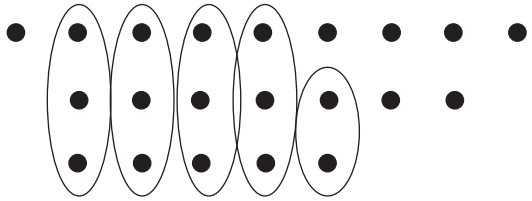


Figure 5: We keep grouping in a similar manner.

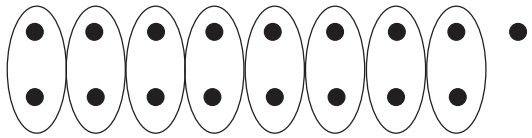


Figure 6: This is the final stage. Note that in this stage we will have a carry propagate-adder. This means that only the right-most adder will be a half adder, and all the others will be full adders (adding the two dots, plus the carry-out of the previous stage).

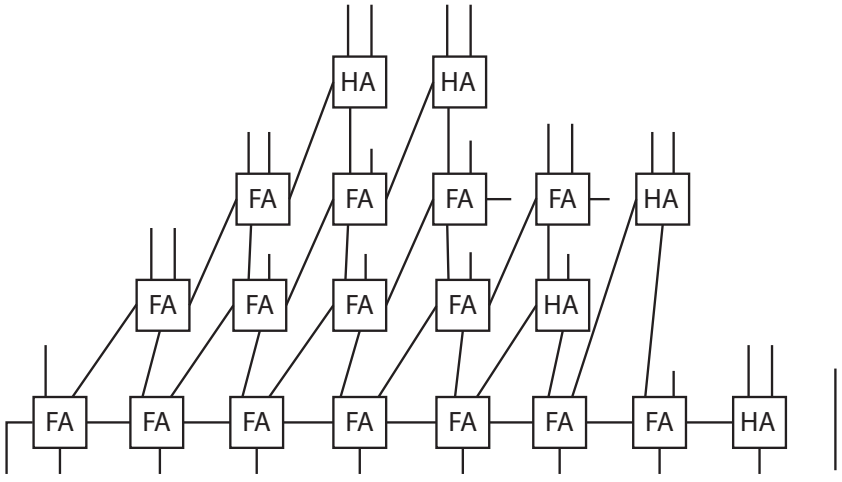
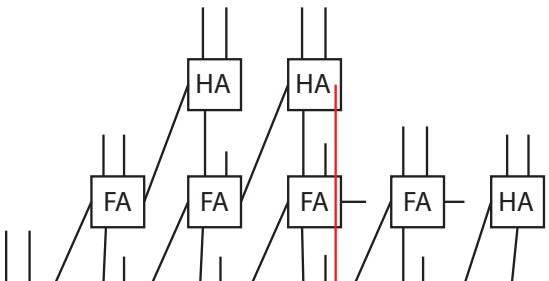



Figure 7: Showing this using Full Adders and Half Adders



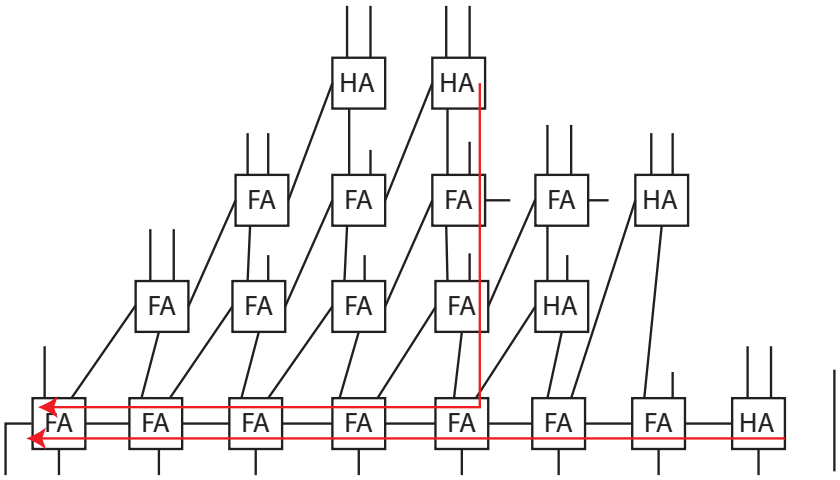



Figure 8: The critical path.

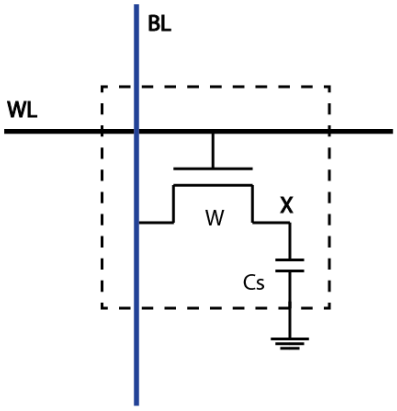
The new critical path is shown above, and is $t_d = 4 * t_{FA}$.

## Problem 5: The Multiplier in an Exam

Consider the design of a combinational **signed** 2-bit multiplication circuit. It takes as inputs two 2's complement integers, A and B, and outputs the appropriate number of result bits. Using full-adder cells (FA) and simple logic gates, draw a circuit that implements this function.

## Problem 6: DRAM Design

The figure below shows a $1\mu m \times 1\mu m$ DRAM cell with NMOS width $W = 0.5\mu m$ and $Cs = 55fF$. Assume that $C_D = C_G = 1fF/\mu m$, $V_{DD} = 1V$, $V_{TH} = 0.2V$, $I_{leak,n} = 1nA$ for a $W = 0.5\mu m$ device, that the cell is arranged in a $128 \times 8$ memory block and the wordline and bitline wires have a capacitance of $C_w = 0.2fF/\mu m$ (i.e. 0.2fF per unit length). Each bitline is connected to a sense amplifier with $\Delta V_{sense} = 100mV$ and input capacitance 5fF, and you may ignore loading from any other peripheral circuitry. Use the ideal switch model for the transistor.



(a) WRITE: Assume the cell initially stores a "0". In order to write a "1", BL and WL are raised to $V_{DD}$. What is the final value on node X? Recommend a way to reduce the charge loss on node X without changing the cell size. Discuss the trade-offs.

$$V_{BIT1} = V_{DD} - V_{TH} = 0.8V$$

One way to reduce the charge loss on node X is to increase $V_{DD}$ on the wordline. This typically involves adding extra peripheral circuity to boost the wordlines to a higher voltage. Additional circuitry and increased $V_{DD}$ both increase power dissipation. Another way is to use low-$V_{TH}$ devices, if that is allowed by the given technology. However, this increases the leakage current, affecting both the power dissipation and the refresh frequency for the memory (more in part c).

(b) READ: Going back to the original array (i.e. ignoring any alterations you recommended in part a), after storing values in your memory you pre-charge the bitline to $V_{DD}/2$ and read. What is the final value on the bitline when you are reading a "1" and when you are reading a "0"?

The total capacitance on the bitline is:

$$C_{BL} = C_{wire} + C_{d,nmos,tot} + C_{in,SA} =$$

$$= 0.2\frac{fF}{\mu m} \cdot (1\mu m \cdot 128) + 128 \cdot 1\frac{fF}{\mu m} \cdot 0.5\mu m + 5fF = 94.6fF$$

The total capacitance on node X is:

$$C_X = C_{d,nmos} + C_s = 1\frac{fF}{\mu m} \cdot 0.5\mu m + 55fF = 55.5fF$$

Now we apply conservation of charge:

$$Q_{init} = Q_{fin} \Rightarrow$$

$$V_{BIT}C_X + V_{PRE}C_{BL} = V_{BL}(C_X + C_{BL}) \Rightarrow$$

$$V_{BL} = V_{BIT}\frac{C_X}{C_X + C_{BL}} + V_{PRE}\frac{C_{BL}}{C_X + C_{BL}}$$

So for reading "1":

$$V_{BL,1} = 0.8V\frac{55.5fF}{55.5fF + 94.6fF} + 0.5V\frac{94.6fF}{55.5fF + 94.6fF} \approx 610mV$$

And for reading "0":

$$V_{BL,0} = 0V\frac{55.5fF}{55.5fF + 94.6fF} + 0.5V\frac{94.6fF}{55.5fF + 94.6fF} \approx 315mV$$

(c) RETENTION: Calculate how often we need to refresh the memory in order to maintain its contents during retention. You may assume that the bitline is held at $V_{DD}/2$.

In order to maintain proper operation we need to ensure that the sense-amplifier will always have at least $100mV$ of $\Delta V$ at its input. From the conservation of charge we have:

$$\Delta V = V_{BL} - V_{PRE} = (V_{BIT} - V_{PRE})\frac{C_X}{C_X + C_{BL}}$$

From part b we can see that the critical case is when we are trying to read a "1", and that is caused by the $V_{TH}$ voltage drop (charge loss) that we saw in part a. We need to calculate what is the minimum voltage to which we can allow node X to discharge to before we refresh. So we set $\Delta V = 100mV$ and solve for $V_{BIT}$.

$$V_{BIT1,min} = V_{PRE} + \frac{C_X + C_{BL}}{C_X}\Delta V \approx 0.77V$$

The time it takes to discharge to 0.77V is:

$$t_{d,leak} = C_X\frac{V_{BIT1} - V_{BIT1,min}}{I_{leak}} = 55.5fF\frac{0.8V - 0.77V}{1nA} = 1.67ms$$

Therefore we need to refresh the memory at least every $1.67ms$.

(d) One way to reduce power dissipation during retention is to reduce the retention frequency. Recommend a way to do that without changing the cell size. Discuss the trade-offs. *Hint:* A more realistic model for leakage is $I_{leak} \propto e^{\frac{V_{GS}-V_{TH}}{kT/q}}$.

In order to reduce the retention frequency without changing $C_X$ (which is proportional to the cell size), we need to reduce leakage. One way to do that is to drive the word line to a negative voltage instead of 0, which will cause it to have less leakage current. This means adding extra complexity to generate a negative voltage. Also, it requires a lot more careful design since negative voltages can damage the gate of the device and reduce reliability of the circuit.

Another way is to design sense amplifiers with smaller $\Delta V_{sense} = 100mV$. This goes beyond the scope of this class, but in general in order to do that we need to upsize the devices in the amplifiers in order to reduce mismatch. That in turn cause higher area and power dissipation in the sense amplifiers.