بسمه تعالی

پاسخ دوم درس معماری کامپیوتر

نیم‌سال اول ۰۱-۰۰

۱. حافظه‌ی‌اصلی به بزرگی ۱۲۸ کیلو کلمه و حافظه‌ی‌نهان به بزرگی ۸ بلوک ۸ کلمه‌ای موجود است. با فرض استفاده از روش نگاشت مستقیم و خالی بودن حافظه‌ی‌نهان در ابتدا، اگر آدرس‌های زیر به ترتیب از چپ به راست فراخوانی شوند، مشخص کنید کدام برخورد می‌کنند و نرخ موفقیت را به دست آورید.

در روش نگاشت مستقیم، حافظه‌نهان و حافظه اصلی به تعدادی بلوک با اندازه برابر B تقسیم می شوند. دلیل این کار تامین همجواری مکانی برای داده هایی که احتمالا در آینده مورد استفاده قرار میگیرند است.

به ازای هر آدرس ورودی، ابتدا شماره index آن آدرس را به‌دست می‌آوریم تا بدانیم آدرس مدنظر به کدام بلوک از حافظه‌نهان نگاشت خواهد شد.سپس tag آن بلوک باید با tag آدرس درخواستی مقایسه شود تا مطمئن شویم داده در حافظه‌نهان موجود است. اگر نبود، بلوک متناظر با آن داده مد نظر به حافظه‌نهان آورده می‌شود و جایگزین بلوک قبلی می‌شود.

به بیان دقیق تر، از آنجا که کل فضای آدرس‌دهی 2^{17} است و تعداد بلوک های حافظه‌نهان ۸ است که اندازه آن هم ۸ است، پس tag باید ۱۱ بیت پر ارزش آدرس‌ها باشد، و index باید ۳ بیت بعدی باشد و آفست هم ۳ بیت آخر باشد.

```
address :   10 tag :  00000000000  index :  000  offset :  101
address :   20 tag :  00000000000  index :  001  offset :  010
address :    3 tag :  00000000000  index :  000  offset :  001
address :   15 tag :  00000000000  index :  000  offset :  111
address :  1000 tag :  00000000111  index :  110  offset :  100
address :   60 tag :  00000000000  index :  011  offset :  110
address :   16 tag :  00000000000  index :  001  offset :  000
address :   17 tag :  00000000000  index :  001  offset :  000
address :   18 tag :  00000000000  index :  001  offset :  001
address :   19 tag :  00000000000  index :  001  offset :  001
address :   20 tag :  00000000000  index :  001  offset :  010
address :   10 tag :  00000000000  index :  000  offset :  101
address :   12 tag :  00000000000  index :  000  offset :  110
address :   20 tag :  00000000000  index :  001  offset :  010
address :   28 tag :  00000000000  index :  001  offset :  110
address :   15 tag :  00000000000  index :  000  offset :  111
address :  1001 tag :  00000000111  index :  110  offset :  100
address :   32 tag :  00000000000  index :  010  offset :  000
address :   53 tag :  00000000000  index :  011  offset :  010
address :   39 tag :  00000000000  index :  010  offset :  011
address :   44 tag :  00000000000  index :  010  offset :  110
address :   50 tag :  00000000000  index :  011  offset :  001
address :   72 tag :  00000000000  index :  100  offset :  100
```

آدرس های سبز اصابت می کنند و مشکی‌ها نمی کند. بنابرین نرخ اصابت برابر خواهد بود با :

Hit rate = 13/23 = 0.565

۲. دو تابع first و second تعریف شده‌اند که مجموع مقادیر موجود در دو آرایه بزرگ A و B را حساب کنند. کدام یک از پیاده‌سازی‌ها از منظر سخت‌افزاری بهتر است؟ این بهتر بودن چه نتیجه‌ای دارد؟ برای پاسخ‌های خود دلیل بیاورید.

پیاده سازی تابع first بهتر است. زبان‌های برنامه‌نویسی از دو روش برای ذخیره سازی آرایه‌های دوبعدی استفاده می‌کنند که میتوانید در اینجا بیشتر در مورد آن‌ها مطالعه کنید.

به هر حال، آرایه‌های این سوال یک‌بعدی هستند و اعضای یک آرایه با هم همجواری مکانی دارند. بنابرین بهتر است که عملیات جمع به صورت جداگانه روی اعضای آرایه ها صورت بگیرد تا نرخ اصابت بالا بماند و سرعت اجرای کد بیشتر شود.

۳. در مورد حافظه از نوع SRAM و حافظه از نوع DRAM به سوالات زیر پاسخ دهید:

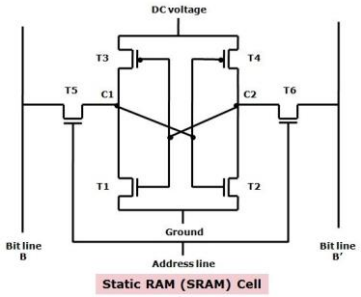
الف) چرا DRAM هر از چندگاهی به حضور جریان الکتریکی نیاز دارد ولی SRAM ندارد؟

ب) چرا هزینه ساخت یک واحد حافظه SRAM از یک واحد حافظه DRAM بیشتر است؟

تدریجی بارشان خالی می‌شود و نیاز به جریان دوباره برق است تا این بار حفظ شود. حافظه DRAM هرچند میلی‌ثانیه نیاز به این جریان دارد تا اطلاعاتش حفظ شود. در مقابل، حافظه SRAM از ترانزیستوها و latch ها بهره می‌گیرد و مشکل نشتی بار ندارد.

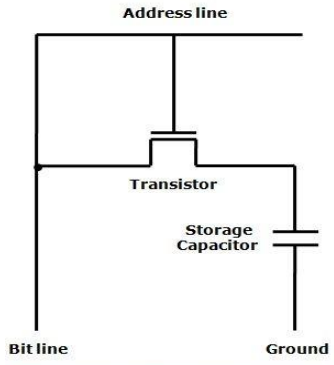
الف) در ساخت حافظه DRAM از خازن‌ها استفاده می شود. خازن‌ها به دلیل حضور دی‌الکتریک، در غیاب جریان، به صورت

ب) حافظه SRAM از ترانزیستور استفاده میکند. هر واحد حافظه دارای ٦ ترانزیستور مطابق شکل زیر است:



Static RAM (SRAM) Cell

در مقابل در ساخت حافظه SRAM از خازن و یک ترانزیستور استفاده می شود:



Dynamic RAM (DRAM) Cell

به دلیل پیچیده بودن سخت افزار SRAM و استفاده از تعداد بیشتری ترانزیستور، قیمت آن بالاتر از DRAM است.

۴. حافظه‌های ROM[1] همانطور که از نامشان پیداست، حافظه‌هایی هستند که تنها می‌توانیم به آن‌ها دسترسی خواندن داشته باشیم. این حافظه‌ها تنها یک بار به صورت سخت‌افزاری برنامه‌ریزی می‌شوند.

در شکل زیر نمونه‌ای از این نوع حافظه را مشاهده می‌کنید. همانطور که می‌بینید برخی از خانه‌ها این حافظه برنامه‌ریزی شده‌اند. با توجه به ساختار این حافظه توضیح دهید، مقدار خانه‌ی حافظه‌ی دوم چند است و این مقدار چطور خوانده می‌شود.

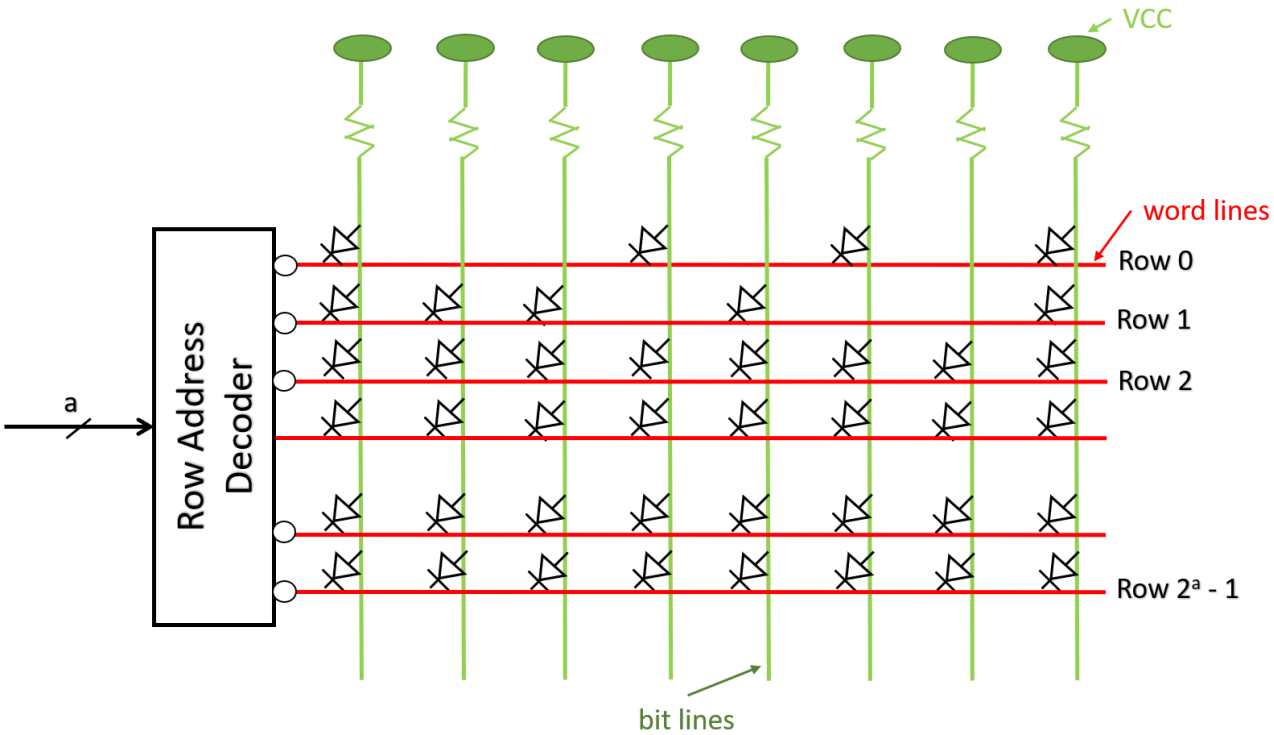
برای انکه مقدار این خانه خوانده شود باید به decoder آدرس ۱ داده شود. در این صورت با توجه به low active بودن decoder، تمامی word lineها مقدار ۱ خواهند داشت و Row1 مقدار ۰ خواهد داشت. همچنین از آنجا که خطوط bit line از بالا به یک منبع ولتاژ متصل هستند، بنابراین دارای مقدار ۱ هستند.

می‌دانیم که دیود یک مدار یک‌سوکننده است و تنها از یک سمت جریان را از خود عبور می‌دهد. تمامی دیودها در مدار از سمت خطوط bit line به word lineها متصل شده‌اند. بنابراین هر زمان که مقدار یک bit line یک باشد و word line صفر باشد، آن دیود جریان را عبور می‌دهد. در غیر اینصورت جریانی از آن دیود نمی‌گذرد. (اگر هر دو ۱ باشند که جریانی بین دو ولتاژ برابر نخواهیم داشت. حالت‌های دیگر نیز با توجه به ۱ بودن خطوط bit line به وجود نمی‌آید.)

پس تنها دیودهایی که می‌توانند جریان را از خود عبور دهند، دیودهای متصل به Row1 هستند. اما برخی از این دیودها سوزانده شده‌اند در نتیجه از آن گره جریانی نمی‌گذرد. در جاهایی که دیودی نیست، مقدار انتهای خطوط bit line به دلیل اتصال به VCC برابر ۱ خواهد بود. در جاهایی که دیودها سوزانده شده‌اند، این مقدار ۰ می‌شود.

با توجه به توضیحات بالا به مقدار خانه‌ی دوم یا Row1 برابر خواهد بود با: ۰۰۰۱۰۱۱۰

برای نوشتن مقدار ۰۱۱۱۰۱۰۰ در خانه‌ی سوم باید دیودهای دوم، سوم، چهارم و ششم (به ترتیب از سمت چپ) را بسوزانیم تا مقدار موردنظر از Row2 خوانده شود.



[1] Read Only Memory