BÀI THỰC HÀNH TUẦN 13
KIẾN TRÚC MÁY TÍNH

Họ và tên: Đinh Huy Dương

MSSV: 20215020

TRUY CẬP MA TRẬN THEO CHIỀU NGANG

- Hit rate của Cache cuối cùng sau khi chạy chương trình là 75%. Điều này được giải thích bởi sau mỗi lần bị Miss trong 1 block chứa 4 word, thì 3 word còn lại sẽ chắc chắn Hit (vì nó đã được ở trong block đấy khi load từng 4 word một cách tuần tự), quá trình này được lặp lại và tổng lại trong các lần được truy cập bộ nhớ chính, sẽ có ¾ lần Cache sẽ Hit.
- Tương tự như thế, khi tăng độ rộng của 1 block trong Cache lên 8 word, thì 1 lần Miss sẽ được theo bởi 7 lần Hit, khiến cho tỷ lệ Hit của Cache trở thành 7/8 = 87.5%. Với 1 block chứa 2 word, tỷ lệ Hit sẽ trở thành 50%

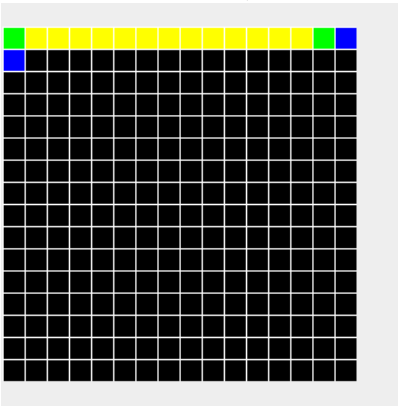
TRUY CẬP MA TRẬN THEO CHIỀU DỌC

- Hit rate của Cache luôn luôn là 0%. Ở đây chương trình duyệt mảng theo chiều dọc, nghĩa là ô nhớ tiếp theo cần truy cập sẽ cách ô nhớ trước 16 word, chứ không phải tuần tự như ví dụ trước. Vậy nên khi Cache load 4 word 1 cách tuần tự vào trong 1 block, sẽ không có một cái nào Hit cả.
- Ta chỉnh độ rộng 1 block sang 16 word với 8 block. Đồng thời để một Tool mới với 16 block, 1 block chứa 16 word. Chạy chương trình song song ta nhận thấy đặt 16 word cho mỗi block ở 8 block sẽ không tăng được Hit rate lên do tối đa 1 block Cache sẽ chỉ chứa được 1 hàng của ma trận mà chưa thể chuyển sang hàng mới, không thể Hit được với truy cập theo từng hàng.

- Trong khi đó, nếu tăng số lượng block từ 8 block lên 16 block , ta có thể chứa được tất cả các phần tử của ma trận khiến cho Hit rate của sẽ gần đến 100% (Cụ thể 94%). Lý do nó không được 100% là do các phần tử đầu được truy cập chưa được cho vào block nên sẽ bị Miss

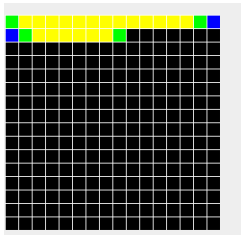
VISUALIZE BỘ NHỚ ĐƯỢC TRUY CẬP :

- Trong bài duyệt theo chiều ngang, bộ nhớ được truy cập tuần tự từ trái sang phải. Ngược lại trong bài duyệt chiều dọc, bộ nhớ được truy cập từ trên xuống dưới theo từng cột.
- Trong bài Fibonacci, chương trình thực hiện thủ tục đệ quy để có thể khởi tạo dãy Fibonacci vào trong mảng. Khi đó sẽ có $t0 để làm địa chỉ ban đầu của mảng. Phần tử đầu và thứ 2 (=1) sẽ được cho vào mảng dùng (sw). 2 ô nhớ ở đây được truy cập 1 lần. Sau đó lw 2 ô nhớ này để tính toán được phần tử tiếp theo cho mảng (1+1=2) và sw để lưu vào phần tử thứ 3 vào trong mảng. Tăng địa chỉ của $t0 thêm 4 để có thể tính được phần tử tiếp theo. Lặp lại quá trình này. Cuối cùng ta thu được mảng, và số lần truy cập sẽ được tăng lên 3 ở các ô nhớ trừ ô nhớ đầu và gần cuối là 2, và ô nhớ cuối là 1 (vì chỉ dùng 1 lệnh sw)

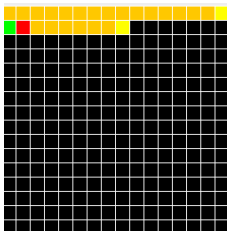


Các phần tử của mảng được lưu tại các ô nhớ ở hàng đầu. Ô xanh thứ 2 ở hàng 2 là array size được lưu ở đầu chương trình

- Đến với thủ tục Print để in ra màn hình, sau lệnh syscall, ta có thể thấy phần bộ nhớ ở hàng 2 được truy cập để có thể in ra xâu được lưu trong Data



Và sau đó ta được thấy chương trình truy cập lại bộ nhớ của mảng để có thể in các phần tử, khiến cho màu của các ô nhớ đấy chuyển màu.



- Khi khởi động Giả lập Cache, ta quan sát được Hit rate của Cahe là 94% (do các phần tử đầu bị MISS) và không sử dụng hết các block: