Ford as a Software Company

http://www.informationweek.com/news/global-cio/interviews/231902920

2 Cycle development

- Stalling an instruction = hold it back in the pipeline, including all instructions behind it
- When stalling, need to inject a bubble in place of the stalled instruction

- bomb code → next lab period

- memory mountain: see pp. 621-624

  "It is not entirely clear why these dips occur. The only way to be sure is to perform a detailed cache simulation…."

Summary of Matrix Multiplication

ijk (& jik):
- 2 loads, 0 stores
- misses/iter = **1.25**

```
for (i=0; i<n; i++)  {
   for (j=0; j<n; j++) {
      sum = 0.0;
      for (k=0; k<n; k++)
         sum += a[i][k] * b[k][j];
      c[i][j] = sum;
   }
}
```

kij (& ikj):
- 2 loads, 1 store
- misses/iter = **0.5**

```
for (k=0; k<n; k++) {
   for (i=0; i<n; i++) {
      r = a[i][k];
      for (j=0; j<n; j++)
         c[i][j] += r * b[k][j];
   }
}
```

jki (& kji):
- 2 loads, 1 store
- misses/iter = **2.0**

```
for (j=0; j<n; j++) {
   for (k=0; k<n; k++) {
      r = b[k][j];
      for (i=0; i<n; i++)
         c[i][j] += a[i][k] * r;
   }
}
```

Practice: Which is better and why?

```
#define N 1000

typedef struct {
    int vel[3];
    int acc[3];
} point;

point p[N];
```

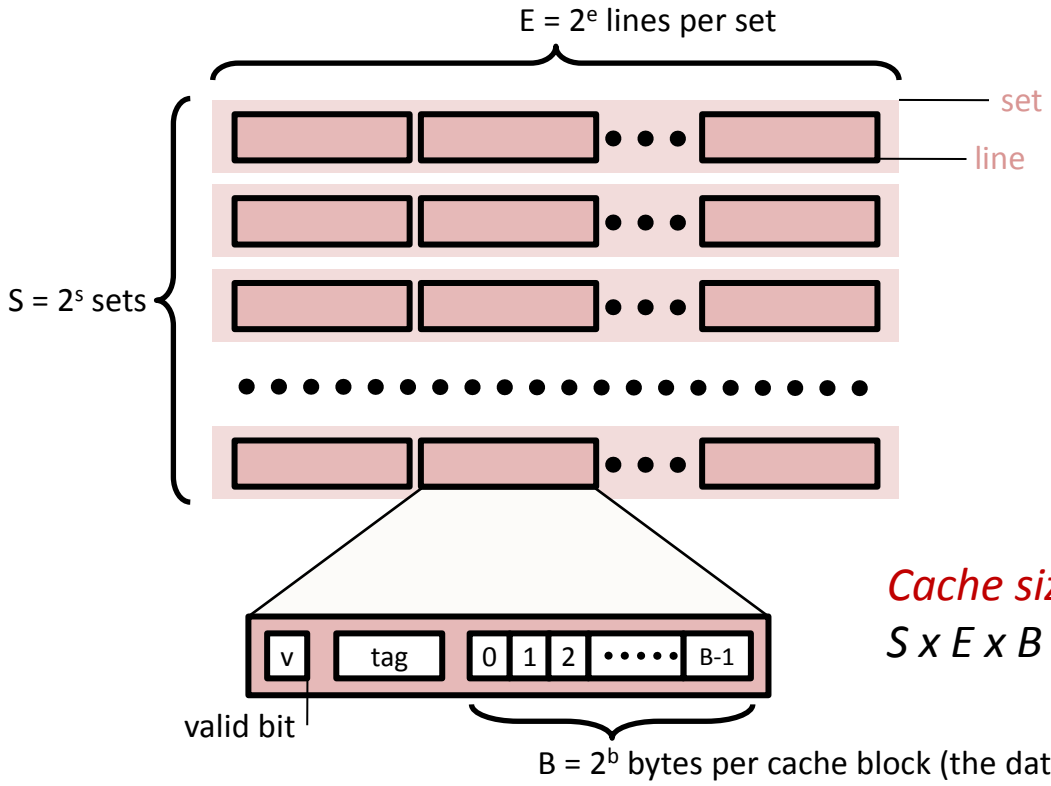
```
void clear1 (point *p, int n){
    int i, j;

    for (i=0; i<n; i++){
        for (j=0; j<3; j++)
            p[i].vel[j] = 0;
        for (j=0; j<3; j++)
            p[i].acc[j]=0;
    }
}
```

```
void clear3 (point *p, int n){
    int i, j;

    for (j=0; j<3; j++){
        for (i=0; i<n; i++)
            p[i].vel[j] = 0;
        for (i=0; i<n; i++)
            p[i].acc[j]=0;
    }
}
```

General Cache Organization (S, E, B)



Cache size:

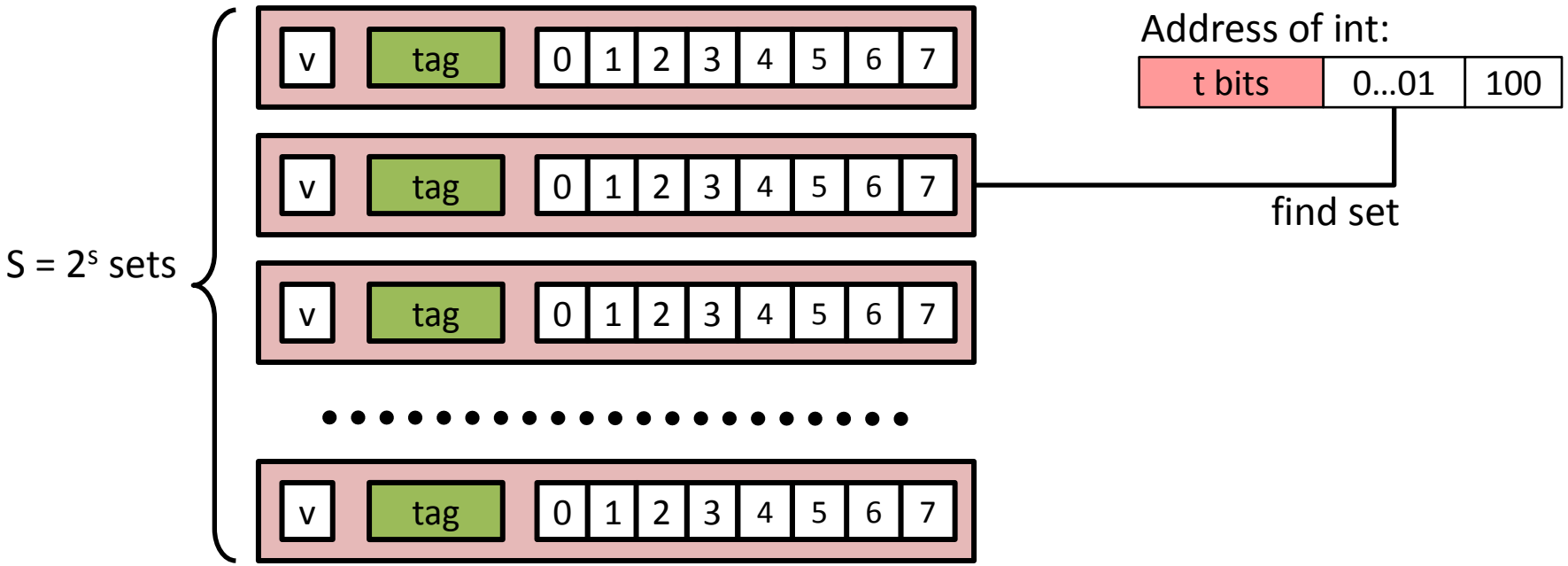
S x E x B data bytes

Cache

- Direct-mapped cache (E=1):
  - Primary cache memory has **S** blocks of memory of **B** size.
  - Secondary memory has **S'** blocks of memory of **B** size.
  - **S'** is a multiple of **S**.
  - Each block will map into a specific cache location based on the block number modulus **S.**
  - If a block exists in a specific location, then it will be evicted.

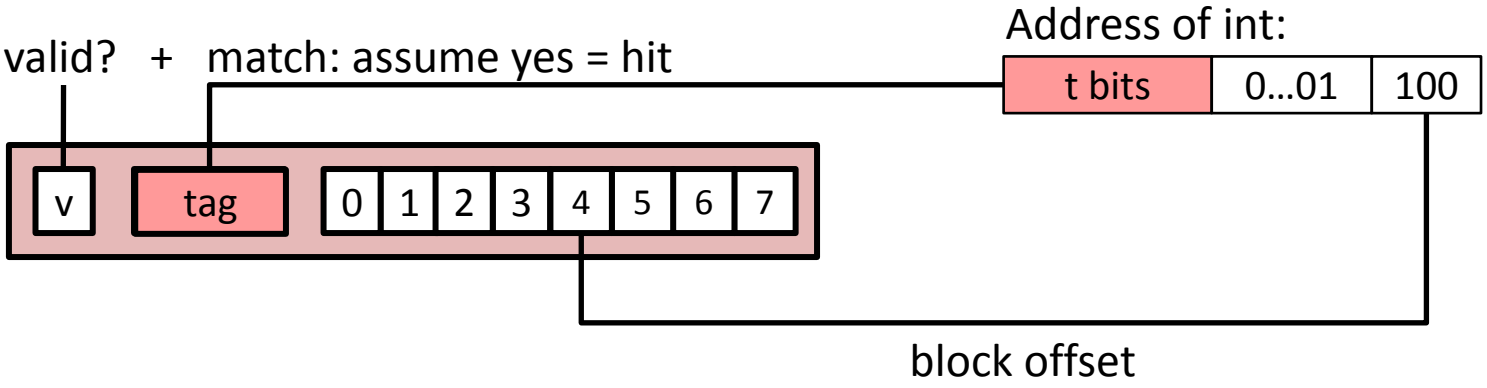
Example: Direct Mapped Cache (E = 1)

Direct mapped: One line per set
Assume: cache block size 8 bytes



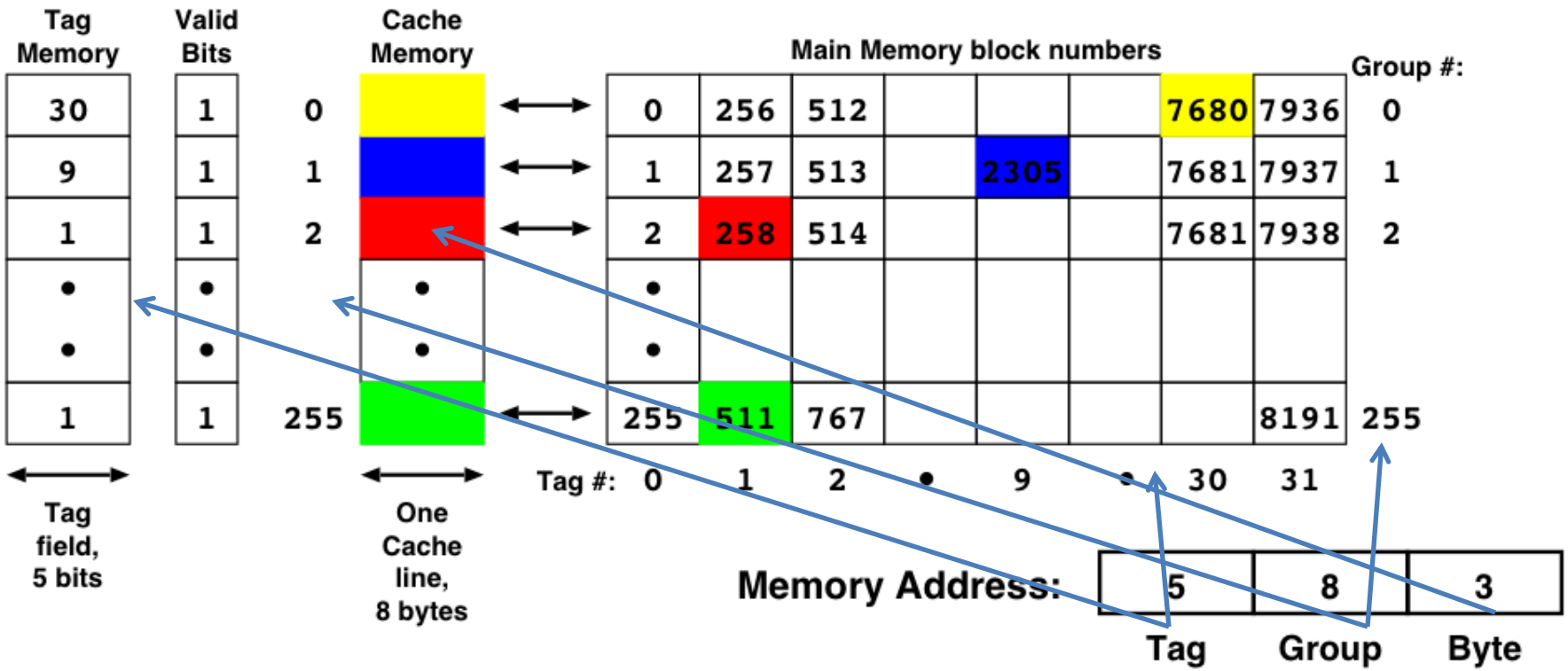
Example: Direct Mapped Cache (E = 1)

Direct mapped: One line per set
Assume: cache block size 8 bytes



No match: old line is evicted and replaced

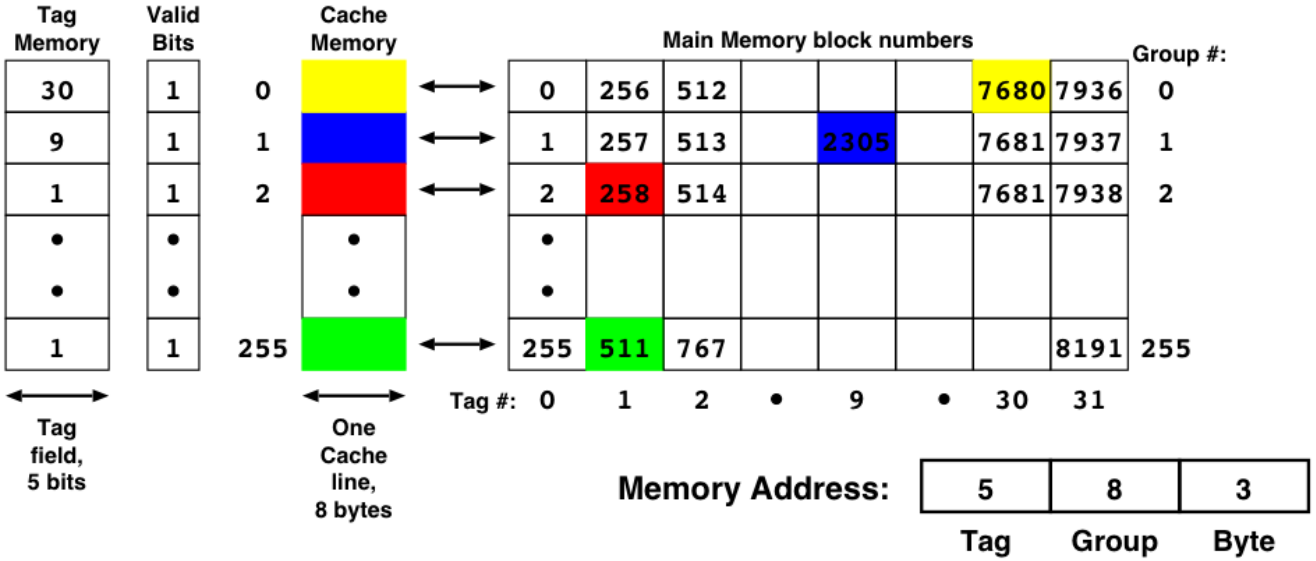
2nd (Colorful) Example



Size of this Cache?

Size of this Main Memory?

2nd (Colorful) Example



Main Memory contains:

$$2^5 \times 2^8 \times 2^3 = 2^{16} Bytes = 64\text{KB}$$

Cache Memory contains:

$$2^8 \times 2^3 = 2^{11} Bytes = 2\text{KB}$$

3rd Example

4 sets, one line per block,
2 bytes per block, 4-bit addresses

Address (decimal)	*Tag bit (t=1)*	*Index bits (s=2)*	*Offset bits (b=1)*	*Block number (decimal)*
0	*0*	*00*	*0*	*0*
1	*0*	*00*	*1*	*0*
2	*0*	*01*	*0*	*1*
3	*0*	*01*	*1*	*1*
4	*0*	*10*	*0*	*2*
5	*0*	*10*	*1*	*2*
6	*0*	*11*	*0*	*3*
7	*0*	*11*	*1*	*3*
8	*1*	*00*	*0*	*4*
9	*1*	*00*	*1*	*4*
10	*1*	*01*	*0*	*5*
11	*1*	*01*	*1*	*5*
12	*1*	*10*	*0*	*6*
13	*1*	*10*	*1*	*6*
14	*1*	*11*	*0*	*7*
15	*1*	*11*	*1*	*7*

Read at address 0
Read 1
Read 13
Read 8
Read 0