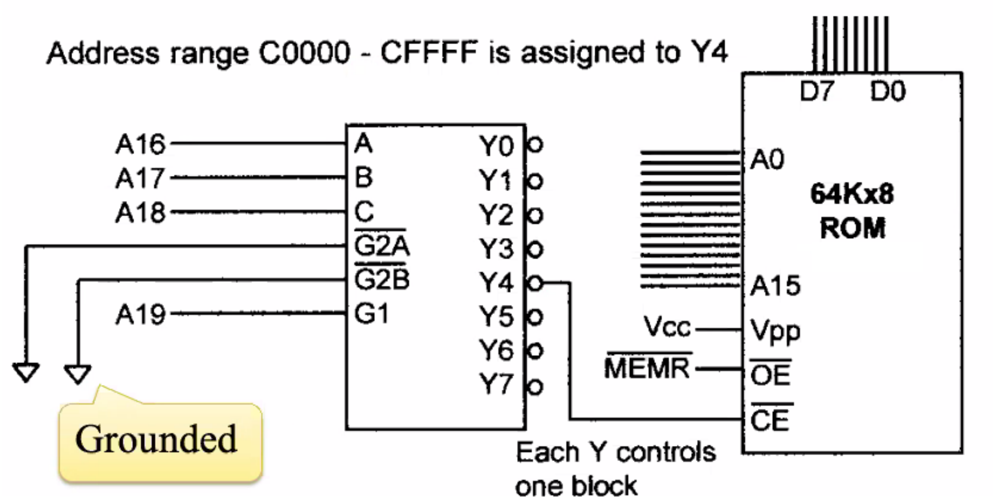
6 Memory and I/O in x86

6.1 Memory in x86 Family

Memory Address Decoding: CPU calculates the physical address of the operand and put corresponding signals on the address bus. We need a **memory address decoding circuitry** to locate the specific memory chip that stores the desired data.

- Two methods:
  - Use logic circuitry;
  - Use decoder chip;

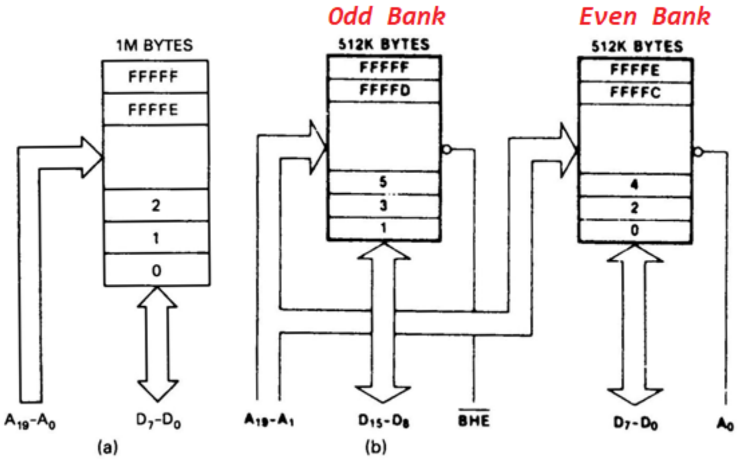
> [*Example*] 74LS138 Decoder Chip
>
> 
>

- Two classes:
  - Absolute address decoding（全译码）: All address lines are decoded.
  - Linear select decoding （部分译码）: Only selected lines are decoded, cheap, but with **aliases**: the same memory unit (I/O port) with multiple addresses.

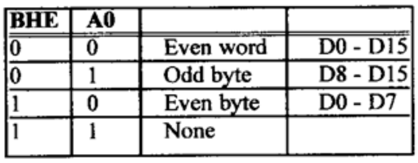
Data Integrity（数据完整性）

- **Checksum byte** for ROM: one checksum byte for N-bit data.
  - Calculation: Add all bytes together and drop all carries; And take the two's complement of the sum (which means negation);
  - Store the checksum byte together with data;
  - Check the integrity by adding data and the checksum together (the result is 0 or not).
- **Parity bit**（奇偶校验码） for DRAM: one parity bit for N-bit data
  - Even parity: Set the parity bit so that total number of 1s in $(N+1)$ bits is even.
  - Odd parity: Set the parity bit so thattotal number of 1s in $(N+1)$ bits is odd. (8086 use.)
- CRC for disks and the Internet.

Memory Organization in 8086: Even and odd banks.



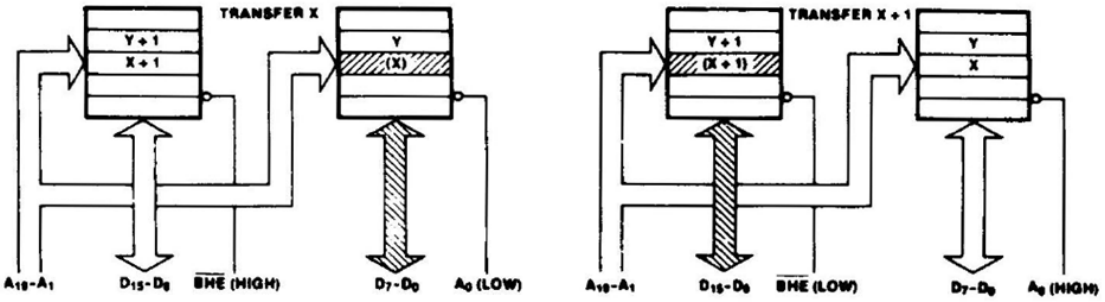
Address bits A_1 through A_{19} select the storage location that is to be accessed. They are applied to both banks in parallel. A_0 and bank high enable ($\overline{BHE}$) are used as **bank-select** signals.



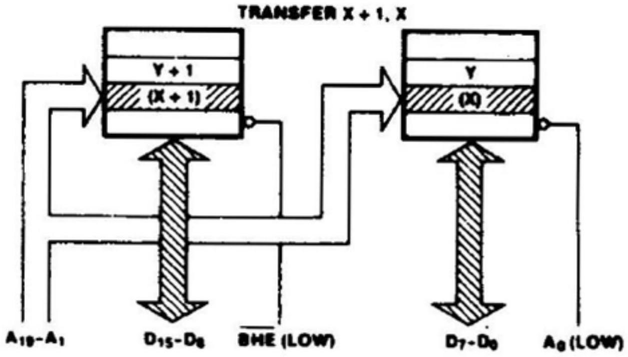
BHE	A0		
0	0	Even word	D0 - D15
0	1	Odd byte	D8 - D15
1	0	Even byte	D0 - D7
1	1	None	

Notice that in the memory chip, the enable pin is $\overline{CE}$, which means low voltage enables the memory chip. Therefore, we put $\overline{BHE}$ and A_0 into the two pins.

- **Single Byte Memory Operations**: you can refer to the previous table.

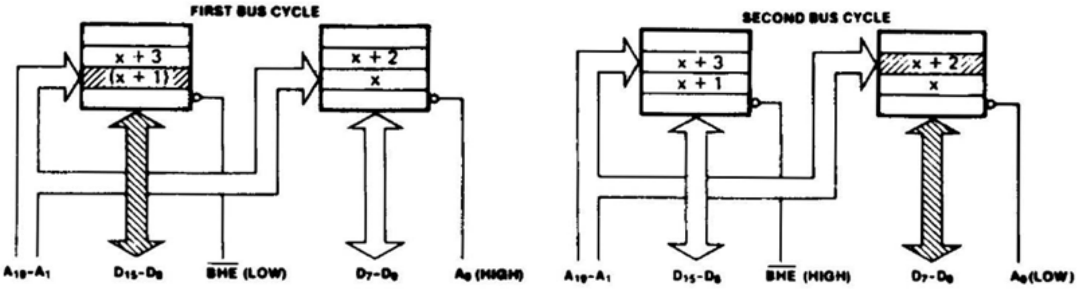


- **Aligned Word-Memory Operations**: accessing an aligned word at even address X.



Both the high and low banks are accessed at the same time. Both A_0 and $\overline{BHE}$ are set to 0. This 16-bit word is transferred over the complete data bus D_0 through D_{15} in just **one bus cycle**.

- **Misaligned Word-Memory Operations**: accessing a misaligned word at odd address $(X + 1)$.



Two bus cycles are needed. During the first bus cycle, the byte of the word located at address $(X+1)$ in the high bank is accessed over D_8 through D_{15}. Even though the data transfer uses data lines D_8 through D_{15}, to the processor it is the low byte of the addressed data word. In the second memory bus cycle, the even byte located at $(X+2)$ in the low bank is accessed over bus lines D_0 through D_7.

- Therefore, misaligned word-memory operations is more time-consuming than aligned word-memory operations.

6.2 I/O in x86 Family

Introduction of I/O in x86

- x86 microprocessors have an I/O space in addition to memory space, which is also called **peripheral I/O** or **isolated I/O**.
- Use special I/O instructions accessing I.O devices at *ports* (i.e., addresses for I/O).
- Memory can contain machine codes and data, I/O ports only contain data.

Instructions to access I/O

```
IN dest, source    ; input data
OUT dest, source   ; output data
```

`dest` and `source` are I/O addresses called I/O ports.

- Direct I/O instructions: The port number ranges from `00H` to `0FFH`, 256 ports in total.

  > [*Example*]
  >
  > ```
  > IN AL, 05H
  > OUT 25H, AL
  > ```

- Indirect I/O instructions: The port number ranges from `0000H` to `0FFFFH`, 65536 ports in all. Use a 16-bit address that resides in `DX` register.

  > [*Example*]
  >
  > ```
  > MOV DX, 2333H
  > IN AL, DX
  > OUT DX, AL
  > ```

- 16-bit I/O access is similar.