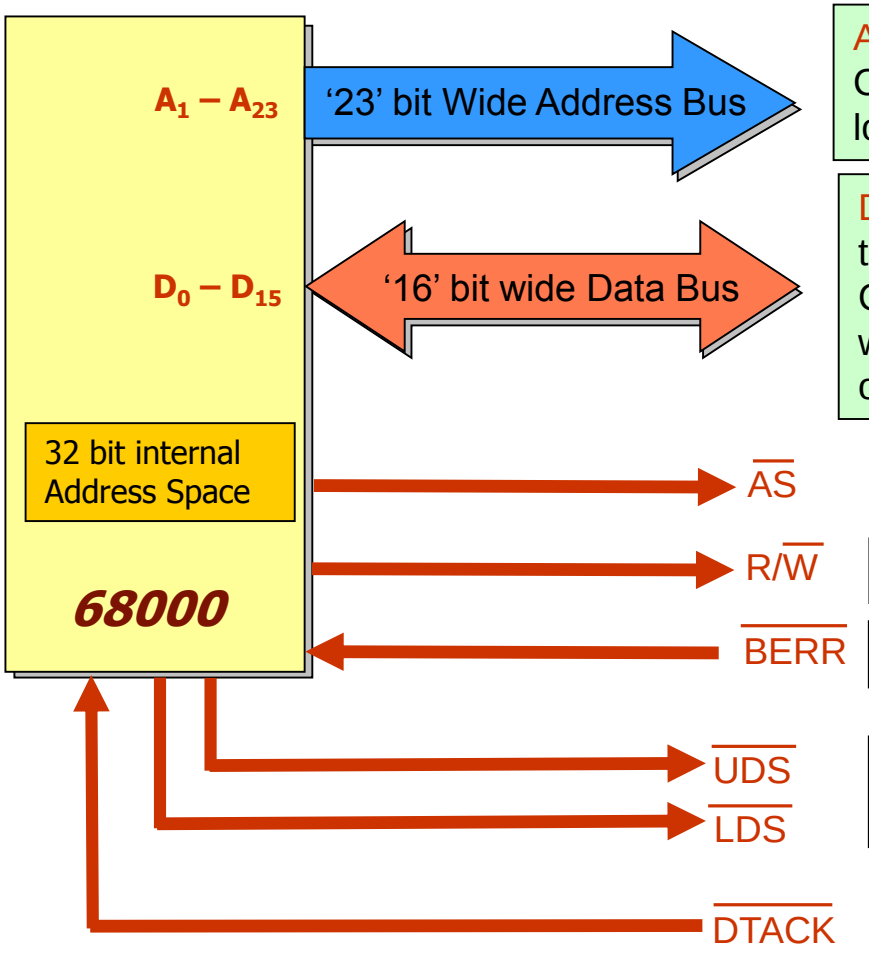
Motorola 68000: A Typical 16-bit data bus Interface:



http://experts.about.com/e/m/mo/Motorola_68000.htm

Address Bus – Address limit is 8M words. 2^{23}). Issued by CPU whenever it wishes to access any individual memory location in 8M word space. **Note there is no** A_0

Data Bus – Each memory location is 16 bits wide, thus total addressable storage is 16MBytes. Most 16/32 bit CISC CPUs have the ability to address either a long word, word or a byte at a time using some combination of signals such as LWORD, UDS and LDS etc. signals

Address Strobe: Logic 0 when address is Valid

Read/Write: Logic 0 for a write operation, 1 for a read

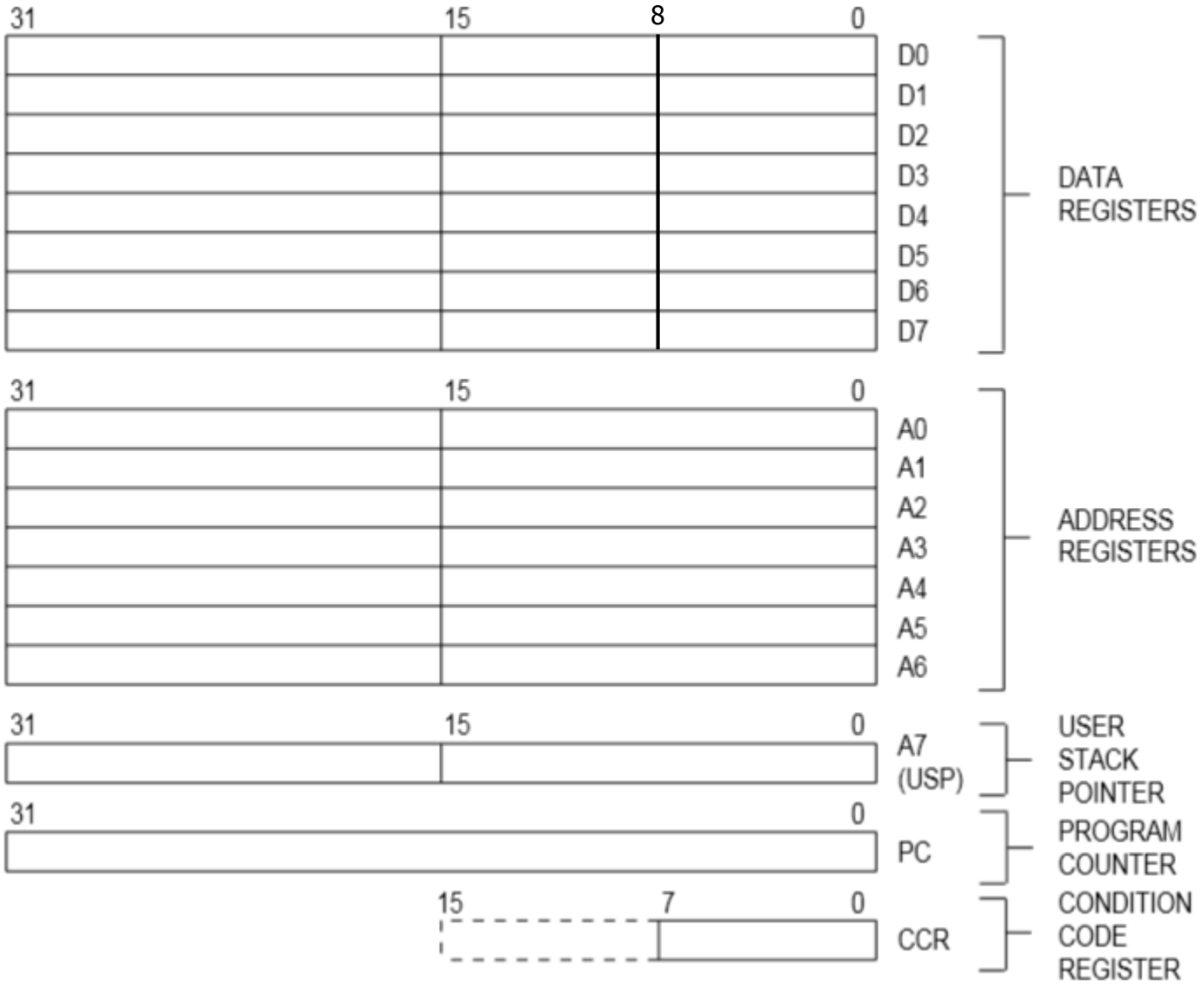
Bus Error: Timeout from non-responding memory

Upper and Lower Data Strobes that replace an A_0 Used to access individual bytes, within a 16 bit word

Data Transfer Acknowledge: A handshake back from the memory system saying data transferred.

Some very high speed 16/32 Bit Processors like the 68000, Pentium, ARM and NIOS II use Asynchronous data transfers. This means the CPU issues an address and the memory system must supply an acknowledge (like the DTACK signal here) back to the CPU to complete the access, otherwise the CPU will enter wait states. This means CPU can adjust its speed to that of the device it is accessing at the time. BERR can be issued when the CPU attempts to access memory that does not exist and therefore will not produce a DTACK

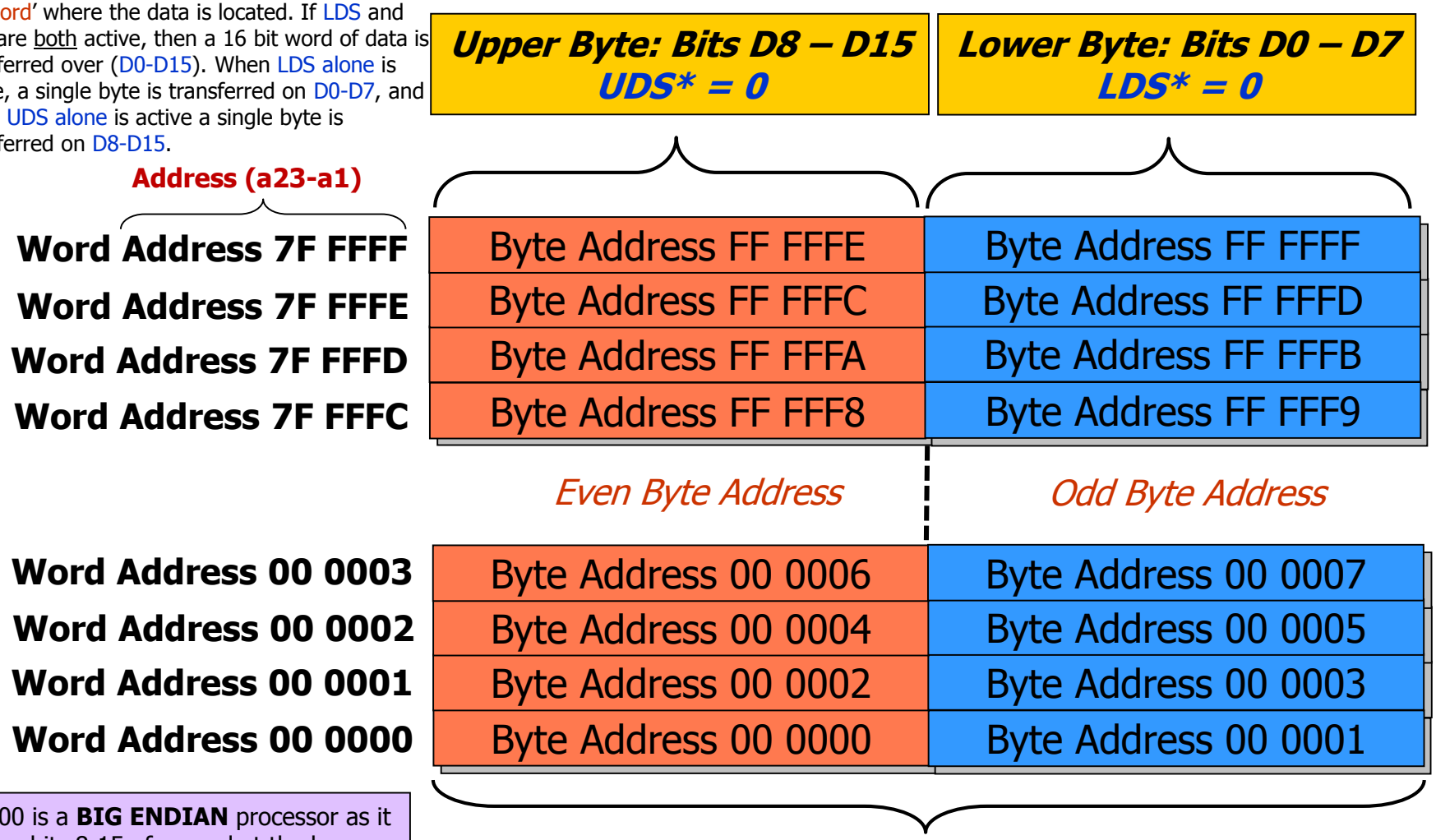
68000 Programmers Model (Internal Registers)



Motorola 68000: The LDS*/UDS* signals

Because there is no 'A0' (only A1 to A23), the 68000 address bus refers to the address of a 16 bit 'word' where the data is located. If LDS and UDS are both active, then a 16 bit word of data is transferred over (D0-D15). When LDS alone is active, a single byte is transferred on D0-D7, and when UDS alone is active a single byte is transferred on D8-D15.

Note: Every byte in memory has its own unique address. The 68000 issues a word address (i.e. the address of 2 bytes) and uses UDS* and LDS* to access either or both bytes.



68000 is a **BIG ENDIAN** processor as it stores bits 8-15 of a word at the lower byte address (i.e. the big end of a 16 bit value is read/written first) (see

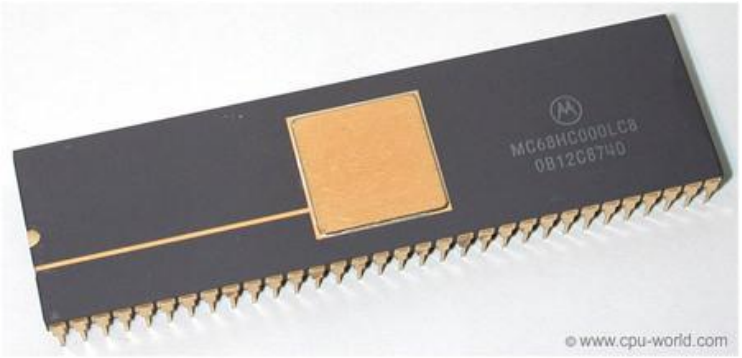
http://en.wikipedia.org/wiki/Endianness)

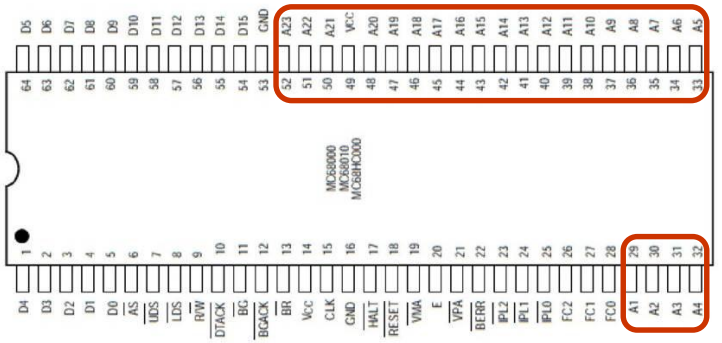
Intel Processors are **LITTLE ENDIAN**

Accessing a 16/32 bit word/long word is only permitted at even byte addresses by asserting LDS* and UDS* simultaneously – why?

Motorola 68000 Address Bus

- The original 68000 was limited to a 23 bit *external* address bus, even though *internally* its address bus was 32 bit. In fact there was not even an A0 signal.
- This restriction allowed the processor to fit inside a 64 pin DIP (dual in-line package) which was a common (if large) format at the time.





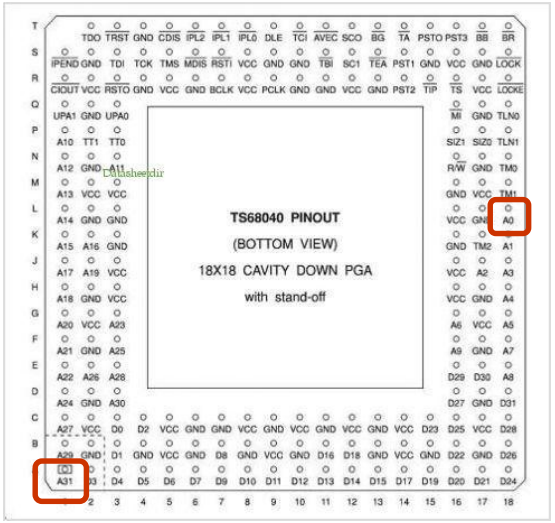
- The result of this is that the internal address bits A31-A24 of any address issued by the program got discarded during a memory access.
- Consequently an instruction such as this which writes to location FF000000

```
move.b   d0, $FF000000
```

results in a byte being copied from the lower 8 bits of register d0 to memory location/address $0000 0000

Motorola 68000 Address Bus

- There is no reason why A0 should not exist (*except to fit in the 64 pin package*), as it can be created from $\overline{\text{UDS}}$, i.e. A0 = $\overline{\text{UDS}}$ (*provided a byte transfer is under way, A0 is meaningless for a word transfer*).
- Subsequent 68000 family processors such as the 68020/30/40/60 had an A0 and also had address lines A24-31 as they used Ball Grid Array or PLCC packaging rather than DIP, thus they had more pins.



- The soft core 68k processor used in CPEN 412 does <u>not</u> need to constrain itself to a 64 pin package, consequently it has the full complement of address lines A31-A0 and can thus address 4Gbytes of memory (2^{32}). Although we only **<u>use</u>** A31-A1

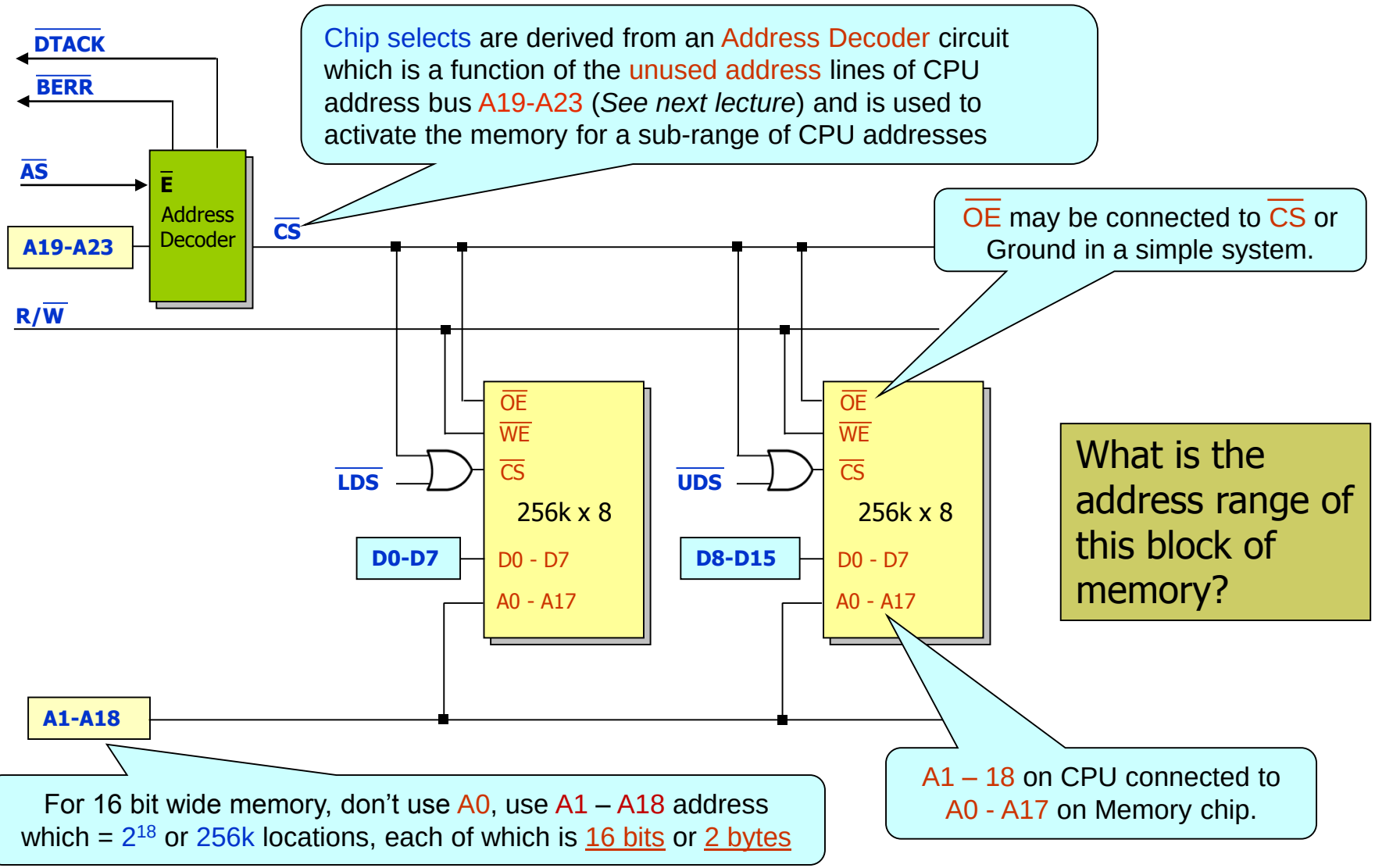
Motorola 68000 Data Transfers

- What data bus lines carry the data in the following assembly language instructions and what values appear on the Address and Data Buses, R/W*, UDS* and LDS* during the transfer? Describe the operation in detail.

```
move.b    d0,$0        write a byte from register d0 to address 0
move.b    d0,$1        write a byte from register d0 to address 1
move.b    d0,$2        write a byte from register d0 to address 2
move.b    d0,$FF       write a byte from register d0 to address hex FF
move.w    d0,$0        write a 16 bit word from register d0 to address 0
move.w    d0,$2        write a 16 bit word from register d0 to address 2
move.w    d0,$3        write a 16 bit word from register d0 to address 3
move.l    d0,$0        write all 32 bits of register d0 to address 0
move.l    d0,$1        write all 32 bits of register d0 to address 1
move.l    d0,$2        write all 32 bits of register d0 to address 2

move.b    $0,$2        transfer a byte from address 0 to address 2
move.w    $0,$2        transfer a 16 bit word from address 0 to address 2
move.w    $0,$1        transfer a 16 bit word from address 0 to address 2
move.l    $0,$4        transfer a 32 bit word from address 0 to address 4
```

A Typical 256k word (512K byte) Memory Block based on 256Kbyte devices

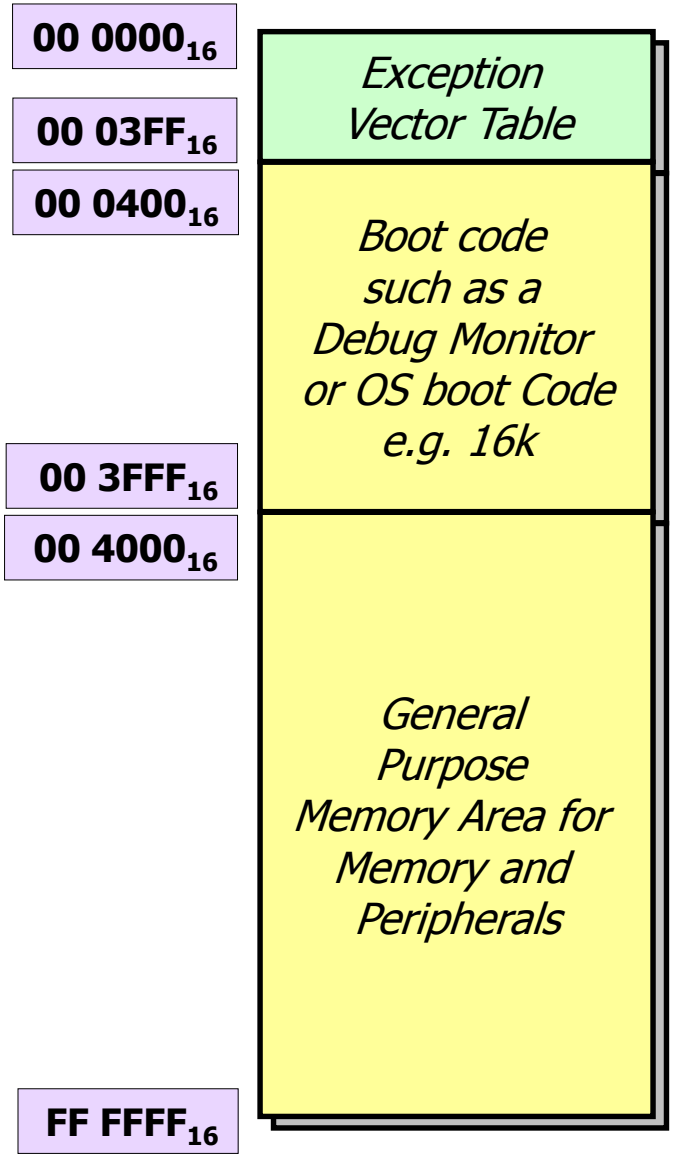


Important: 68000 memory must be 16 bits wide, accessible as independent 8 bit bytes

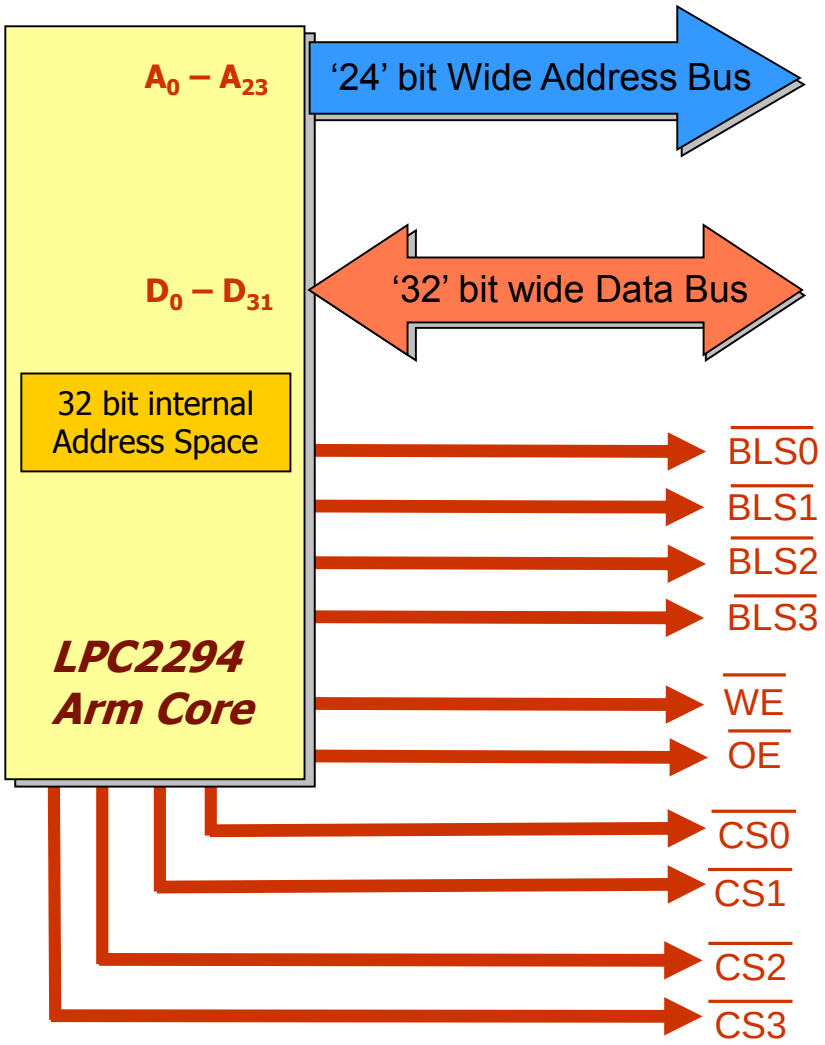
68000 Memory Organization

The 68000 Memory Space

- All CPUs have some reserved memory space set aside for special boot vectors and BIOS type code to bring the system to life.
- In the case of the original 68000, Memory locations 00 0000 to 00 03FF are reserved for the storage of a Vector Table containing interrupt, reset and other exception vectors. The designer may use the remaining space for whatever purpose they see fit.
- Usually the space immediately above the vector table is occupied by boot code. By placing vector table and boot code back to back they can be stored in the same ROM chips so that the table and boot code is always there to bring the system to life after a reset or power down.
- The rest of the memory space up to FF FFFF is for general purpose use and can contain memory and memory mapped peripherals.
- In small systems it is common for the general purpose memory area to be only partly populated with actual memory chips, thus there may be large gaps in the address space to which no memory chip will respond.



Typical 32-bit RISC CPU Interface: LPC2294 ARM Core from Phillips



Address Bus – External Address limit is 16Mbytes (i.e. 2^{24}). Note A_0 and A_1 may not get used depending upon chosen data bus width which is programmable within chip.

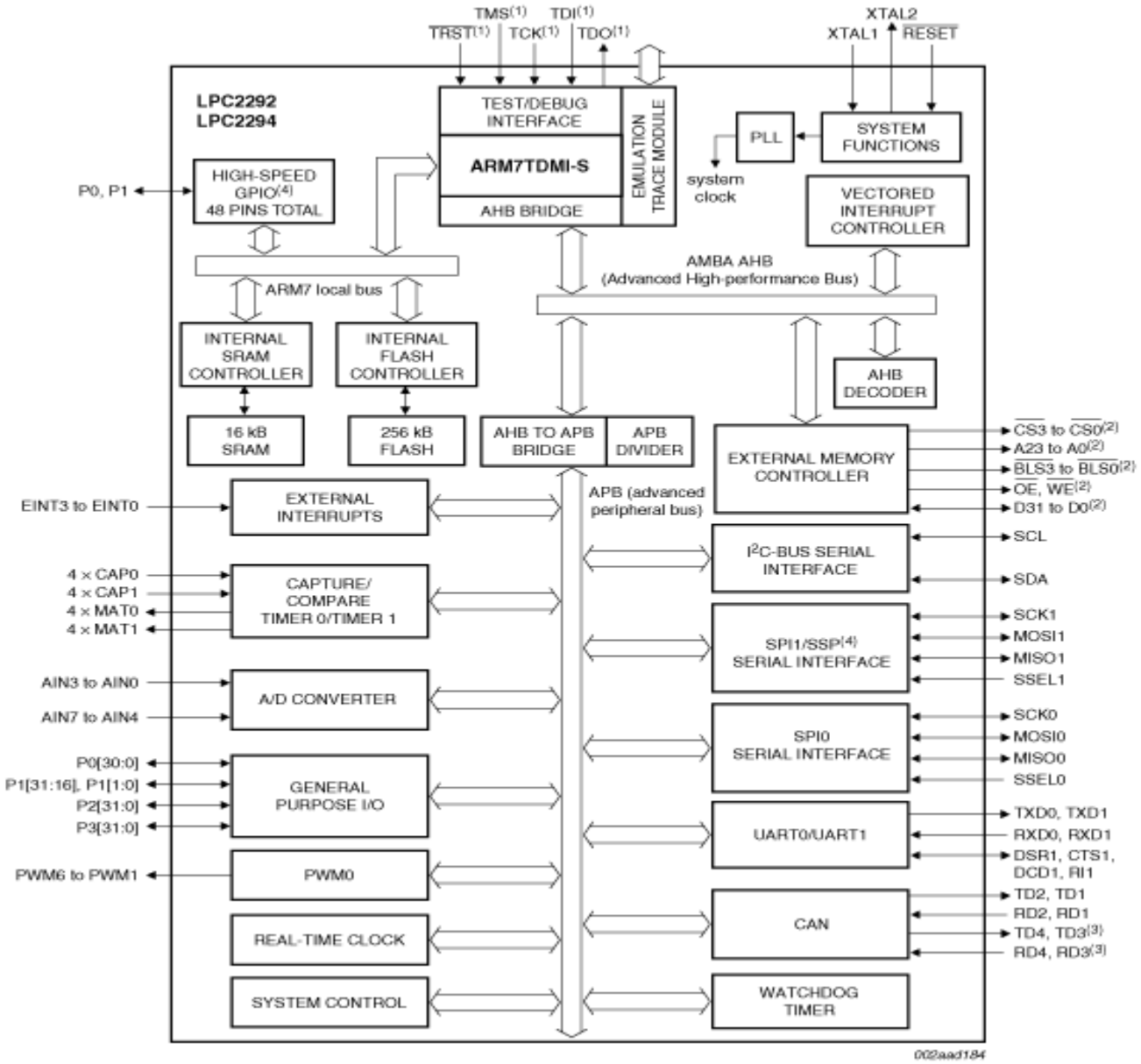
Data Bus – Each location is UP TO 32 bits wide, but may be configured to be 8, 16 or 32 bits, (Programmable within chip). Access to bytes is via BLS signals below

Byte Lane Select signals – Used to select individual Bytes during a memory access. 32 bit memory interfaces use all 4 signals, 16 bit wide memory interfaces use BLS0/BLS1 and 8 bit wide memory interfaces use BLS0 only. Similar concept to LDS*/UDS* on the 68000

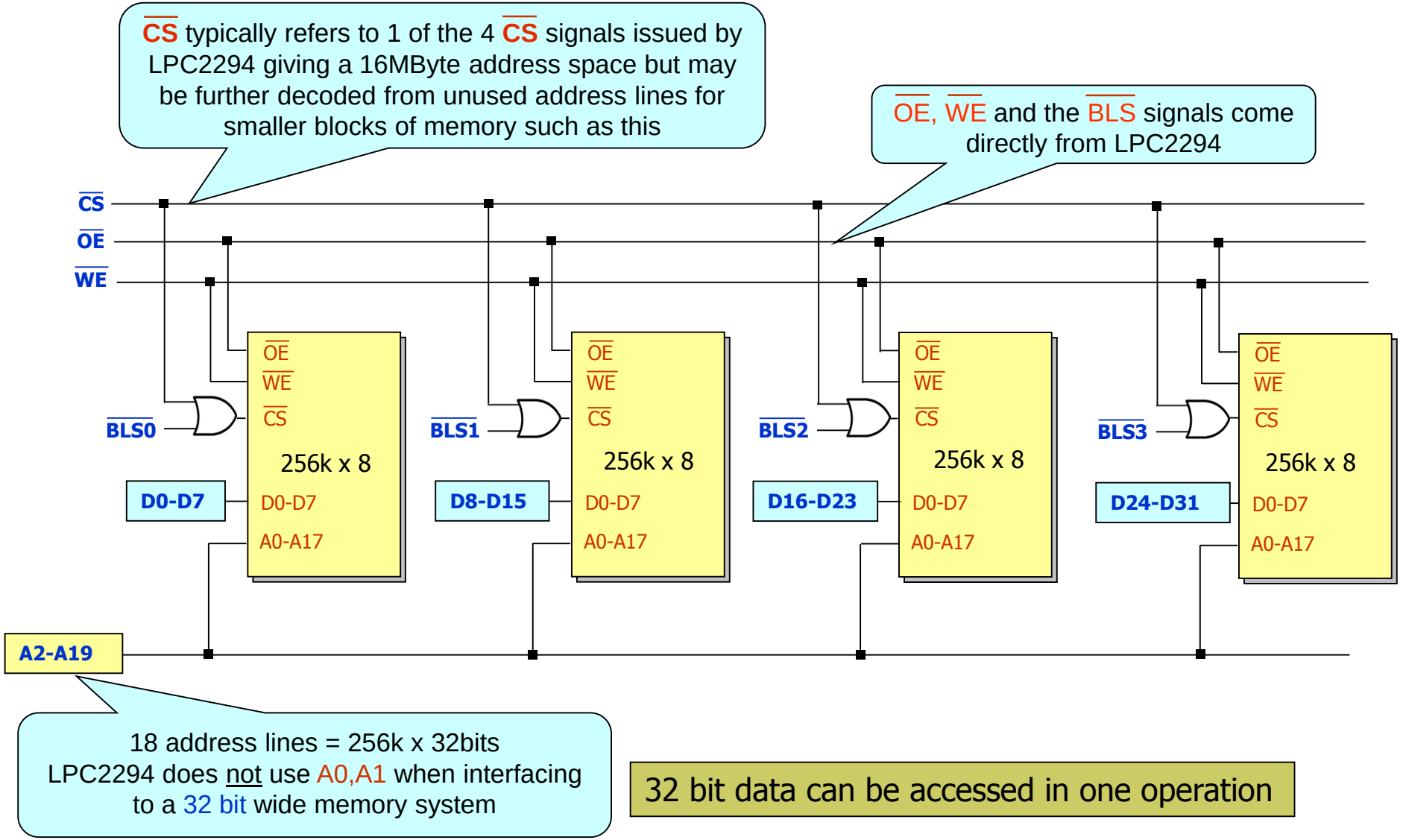
4 chip select outputs to allow the creation of 4 banks/blocks of up to 16Mbytes each. The signals are internally decoded from the 32 bit Address space of the CPU core and are asserted when a valid address is present. Addresses are programmable.

Philips LPC2294 is a specialist 32 bit **RISC** microcontroller based on ARM7 core with on chip memory, CANBUS, I^2C bus, A/D conversion, timers/counters. External Memory interface (shown above) is synchronous and EVERYTHING about it is programmable from **width**, **address Map** to **speed**. Four chip select signals give access to 4 banks of 16MBytes. Being a **RISC** based processor, all accesses involve 32 bit transfers, i.e. individual access to bytes or words is not permitted (unlike 68000)

LPC2294 ARM Core from Phillips

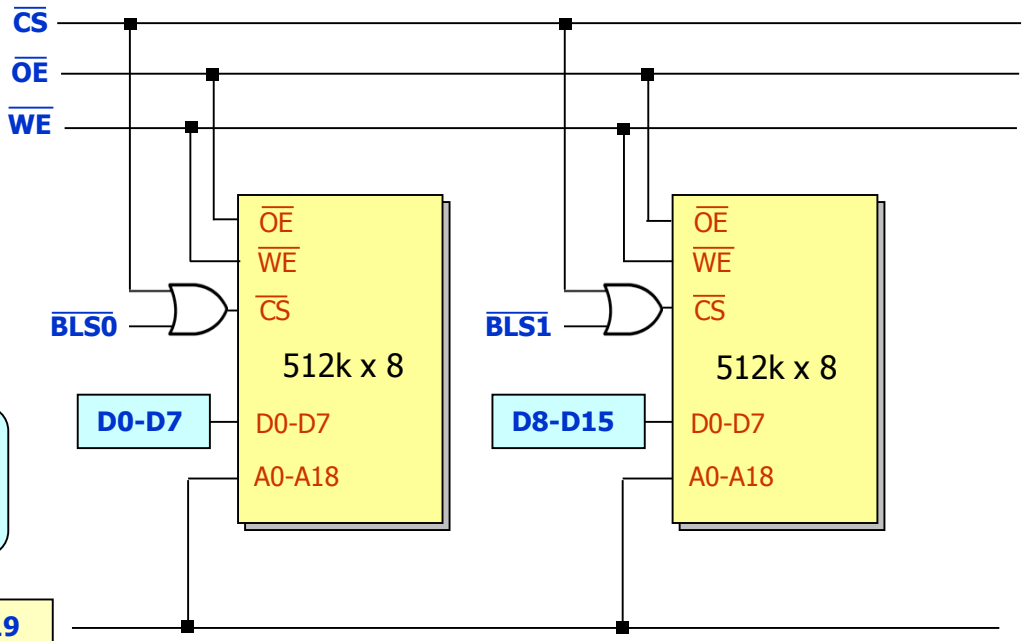


Typical 256k Long Word (1M Byte) Memory Bank for LPC2294



32 Bit Memory Interface using byte wide memory chips

Typical 512k Word (1M Byte) Memory Bank for LPC2294



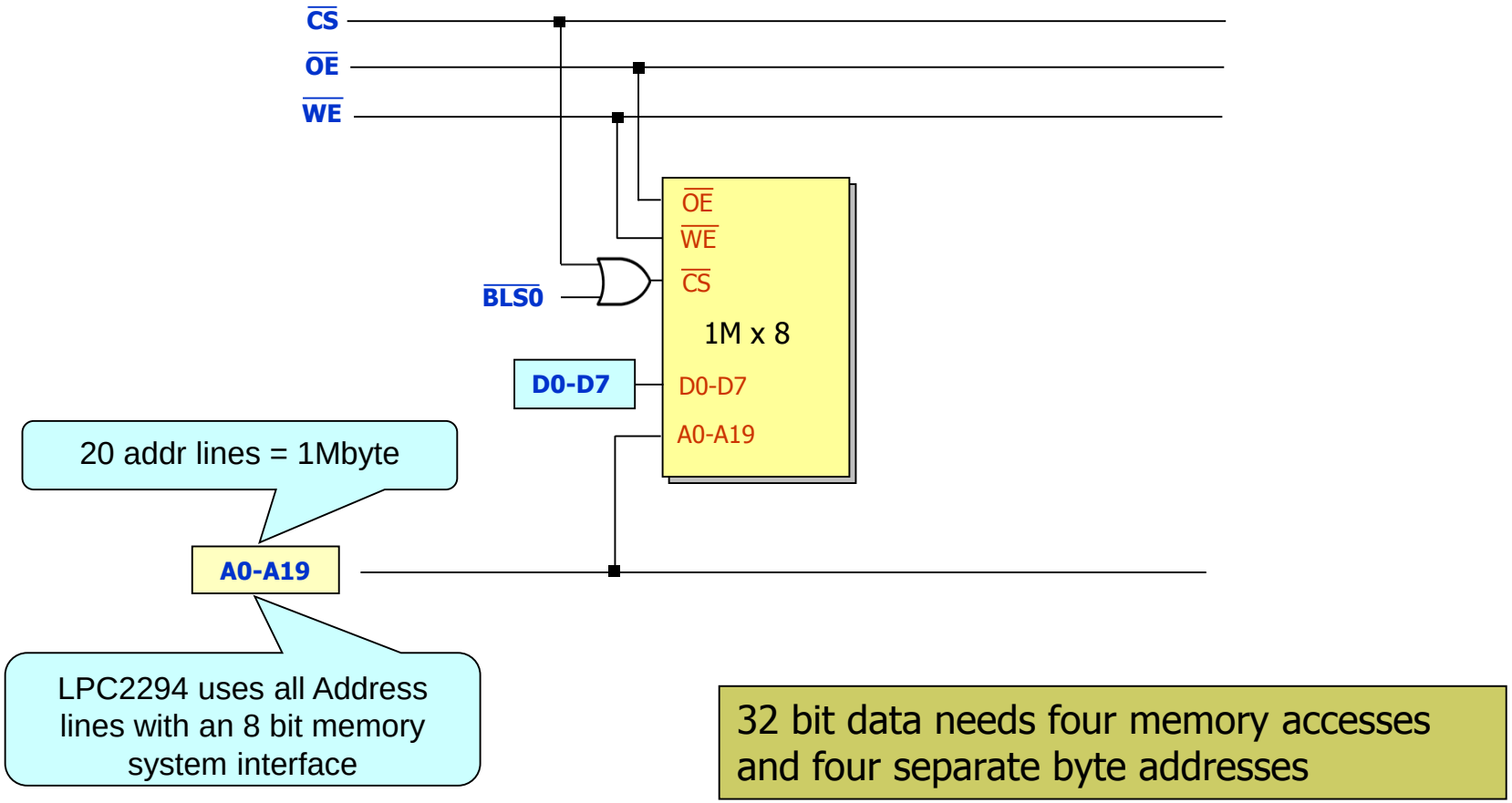
19 address lines = 512K locations of 16 bits

LPC2294 does not use A0 when interfacing to a 16 bit wide memory system.

32 bit data access needs two memory accesses and two separate word addresses

16 Bit Memory Interface using byte wide memory chips.

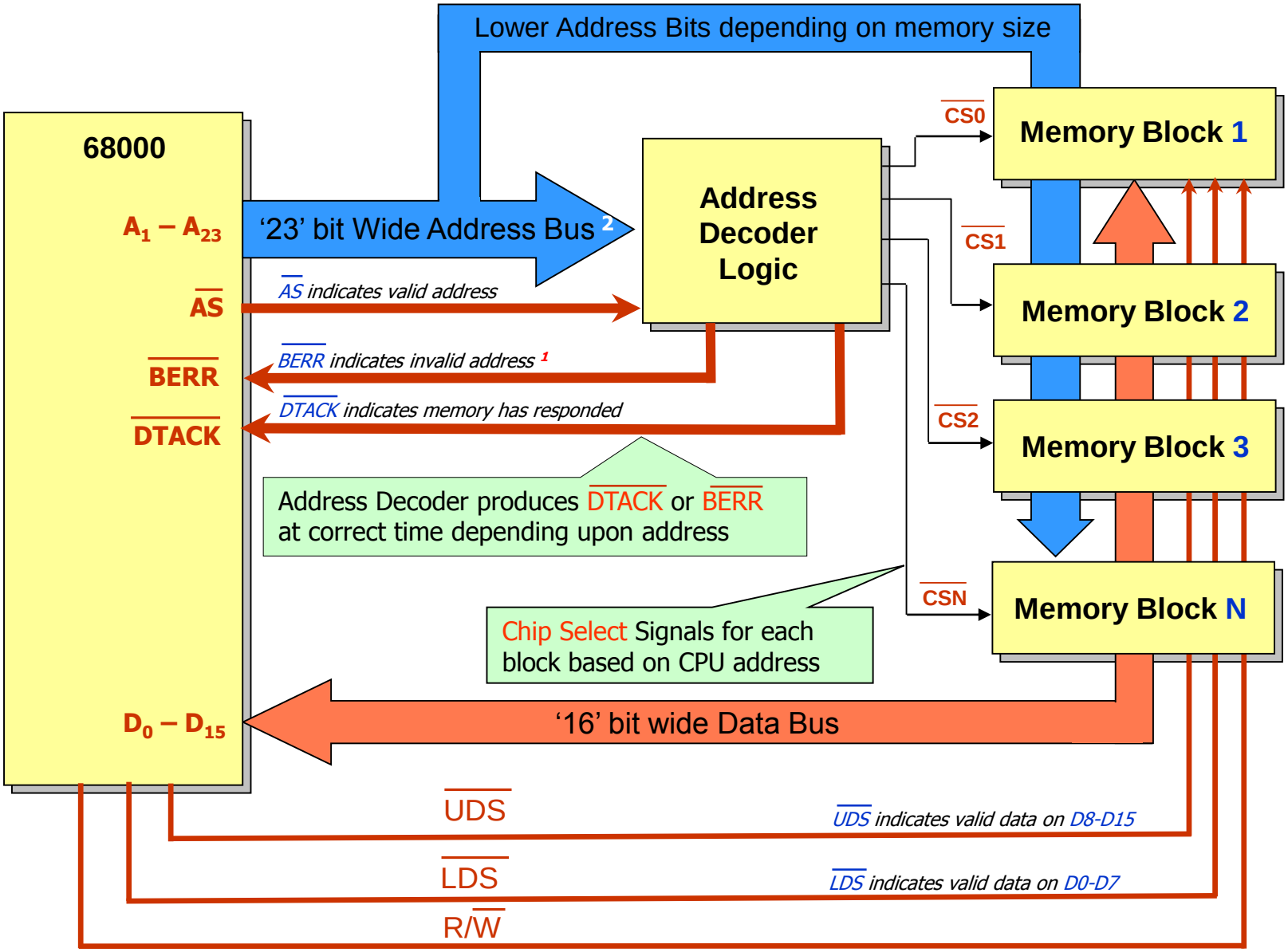
Typical 1Mbyte Memory Bank for LPC2294



32 bit data needs four memory accesses and four separate byte addresses

8 Bit Memory Interface using byte wide memory chips.

More Generic Memory Schemes: Example 68000



1 not present on soft core 68000 used in Cpen 412 2 soft core 68000 used in Cpen 412 has A31-A0