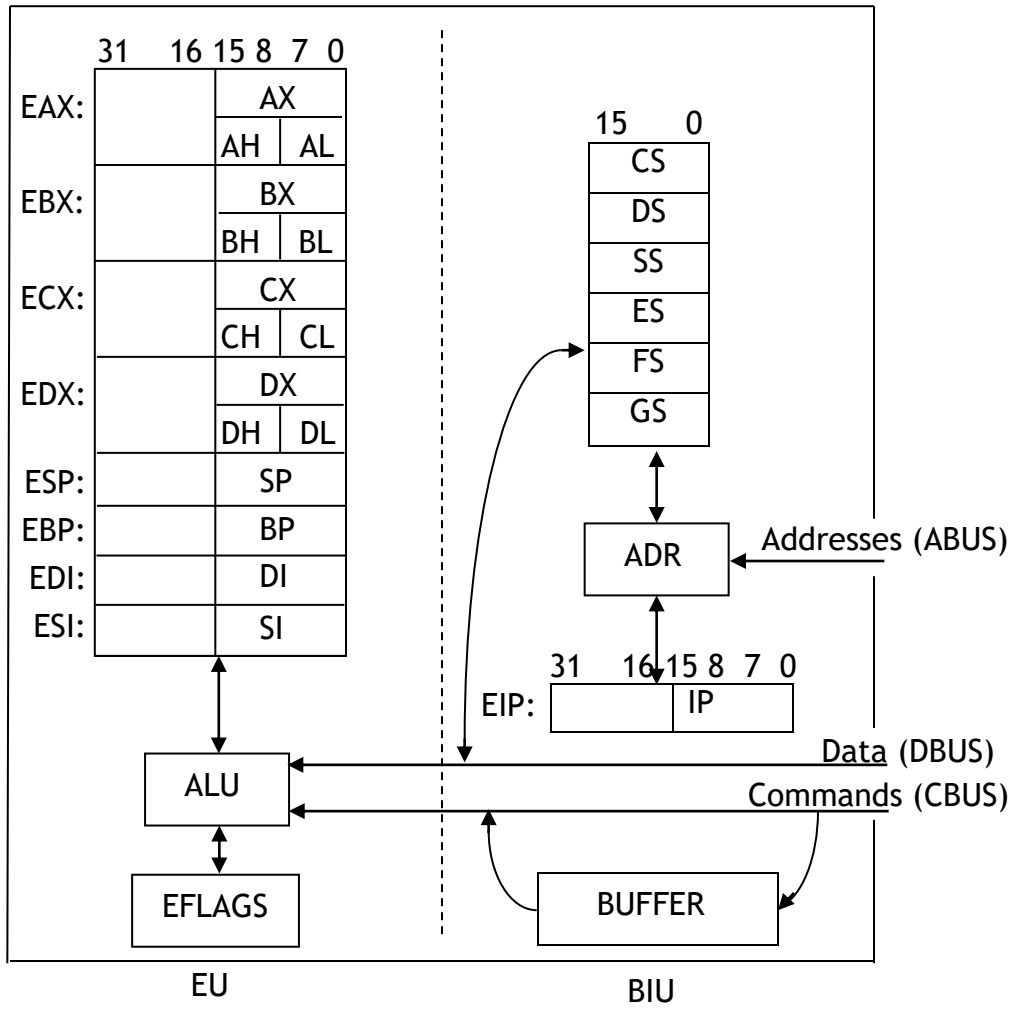
2.6. THE x86 MICROPROCESSOR ARCHITECTURE (IA-32)

2.6.1. x86 Microprocessor's structure

The x86 microprocessor has two main components:

- **EU** (*Executive Unit*) – run the machine instr. by means of **ALU** (*Arithmetic and Logic Unit*) component.
- **BIU** (*Bus Interface Unit*) - prepares the execution of every machine instruction. Reads an instruction from memory, decodes it and computes the memory address of an operand, if any. The output configuration is stored in a 15 bytes buffer, from where EU will take it.



EU and **BIU** work in parallel – while **EU** runs the current instruction, **BIU** prepares the next one. These two actions are synchronized – the one that ends first waits after the other.

2.6.2. The EU general registers

EAX - *accumulator*. Used by most of instructions as one of their operands.

EBX – *base register* (used with that meaning in 16 bits programming).

ECX - *counter register* – mostly used as numerical upper limit for instructions that need repetitive runs.

EDX – *data register* - frequently used with EAX when the result exceeds a doubleword (32 bits).

"Word size" refers to the number of bits processed by a computer's CPU in one go (these days, typically 32 bits or 64 bits). Data bus size, instruction size, address size are usually multiples of the word size. So, for a CPU the "word size" is a basic attribute/feature influencing the above mentioned elements.

Just to confuse matters, for backwards compatibility, Microsoft Windows API defines a WORD as being 16 bits, a DWORD as 32 bits and a QWORD as 64 bits, regardless of the processor. So, WORD and DWORD DATA TYPES are ALWAYS on 16 and 32 bits respectively FOR THE ASSEMBLY LANGUAGE , regardless of the CPU's "word size" (16, 32 or 64 bits CPU).

ESP and **EBP** are *stack* registers. The stack is a LIFO memory area.

Register **ESP** (*Stack Pointer*) points to the last element put on the stack (the element from the top of the stack).

Register **EBP** (*Base pointer*) points to the first element put on the stack (points to the stack's basis).

EDI and **ESI** are *index registers* usually used for accessing elements from bytes and words strings. Their functioning in this context (*Destination Index* and *Source Index*) will be clarified in chapter 4 (Instructions).

EAX, EBX, ECX, EDX, ESP, EBP, EDI, ESI are doubleword registers (32 bits). Every one of them may also be seen as the concatenation of two 16 bits subregisters. The upper register, which contains the most significant 16 bits of the 32 bits register, doesn't have a name and it isn't available separately. But the lower register could be used as single so we have the 16 bits registers **AX, BX, CX, DX, SP, BP, DI, SI.** Among these registers, AX, BX, CX and DX are also a concatenation of two 8 bits subregisters. So we have **AH, BH, CH, DH** registers which contain the most significant 8 bits of the word (the upper part of AX, BX, CX and DX registers) and **AL, BL, CL, DL** registers which contain the least significant 8 bits of the word (the lower part).