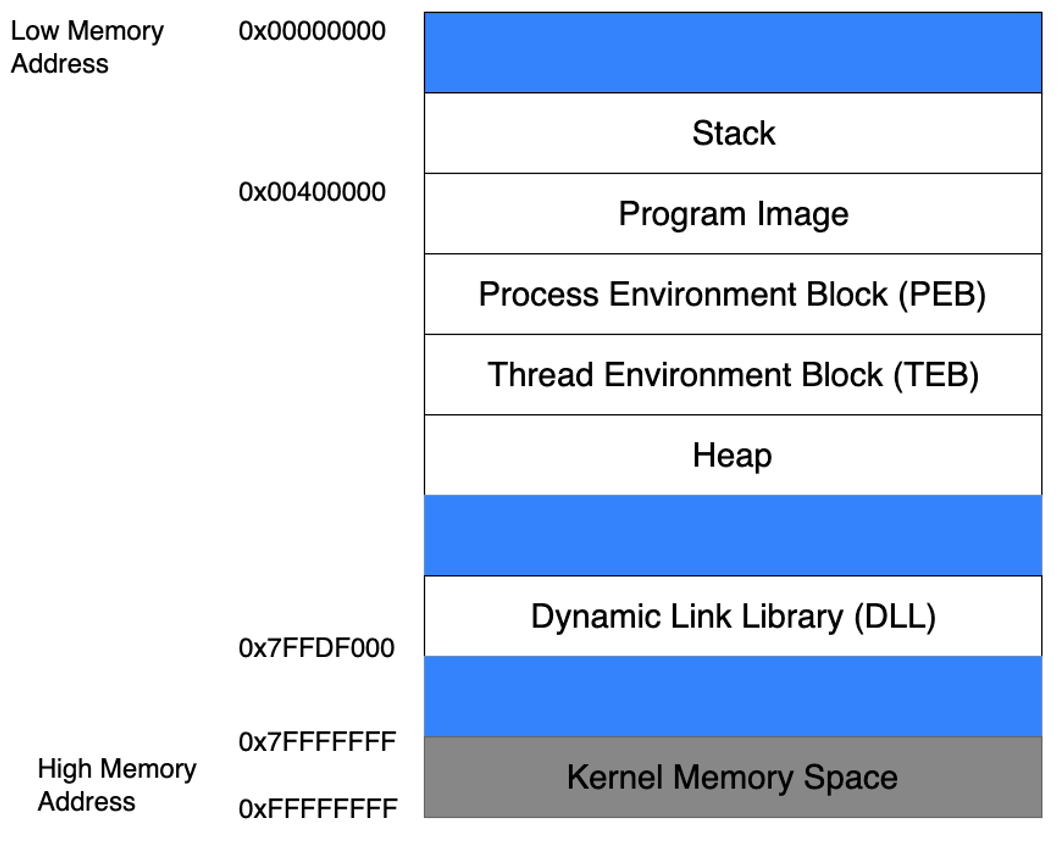
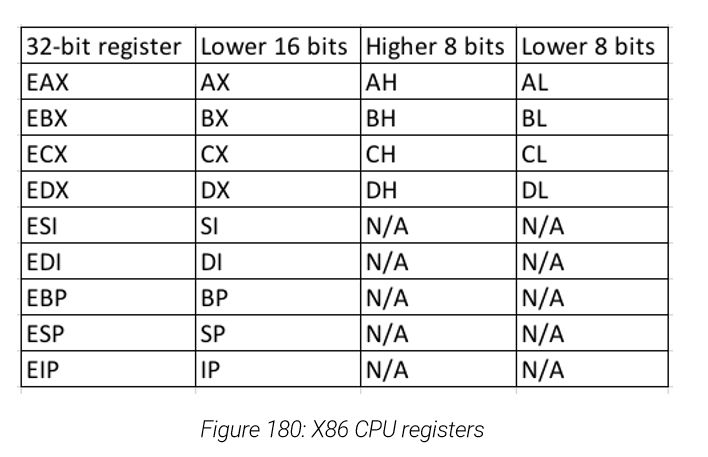
Program memory :

* Lowest address is 0x00000000, while the highest address is 0x7FFFFFFF, after this address to 0xFFFFFFFF, is the kernel space.
* 
* When a thread is running, it executes code from within the program Image or from various Dynamic link libraries ( DLL ).
* The thread requires a short-term data area for functions, local variables, and program control information, which is known as stack.
* Each thread has it’s own stack.
* Stack memory is “viewed” by the CPU as a Last-In First-Out (LIFO) structure.
* A long-term and more dynamic data storage area may also be needed, which is called the heap, which will be explained later.
* There are nine 32-bit CPU registers (on a 32-bit platform).
* 
* General Purpose Registers include EAX,EBX,ECX,EDX,ESI and EDI.
* EAX (accumulator): Arithmetical and logical instructions.
* EBX (base): Base pointer for memory addresses
* ECX (counter): Loop, shift, and rotation counter
* EDX (data): I/O port addressing, multiplication, and division
* ESI (source index): Pointer addressing of data and source in string copy operations
* EDI (destination index): Pointer addressing of data and destination in string copy operations
* As previously mentioned, the stack is used for storage of data, pointers, and arguments
* Since the stack is dynamic and changes constantly during program execution, ESP, the stack pointer, keeps “track” of the most recently referenced location on the stack (top of the stack) by storing the address of it.
* EBP - The Base Pointer : Since the stack is in constant flux during the execution of a thread, it can become difficult for a function to locate its own stack frame
* which stores the required arguments, local variables, and the return address.
* By accessing EBP, a function can easily reference information from its own stack frame (via offsets) while executing.
* EIP - The Instruction Pointer