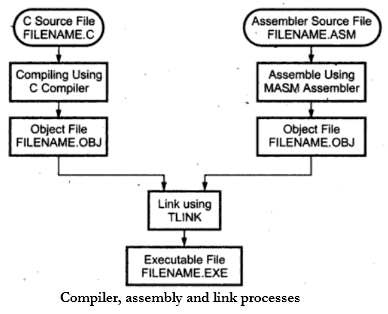
**MIXED MODE PROGRAMMING**

1. There are times when **programs need to call programs written in other languages referred as mixed language programming**. For example, when a particular subprogram is available in a language other than language you are using, or when algorithms are described more naturally in a different language, you need to use more than one language.
2. **Mixed-language programming always involves a call to a function, procedure, or subroutine**. Mixed-language calls involve calling functions in separate modules. **Instead of compiling all source programs with same compiler, different compilers or assemblers are used as per the language used in the programs**.
3. **Microsoft C supports this mixed language programming**. So it can combine assembly code routines in C as a separate language.
4. C program calls assembly language routines that are separately assembled by-MASM (MASM Assembler). These assembled modules are linked with the compiled C modules to get executable file. Fig shows the compile, assemble and link processes using C compiler, MASM assembler, and TUNIC.



Rules:

* Register Naming: Register names are prefixed with %, so that registers are %eax, %cl etc, instead of just eax, cl. We use %% for accessing the registers.
* Ordering of operands: Unlike Intel convention (first operand is destination), the order of operands is source(s) first, and destination last. For example, Intel syntax "mov eax, edx" will look like "mov %edx, %eax" in AT&T assembly.
* Operand Size: the size of memory operands is determined from the last character of the op-code name. The suffix is b for (8-bit) byte, w for (16-bit) word, and l for (32-bit) long. For example, the correct syntax for the above instruction would have been "movl %edx, %eax".
* Immediate Operand: Immediate operands are marked with a $ prefix, as in "addl $5, %eax", which means add immediate long value 5 to register %eax).
* Memory Operands: Missing operand prefix indicates it is a memory-address; hence "movl $bar, %ebx" puts the address of variable bar into register %ebx, but "movl bar, %ebx" puts the contents of variable bar into register %ebx.
* Indexing: Indexing or indirection is done by enclosing the index register or indirection memory cell address in parentheses. For example, "movl 8(%ebp), %eax" (moves the contents at offset 8 from the cell pointed to by %ebp into register %eax).

