

The background of the slide features a dark blue gradient. Overlaid on this are several lines of binary code (0s and 1s) in a lighter blue color, arranged diagonally from the top-left towards the bottom-right. Additionally, there are large, semi-transparent blue geometric shapes, including triangles and parallelograms, that intersect with the binary text.

系統程式期末報告

組合語言

組合語言是把「機器指令」(或稱「機器碼」)表示成「文字指令」所構成的程式語言，使用組合語言所撰寫的程式需經 Assembler(組譯器)將文字指令編譯成機器碼，所以組合語言可說是機械語言的進化。

· 執行時的順序: ·

組合語言 ----- · 組譯器 ----- · 機械碼 ·

MOV AH 09 Assembler 00010010

組譯器(Assembler)：

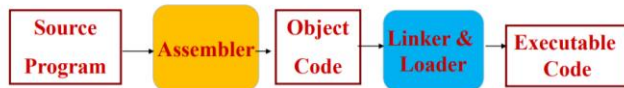
組譯器用來將組合語言使用的原始檔(.asm)翻譯成含有機器碼的二進制目的碼(Object Code)，通常目的檔為16位元的2進制指令檔

。

連結器(Linker)：


連結器用來將一個或多個目的碼檔案連結函式庫為一個可執行檔案。

組譯器 Assemblers



組譯器的基本功能

- 1.將助憶碼 (opcode, 組合語言指令) 轉換成機器碼
- 2.指定機器位址給符號標籤
- 3.組合語言 -> 目的碼 (object code)



組譯程式之類型:

One-Pass Assembler(單回合組譯程式 單回合組譯程式)

Two-Pass Assembler(雙回合組譯程式)

Multi-Pass Assembler(多回合組譯程式)

組譯程式之基本功能:

將助憶碼轉成對應之機器碼

指機位所義符號標定機器位址給所定義的符號標記(symbolic label)

處理虛擬指令(假指令、組譯器指引命令)

處理前向參考(forward reference)

產生報表檔(listing)

產生目的碼(object code)



高階語言

組合語言為低階語言，語言稱之為低階或高階在於其較接近機器碼或較接近人類語言，較接近機器碼的語言稱為低階語言，較接近人類語言的語言稱為高階語言。

高階語言以類似英文和數學的型態出現，將所能使用者可能用到的功能、函數等，全都內建在語言之中，使用者所需要的，高階語言中都有，由於有了編譯器(compiler)，使得程式語言有極大的進步，不必再使用艱澀難懂的機械語言，可以用類似人類語言的程式語言，來撰寫程式。

編譯器 (COMPILER)

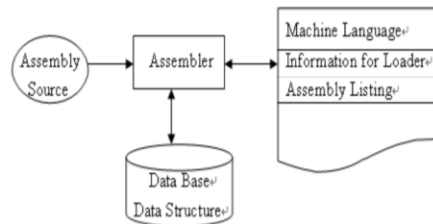
編譯器就是一種能接受高階語言程式當作輸入,而產生等效的機械語言為輸出的軟體程式,例如Turbo C、C ++,Borland C ++,Microsoft C,Visual C++ etc.

編譯器基本上由語言分析 (Lexical Analysis),語法分析 (S t A l i)(Syntax Analysis),編碼產生器(C d G t)(Code Generator),符號表(Symbol Table) 所組成。 ·

原始程式 --> 經由編譯器 --> 變成目的檔 --> 經由連結器 --> 變成執行檔 ·

SOURCE.C --> COMPILER --> SOURCE.OBJ --> LINKER --> SOURCE.EXE

組譯器之結構



假指令(Assembler Directive)

並非機器程式語言的助記碼

告訴組合程式如何做組譯工作的指令。(亦即可改變 組合程式本身的狀況)

並不會被轉入到 Object Program

例如:

START

END

BYTE、WORD (宣告型態

RESB、RESW (保留緩衝區)