**東海大學資訊工程學系**

編譯器期末報告

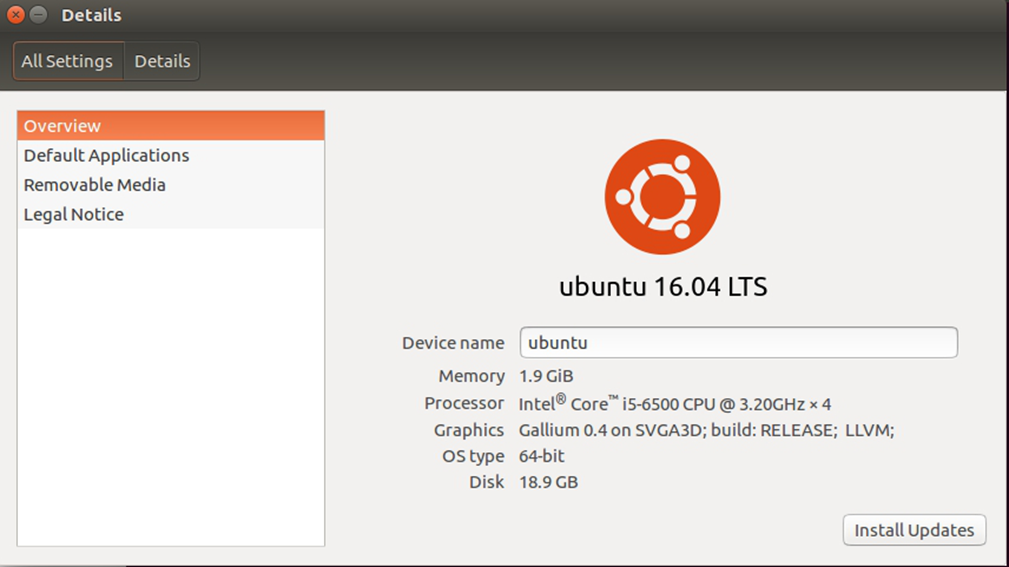
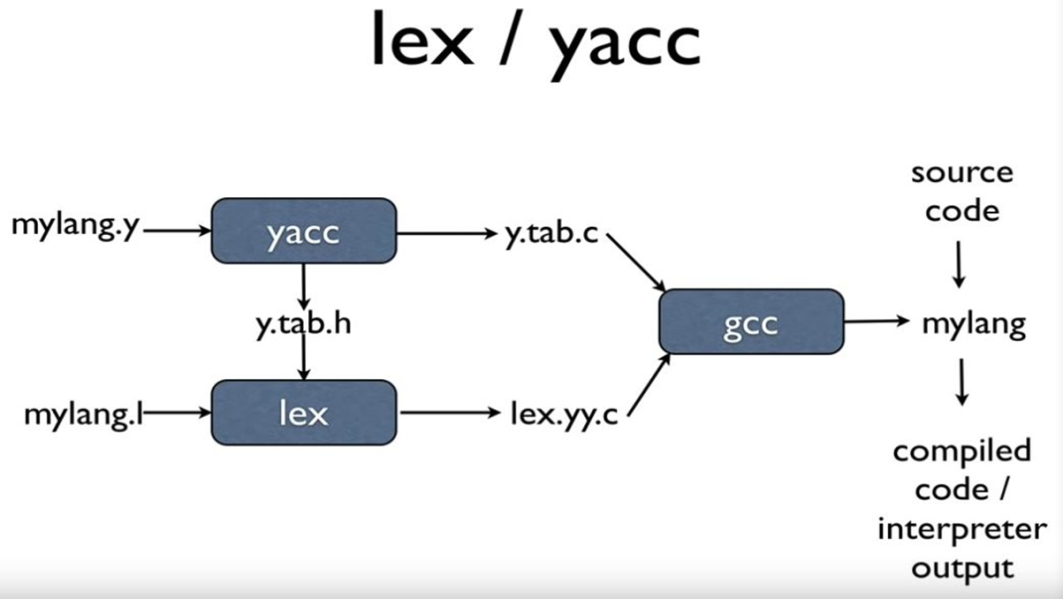
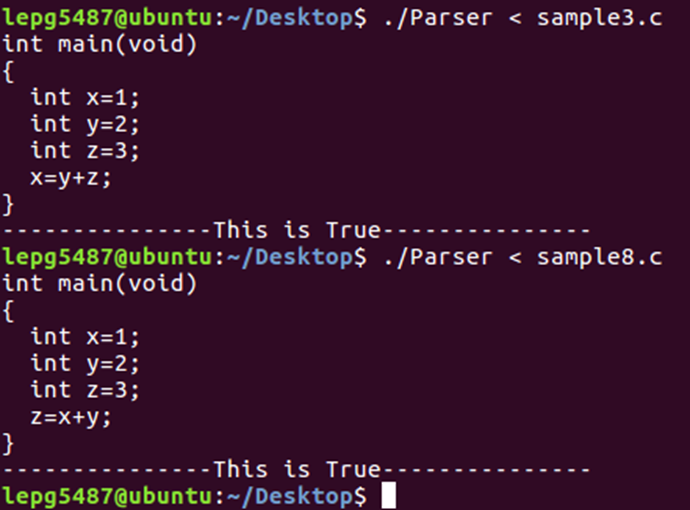
Lex & Yacc 建製 C grammar 的 Parser

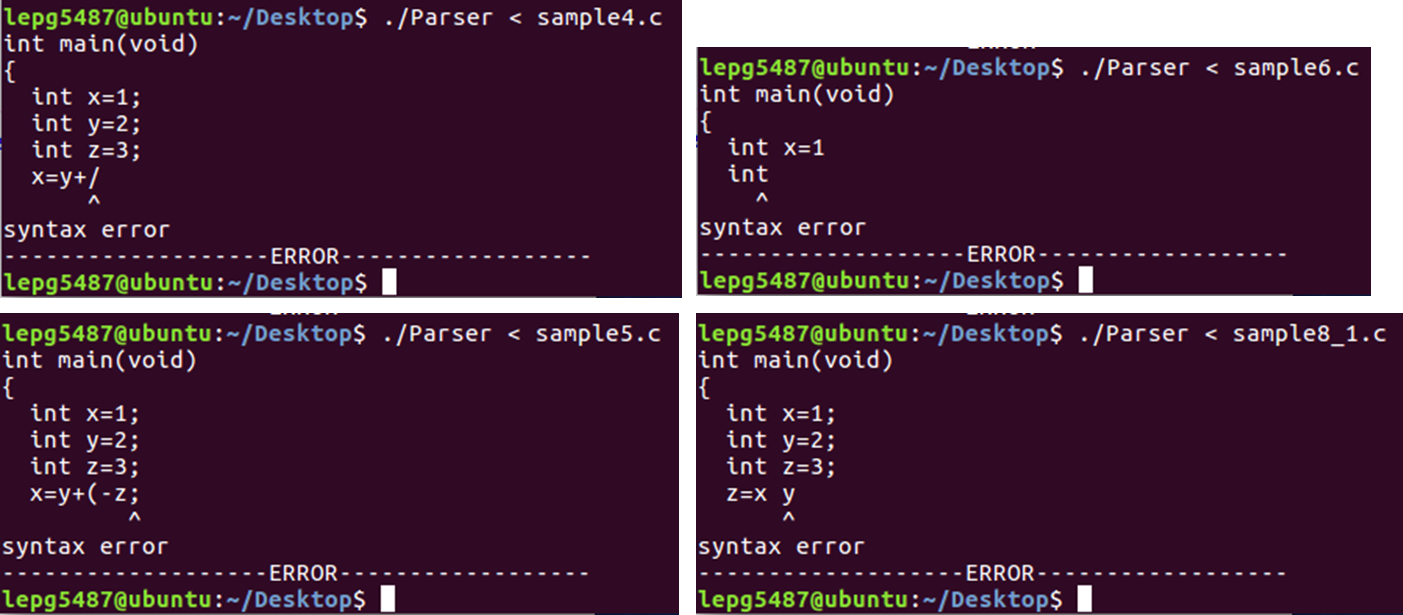
LLVM 產生 x86組合語言

姓名：**李彥葦、尤詩涵**

學號：S04353044、S04353041

指導教授：楊朝棟教授

* **系統環境：**
* 軟硬體版本Ubuntu 16.04 LTS虛擬機
* **安裝YACC與LEX**
* sudo apt-get install flex → install Lex package
* sudo apt-get install bison → install Yacc package
* sudo apt-get update → 出現錯誤更新
* **使用YACC與LEX進行文法分析**
* lex lex.l → 產生lex.yy.c
* yacc -d yacc.y → 產生 y.tab.c 和 y.tab.h
* gcc lex.yy.c y.tab.c -o Parser → 產生執行檔
* ./Parser<sample1.c → 對sample1.c進行文法分析
* 文法正確時 → 產生 This is True
* 文法錯誤時 → 產生 Error

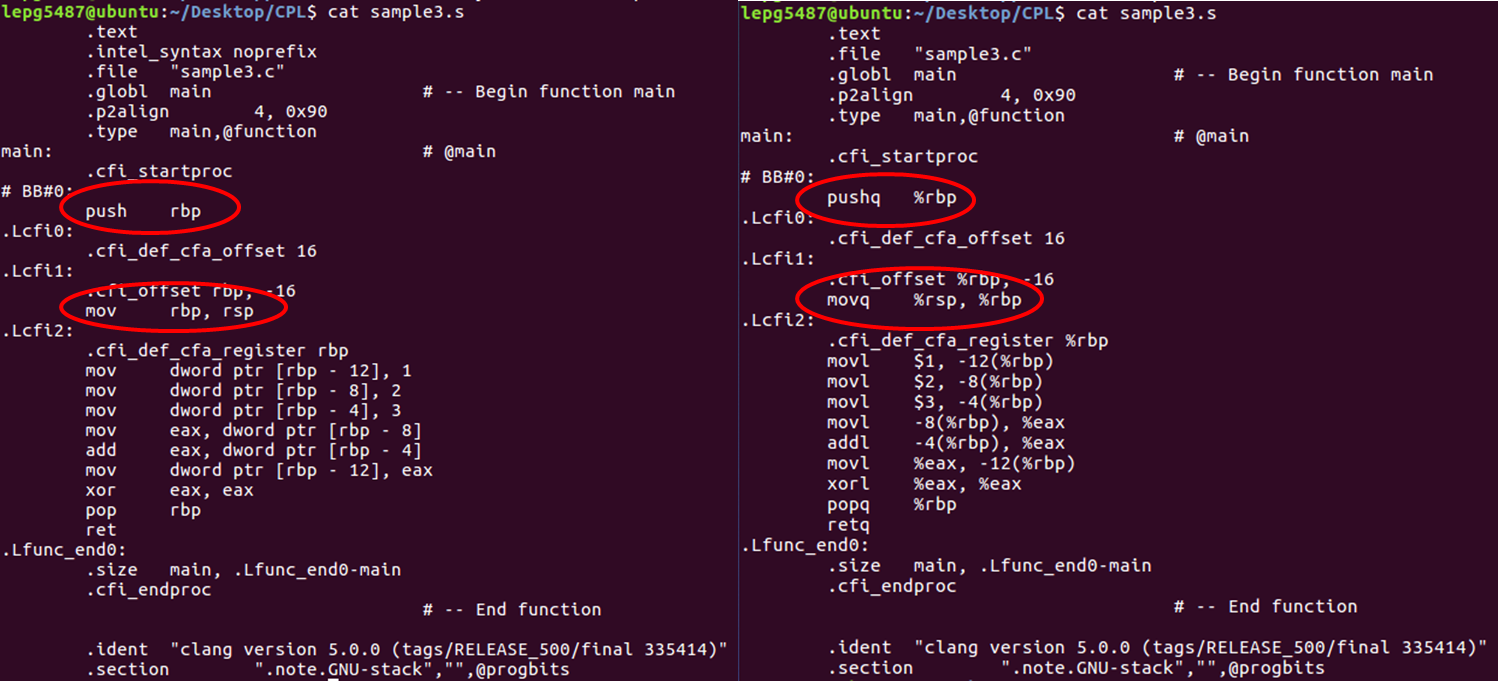


* **安裝LLVM**
* sudo apt install subversion → install subversion
* sudo apt install cmake → install subversion
* **下載LLVM**
* mkdir llvm-source-build → 新增一個資料夾
* cd llvm-source-build → 到資料夾裡
* svn co http://llvm.org/svn/llvm-project/llvm/tags/RELEASE\_500/final llvm → 下載llvm
* **下載Clang**
* cd llvm/tools → 到資料夾裡
* svn co http://llvm.org/svn/llvm-project/cfe/tags/RELEASE\_500/final clang → 下載Clang
* **下載Clang 工具**
* cd clang/tools → 到資料夾裡
* svn co http://llvm.org/svn/llvm-project/clang-tools-extra/tags/RELEASE\_500/final extra → 下載Clang 工具
* **下載Compiler-RT**
* cd
* cd projects → 到資料夾裡
* svn co http://llvm.org/svn/llvm-project/compiler-rt/tags/RELEASE\_500/final compiler-rt → 下載Compiler-RT
* svn co http://llvm.org/svn/llvm-project/libcxx/tags/RELEASE\_500/final libcxx → 下載libcxx
* svn co http://llvm.org/svn/llvm-project/libcxxabi/tags/RELEASE\_500/final libcxxabi → 下載libcxxabi
* **建立bulid資料夾**
* cd llvm-source-build → 到資料夾裡
* mkdir bulid → 新增一個資料夾
* cd build → 到資料夾裡
* cmake -G "Unix Makefiles" -DCMAKE\_BUILD\_TYPE=Release -DLLVM\_TARGETS\_TO\_BUILD="X86" ../llvm → 建立Makefiles
* make (約2小時)
* sudo make install
* **使用LLVM產生IR中間碼(.ll)**
* clang -S -emit-llvm sample3.c → 產生sample3.ll
* **將IR (.ll)編譯成組合語言**
* llc -x86-asm-syntax=intel sample3.ll → 產生sample3.s  
  (編譯成x86組合語言)
* llc -x86-asm-syntax=att sample3.ll →產生sample3.s   
  (產生AT&T/UNIX語法(預設))
* llc sample3.ll → sample3.s (產生AT&T/UNIX語法(預設))
* **產生執行檔**

as sample3.s -o sample3.o → 產生sample3.o  
ld sample3.o -o sample3 → 產生執行檔

* **Intel和AT&T/UNIX語法差異**
* AT&T/UNIX：Register(暫存器)的名稱前面必須加上%。  
  Intel x86：不用。
* AT&T/UNIX：的source(來源)在左邊，destination(目的)在右邊。  
  Intel x86：相反。
* AT&T/UNIX：立即值要加$。  
  Intel x86：不用。
* AT&T/UNIX：b、w和l，分別代表byte、word和long。

Intel x86：不用。等等……

* **心得**

在參考學長姐的資料和網路上的文獻後，了解到更多編譯器的運行原理與過程，加深了我們對編譯器的印象，也學習到許多Linux的知識。利用 Lex 和 Yacc 來幫助自動產生Parser，原本不了解如何產生組合語言，後來發現 gcc、LLVM能產生組合語言，但是又發現產生的組合語言不是之前所教過的，花費許多時間在找能解決此問題的指令。

* **參考資料**

鳥哥的Linux：

http://linux.vbird.org/linux\_basic/0520source\_code\_and\_tarball.php#simple\_ex\_gcc

深入淺出教你寫編譯器：

http://inspiregate.com/programming/other/471-compiler-1.html

AT&T/UNIX、Intel語法差異：

http://nano-chicken.blogspot.com/2010/12/inline-assembly.html

gcc 編譯器的常用參數及其意義：

http://ccckmit.wikidot.com/gnu:gcc

objdump 指令的使用方法與常用參數：

http://ccckmit.wikidot.com/gnu:objdump

如何快速比對C和組合語言：

https://wirelessr.gitbooks.io/working-life/content/gdbyu\_objdump.html

安裝LLVM 和 Clang：

https://www.jianshu.com/p/861c1a630059

實作LLVM ：

https://medium.com/@zetavg/%E7%B7%A8%E8%AD%AF%E5%99%A8-llvm-%E6%B7%BA%E6%B7%BA%E7%8E%A9-42a58c7a7309

LLVM指令：

http://llvm.org/docs/CommandGuide/