



# 微算機系統實習

## MICROPROCESSOR SYSTEMS LAB.

### SPRING, 2021

Instructor: Yen-Lin Chen(陳彥霖), Ph.D.

Professor

Dept. Computer Science and Information Engineering  
National Taipei University of Technology

# 作業評分方式

- 實驗部分佔該次實驗的總比例70%  
報告總分佔該次實驗的總比例30%  
\* 當次實驗會因為難易度不同佔學期總成績的實驗分數比例也不同
- 實驗報告上傳格式, 公告於i學園
- 以小組為單位繳交報告

請同學上傳報告時依照上面的格式上傳,

- 一、組別與組員名單
- 二、實驗步驟截圖與說明
- 三、組員貢獻比例 (組員%數加總必須等於100%)
- 四、個人心得
  - 實驗報告單獨分數為100分, 以上第一、三、四點每缺少一項報告分數扣20分。第二點缺少報告扣40分。另外第二、四點不完整會依照狀況扣分, 最高扣到該項目的上限分數。
  - 貢獻比例分配方式為: 報告總分\*2\*組員貢獻比例=組員報告得分

# 作業繳交

- 基本繳交時間
  - 實驗: 公布實驗後當天(3/5)上課結束前(18:30)
  - 報告: 公布實驗後的隔週星期四(3/11)結束前(23:59)
    - \* 若有因為特殊原因繳交時間有變動助教會另外公布
- 超過時間遲交每隔一週(含一週內)分數打8折, 採累計連乘方式, 實驗與報告打折是分開算的
  - 舉例:
    - 遲交三天—以遲交一週計算  $\text{<遲交的項目單獨分數>} * 0.8 = \text{該項目得到的分數}$
    - 遲交九天—以遲交兩週計算  $\text{<遲交的項目單獨分數>} * 0.8 * 0.8 = \text{該項目得到的分數}$
- 以上配分與注意事項有問題請聯絡助教

# 說明

請依組別與助教領取TX2

- ▶ 領取TX2時，**須登記MAC**
- ▶ 該台TX2即為本學期組員共用機台
- ▶ 每次實作機台為該組固定使用
- ▶ 請好好愛惜機台

若板子有任何問題

請通知助教

- 不可自行燒錄TX2
- 作業系統及Kernel



# 本次實驗目標

- 學習如何使用SSH與其它機台進行連線
- 學習如何撰寫Makefile編譯程式
- 學習如何使用跨平台編譯開發嵌入式系統程式

# 實驗一



## 跨平台連線教學

# 確認乙太網路

- 先確認目前電腦要分享的網路連線。



請參閱

Windows Defender 防火牆

網際網路選項

# 共用網路

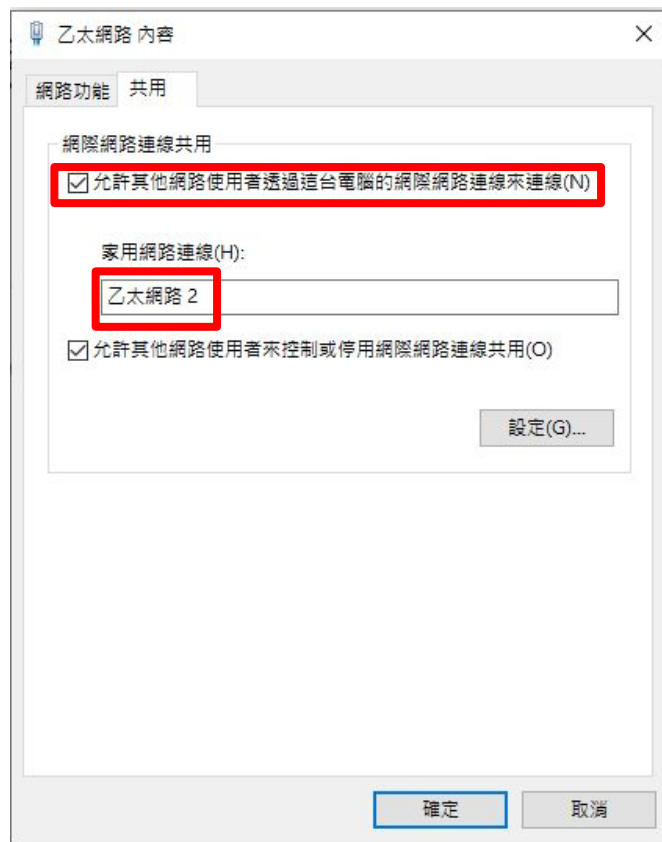
- 請將電腦旁的網路線連接到TX2，出現乙太網路2，右鍵乙太網路開啟共用網路功能。





# 共用網路

- 勾選允許「其他網路使用者...來連線」
- 家用網路連線，選取乙太網路2



# 共用網路

- 在windows底下打開命令提示字元，輸入arp -a 看IP。
- 對TX2的mac號碼，找到TX2 IP。
- 回到虛擬機
- 教室電腦環境上課完會重置，每次使用皆須重新設定網路連線

```
介面: 192.168.137.1 --- 0x13
網際網路網址      實體位址      類型
192.168.137.53     00-04-4b-8c-5b-31 靜態
192.168.137.255    ff-ff-ff-ff-ff-ff 靜態
224.0.0.22         01-00-5e-00-00-16 靜態
224.0.0.251        01-00-5e-00-00-fb 靜態
224.0.0.252        01-00-5e-00-00-fc 靜態
239.255.255.250    01-00-5e-7f-ff-fa 靜態
255.255.255.255    ff-ff-ff-ff-ff-ff 靜態
```

# 跨平台連線教學

- 請使用VM上的Ubuntu系統連到TX2
- 在VM上使用ssh的指令連到TX2
  - ssh -Y 登入帳號@TX2板子IP
  - 範例使用192.168.137.53, 請同學輸入自己TX2所分配的IP

```
nvidia@tegra-ubuntu: ~  
nvidia@ubuntu:~$ ssh -Y nvidia@192.168.137.53  
The authenticity of host '192.168.137.53 (192.168.137.53)' can't be established.  
ECDSA key fingerprint is SHA256:oTgKigtyuJgE0GInUM9EkZ0bS6bfjvfr2ENHU7j+1wE.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '192.168.137.53' (ECDSA) to the list of known hosts.  
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.38-tegra aarch64)  
  
* Documentation:  https://help.ubuntu.com/  
  
665 packages can be updated.  
424 updates are security updates.  
  
Last login: Wed Feb 26 12:31:44 2020  
nvidia@tegra-ubuntu:~$
```

這樣代表連線成功



# 運用MAKEFILE編譯程式專案

# MAKEFILE- 自動化變數

- $\$@$ : 工作目標檔名。
- $\$<$ : 第一個被依賴文件的檔名。
- $\$^$ : 所有被依賴文件的檔名, 並以空格隔開這些檔名
- $\$?$ : 被依賴文件中有更改過的所有檔名。

```
#用「井」號表明注釋。  
target (要生成的文件): dependencies (被依賴的文件)  
#命令前面用的是「tab」而非空格。誤用空格是初學者容易犯的錯誤。  
命令1  
命令2  
命令3  
.  
.  
命令n  
#可以使用「\」表示續行。注意,「\」之後不能有空格。
```

# MAKEFILE-變數(巨集)

- CXX : C++編譯器的名字, 預設值是g++。
- CXXFLAGS : C++編譯器的選項, 沒有定義。
- 例如: OBJ = hellofuc.o hellomake.o
- 使用時在前面加 \$( ) 的符號, 如: \$(OBJ)

```
CXX=g++
CXXFLAGS=-std=c++11
OBJ=hellofuc.o hellomake.o
SRC=hellofuc.cpp hellomake.cpp
EXE=hellomake #filename
all:$(EXE)
$(EXE):$(OBJ)
    $(CXX) -o $(EXE) $(OBJ)
$(OBJ):$(SRC)
    $(CXX) -c $(SRC)
clean:
    rm -f $(EXE)
    rm -f $(OBJ)
```

本次實驗需使用C++11  
的library, 需加此部分

## ❖ Example:

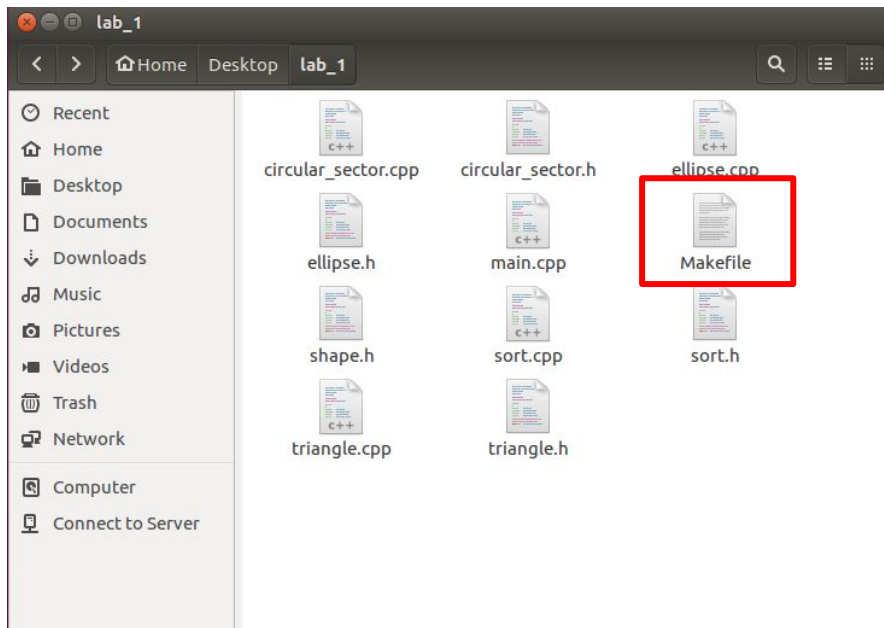
- ❑ CROSS\_COMPILE = aarch64-linux-gnu- (TX2是使用ARMv8指令集)
- ❑ CXX = \$(CROSS\_COMPILE)g++ /\* aarch64-linux-gnu-g++ \*/
- ❑ LD = \$(CROSS\_COMPILE)ld /\*aarch64-linux-gnu-ld \*/



# 跨平台嵌入式程式開發

# 實驗一 SOURCE CODE

- 請同學從i學園下載lab1\_source\_code附件檔



- 程式碼內容不需更動，僅需自己編寫Makefile，讓程式可以成功編譯與執行。



# 實驗一 範例程式說明

- 本次實驗提供下列檔案
- `circular_sector`(扇形)、`ellipse`(橢圓形)及`triangle`(三角形)等種形狀的`.cpp`檔和`.h`檔，程式內容為計算各形狀的周長及面積
- `sort`檔中，透過各形狀依照面積進行升冪排列
- `main`檔中設定各形狀所需的參數，輸出的結果會將各形狀依照面積升冪排列
- 透過本次實驗多種形狀面積的排序，學習如何編寫`Makefile`檔，並完成跨平台編譯。

# 實驗一 要求

- ◆ 項目一: Makefile要有target: all (20%)
  - 編譯出跨平台可執行檔
- ◆ 項目二: Makefile要有target: clean (20%)
  - 刪除該執行檔與所有.o檔
- ◆ 項目三: 在 Makefile 中使用變數, 可參考12頁 (10%)
- ◆ 項目四: 在 Makefile 裡須加入 \$@ (5%), \$< or \$? (5%)
- ◆ 項目五: 將虛擬機CROSS\_COMPILE後的執行檔傳至TX2, 且以ssh連到TX2執行 (10%)
- ◆ 項目六: 實驗報告與Makefile檔 (30%)

# MAKEFILE 範例

---

```
all:hellofuc.o hellomake.o
    g++ -o hellomake hellofuc.o hellomake.o
hellofuc.o:hellofuc.cpp
    g++ -c -o hellofuc.o hellofuc.cpp
hellomake.o:hellomake.cpp
    g++ -c -o hellomake.o hellomake.cpp
clean:
    rm -f hellomake
    rm -f hellomake.o
    rm -f hellofuc.o
```

# 實驗一 執行結果

- 將編譯的跨平台執行檔傳輸至TX2(可利用scp傳輸)
- 在終端機(利用SSH遠端進TX2介面)輸入ls，查看home目錄下的檔案，若tx2\_exe有出現，代表傳送成功
- 在終端機(TX2)執行tx2\_exe，如果出現如下畫面，代表完成跨平台編譯

```
nvidia@tegra-ubuntu:~$ ls
Desktop      NVIDIA_CUDA-9.0_Samples  Templates
Documents    opencv-2.4.9             tx2_exe
Downloads    Pictures                  Videos
examples.desktop  Public                  VisionWorks-SFM-0.90-Samples
jetson_clocks.sh  tegra_multimedia_api     weston.ini
Music         tegrastats
```

在TX2上的終端機

```
nvidia@tegra-ubuntu:~$ ./tx2_exe
原本的: 62.8319 78.5398 52.3599 50
排序後: 50 52.3599 62.8319 78.5398
作業一完成
```

輸出結果如此處