嵌入式作業系統 Embedded Operating Systems

LAB2

312512005 黃名諄

1. 移除不必要的功能, 縮小RPi OS 的 image size (應說明: 目的、移除哪些功能、如何移除、前後image size與功能的比較)

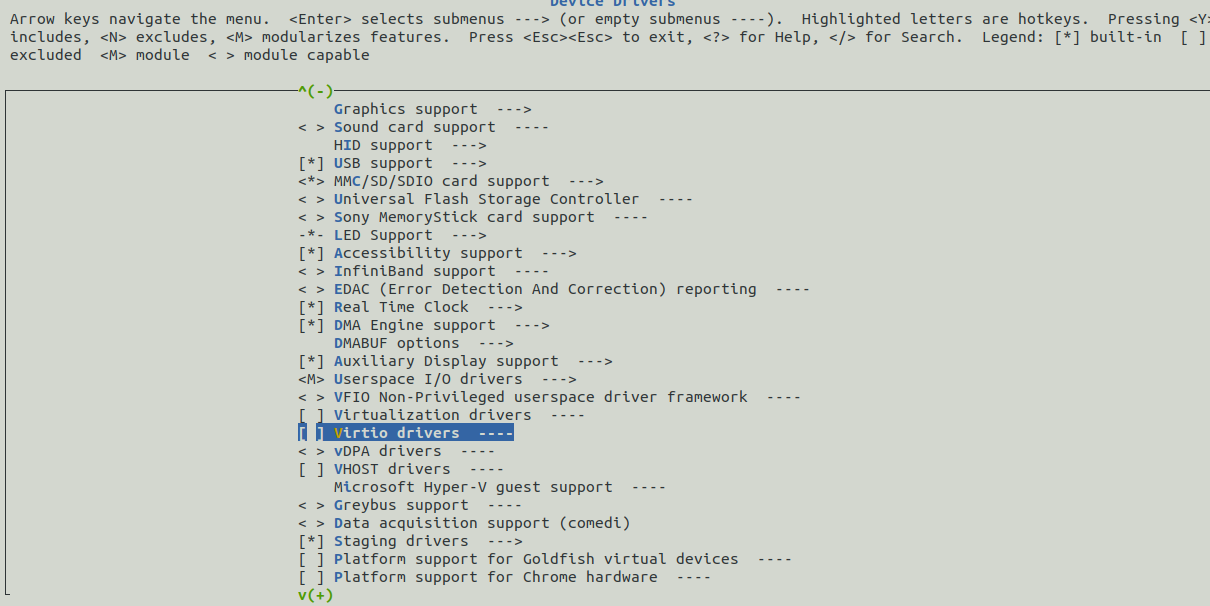
嵌入式系統是為了執行特定功能所設計，因此在需考量到實際在應用上所需要的特定目的去做為設計依據，因此在做此lab縮減image size時，我也是先思考了應用情境，再根據此情境需求及目的去做縮減。我以目前實驗室研究做發想將應用設定為自主式移動機器人平台上之嵌入式系統，自主式移動機器人簡單來說就是類似餐廳中送餐機器人的概念，可自主導航並移動到目標位置，其中整個系統需要特定裝置及算法來處理，例如要處理定位、路徑規劃、避障、底盤控制等等演算法，非常需要能及時且低延遲的運算，且需要sensors來感知環境，因此需求如下:

* 盡可能提升運算速度，使各演算法能互相及時配合
* 降低延遲，使各系統能及時完成計算及互相回應
* 數據紀錄，讀寫速度盡可能提高
* Sensor 如光達、相機靠有線連接，如usb、ethernet，其他不需要的裝置可移除
* 縮減image size，使memory有更多空間能給user program使用

先透過make ARCH=arm64 CROSS\_COMPILE=aarch64-linux-gnu- menuconfig編輯.config，在先前default config之上進行刪減修改，以下是我針對我的需求所做的改動:

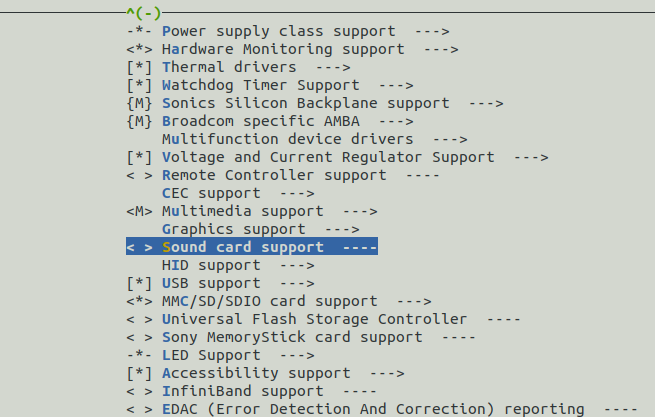
1. 關**Virtualization** :因為我的情境需求上沒有在RPi 上建虛擬機需求，可關閉虛擬化功能
2. 一張含有 文字, 電子產品, 螢幕擷取畫面, 陳列 的圖片

   自動產生的描述在device driver中，也關閉虛擬化相關的選項: 關**Virtio drivers** 及 **VHOST drivers**



1. 在device driver中，關閉不必要的裝置驅動:

* 不需要音效，關閉**Sound card support**
* 不會用到遙控設備，關閉**Remote controller support**

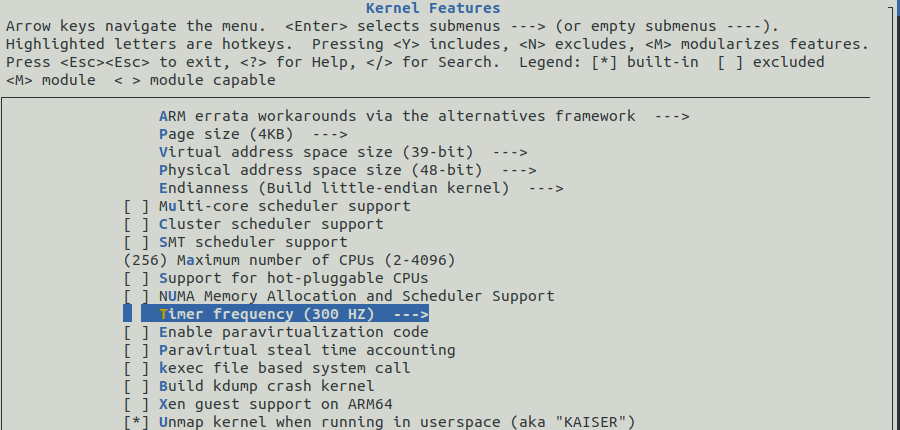


1. 在Networking support中
   * sensor都用有線連接，不需要藍芽，關閉**Bluetooth subsystem support**
   * 一張含有 文字, 字型, 軟體, 數字 的圖片

     自動產生的描述一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

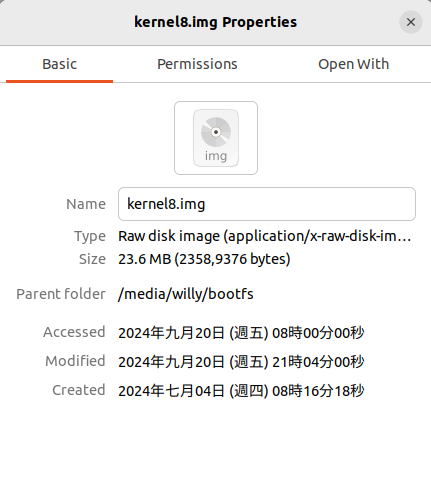
     自動產生的描述不會用到無線電設備，關閉**Amateur Radio support**
2. 一張含有 文字, 電子產品, 螢幕擷取畫面, 軟體 的圖片

   自動產生的描述在CPU Power Management🡪CPU Frequency scaling中， 將**Default CPUFreq governor從預設的powersave改為performance，**讓OS以performance為優先讓CPU以較高frequency去執行以提升效能，我的應用情境相對於節省能源更要求**CPU能有高效能**去快速計算導航演算法。
3. 在Kernel Features中，將**Timer frequency從250Hz改為300Hz**

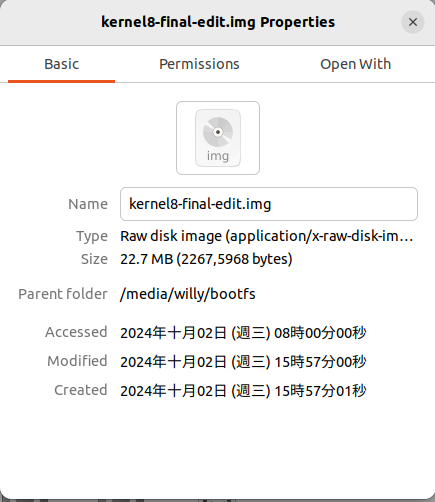
此選項會影響timer頻率，我希望能透過適當提高timer頻率，提升timer interrupt精度使CPU能更有效率執行各項演算法工作，達到**降低latency**的目的。但其實有試過改為最大值1000Hz，發現過高的Timer frequency反而使效能下降，原因我認為是過於頻繁的timer interrupt打斷CPU工作，反而會使其無法順暢工作，因此適度提高才是對系統效能提升最有效的。

* 更改完上述設置後儲存，再進行講義步驟5~9編譯及安裝image進SD卡

刪減前後之image size大小比較:



縮減前(default config)



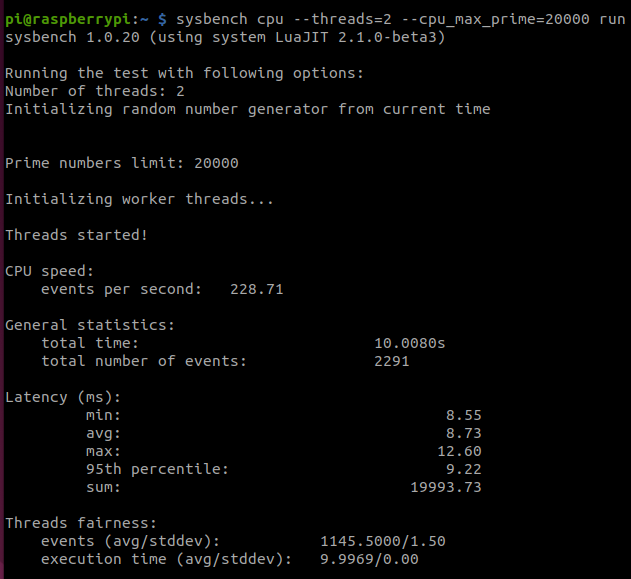
縮減後

可發現設置及移除部分功能後的kernel image大小確實變小了，使kernel占用更少的記憶體空間。

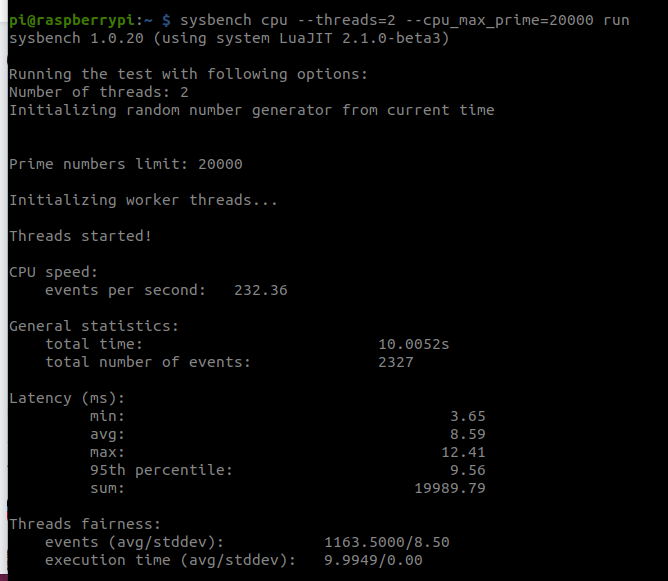
1. 在網路上下載可以評測 kernel 的套件, 比較原本與更改後的 kernel (應說明: 下載哪一套件、改善哪方面的效能與評測的結果)
   * 在網路搜尋後，決定使用**sysbench**這套評測套件，其能簡單評測CPU、memory、磁碟file I/O的效能
   * 安裝sysbench流程:
2. ssh進RPi，輸入sudo apt-get update更新資訊
3. sudo apt-get install -y sysbench 安裝套件
4. 輸入sysbench --version 檢查版本及是否安裝成功
5. CPU效能

指令: sysbench cpu --threads=2 --cpu\_max\_prime=20000 run

利用質數運算測試CPU效能，將最大值設為20000模擬工作量較高情況，且因為實際情況下較常用多核執行任務，因此設置使用2 threads，目的是評測使用多線程跑的效能



縮減前(default config)



縮減後

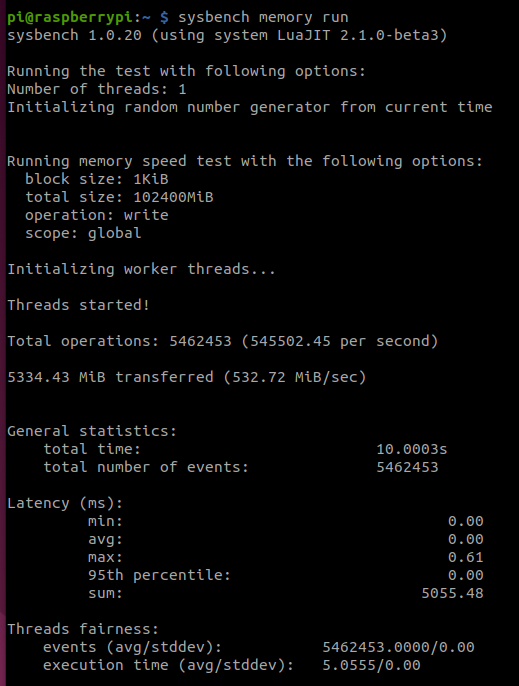
根據我前面設定的情境需求，我著重的效能改善應該是**CPU計算速度**及**latency**:

|  |  |  |
| --- | --- | --- |
|  | 縮減前 | 縮減後 |
| CPU speed  Events per second | 228.71 | **232.36** |
| Latency(ms) avg. | 8.73 | **8.59** |
| Latency(ms) max. | 12.60 | **12.41** |
| Latency(ms) min. | 8.55 | **3.65** |

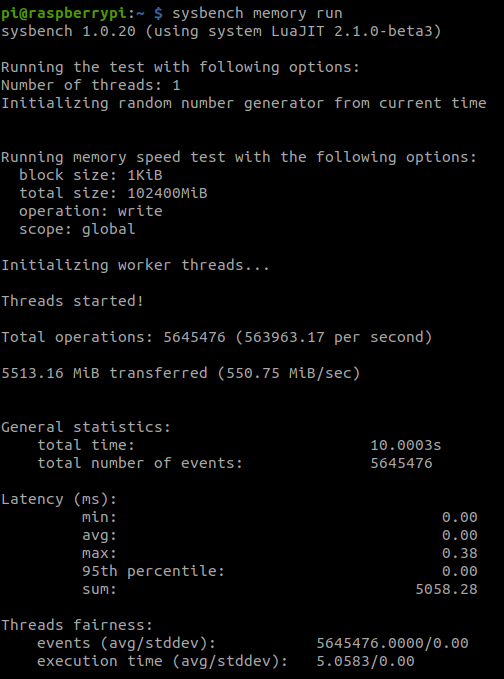
從評測結果可發現，刪減修改過後的kernel在CPU speed上有些許提升，符合我需要較高運算速度的目的，且另一方面我也希望latency越低越好，刪減修改過後的kernel確實也能有較低的平均latency，而且最大值及最小值都較低。因此在CPU方面，雖然提升效果沒有非常顯著，但確實有提升到我需要的效能。

1. Memory 效能

指令: sysbench memory run



縮減前(default config)



縮減後

根據我前面設定的情境需求，我著重的效能改善應該是**傳輸速度**及**latency**:

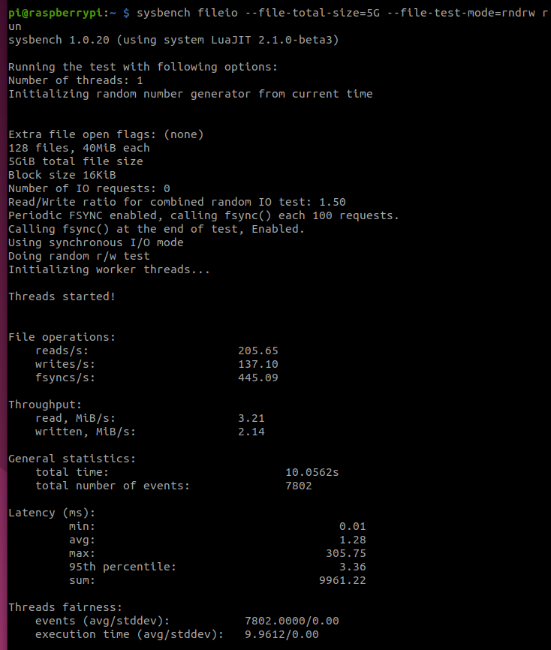
|  |  |  |
| --- | --- | --- |
|  | 縮減前 | 縮減後 |
| Transfer speed (MiB/s) | 532.72 | **550.75** |
| Latency(ms) sum | **5055.48** | 5058.28 |
| Latency(ms) max. | 0.61 | **0.38** |

從評測結果看來，修改過後的kernel能讓memory傳輸速度上升，而latency的平均和最小值都是0.00太小難以比較，因此改為比較最大值及總和，可發現修改過後的kernel能有效降低較大latency的發生，但總和上卻較高，對單一較大延遲的任務有改善。或許因為我沒有修改特別的memory設定，對降低總體latency較無效果，但刪減部分無用裝置驅動對memory的效能還是有部分提升效果，我認為是因為整體image縮減能提供更多空間給user program使用，能使其較少發生page fault需要去disk抓page的需求。

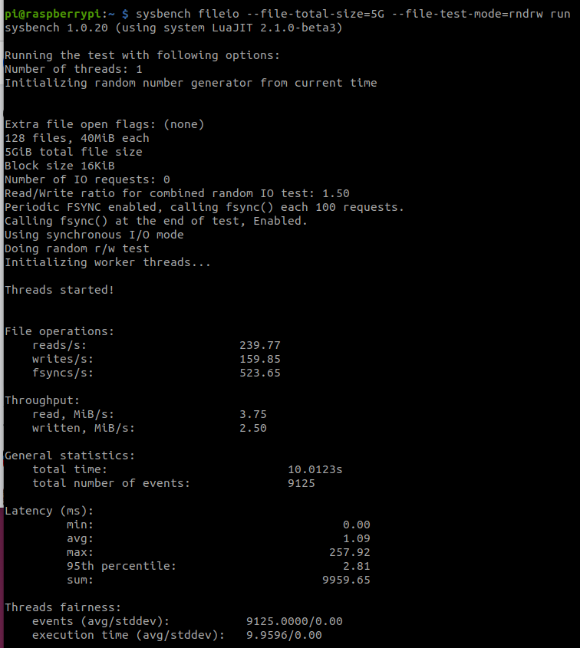
1. 磁碟file I/O的效能

指令:sysbench fileio --file-total-size=5G prepare:準備好5G的檔案做為評測使用

sysbench fileio --file-total-size=5G --file-test-mode=rndrw run



縮減前(default config)



縮減後

類似於memory，根據我前面設定的情境需求，我著重的效能改善應該是**讀寫速度**及**latency**:

|  |  |  |
| --- | --- | --- |
|  | 縮減前 | 縮減後 |
| Throughput read(MiB/s) | 3.21 | **3.75** |
| Throughput written(MiB/s) | 2.14 | **2.50** |
| Latency(ms) avg. | 1.28 | **1.09** |
| Latency(ms) max. | 305.75 | **257.92** |
| Latency(ms) min. | 0.01 | **0.00** |

從評測結果可看出，修改過後的kernel在磁碟file I/O方面效能都有所提升，不論是讀寫速度或是latency都較未修改的好，對我的機器人系統在做紀錄數據或讀取地圖等操作時都會有所幫助。

1. 使 kernel 支援 real-time 功能, 可參照講義的作法, 依照自己 kernel 的版本來 patch (應說明: 欲下載哪一版本的補丁、如何下載與下載的結果)

* 一張含有 文字, 螢幕擷取畫面, 字型 的圖片

  自動產生的描述原本是依照講義方式，查看kernel版本並下載對應版本補丁

但安裝後發現一張含有 文字, 電子產品, 螢幕擷取畫面, 字型 的圖片

自動產生的描述，此**6.1的補丁在我的RPi 3 B+上似乎不相容**，無法開機，因此便根據助教的hint，從步驟3: Get the Kernel Sources中更改branch的版本，clone較舊版本的kernel，查資料後有人用版本4.19在RPi 3 B+成功安裝，因此我也**使用4.19的kernel做RT-patch**，patch過程如下:

1. Clone 4.19的kernel sources至另外的一個資料夾linux\_rt

git clone --depth=1 --branch rpi-4.19.y https://github.com/raspberrypi/linux linux\_rt

1. 在clone下來的資料夾中查看kernel版本

cd linux\_rt

head Makefile -n 4

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

版本為4.19.127

1. 接講義step4，用default config來 build kernel sources and device tree files

KERNEL=kernel8

make ARCH=arm64 CROSS\_COMPILE=aarch64-linux-gnu- bcm2711\_defconfig

1. 到<https://www.kernel.org/pub/linux/kernel/projects/rt/> 搜尋對應版本的real time patch

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 字型, 代數 的圖片

自動產生的描述找到4.19

點older資料夾進去找4.19.127的最新patch



1. 接著根據講義，下載補丁並解壓縮傳入patch utility

Wget <https://www.kernel.org/pub/linux/kernel/projects/rt/4.19/older/patch-4.19.127-rt55.patch.gz>

gunzip patch-4.19.127-rt55.patch.gz

cat patch-4.19.127-rt55.patch | patch -p1

1. 接著重做講義step5~9編譯有補丁的kernel

make ARCH=arm64 CROSS\_COMPILE=aarch64-linux-gnu- Image modules dtbs

此時應該會跳出選項，要選擇最下面fully preemption kernel(RT)

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述選好後繼續編譯

之後就接續將編好的kernel image安裝入SD卡

1. 將SD卡放入RPi，可順利開機了，用指令確認是否有安裝real time patch 成功

uname -r

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述uname -a

可看出是rt版本，patch成功!