

Robotic System များတည်ဆောက်မှုအတွက်

## Raspberry Pi အား C++ ဖြင့် အသုံးပြုခြင်း

ပထမအကြိမ်



[raspberrypi.org](https://raspberrypi.org)

ကိုအေး

## Raspberry Pi အား C++ ဖြင့် အသုံးပြုခြင်း

စာရေးသူ - ရန်နိုင်အေး @ ကိုအေး

Cool Emerald: <http://coolemerald.blogspot.com> မှ free ဖွန့်သည်။

တည်းဖြတ်မှု - ပထမအကြိမ်

ဉာဏ် ၂၀၁၈

Typesetting အတွက် X<sup>E</sup>T<sub>E</sub>X နှင့် Pyidaungsu ဖောင် ကို အသုံးပြု ထားသည်။

ဤ စာအုပ်၏ လိုင်စင် မှာ CC-BY-4.0 (<https://creativecommons.org/licenses/by/4.0/>) ဖြစ်  
သဖြင့် လွတ်လပ်စွာ ကူးယူ၊ ဖြန်ပေါ် ပြပြင်၊ အသုံးပြု နိုင်သည်။ ထို အတွက် သင့်တင့် လျောက်ပတ်  
သော အသိ အမှတ် ပြုမှု credit နှင့် link ဖော်ပြရန် လိုသည်။

နောက်ဆုံး တည်းဖြတ်မှု - <https://yan9a.github.io/rpi/rpi.pdf>

စာအုပ်၏ source code များ - <https://github.com/yan9a/rpi>

# စာရေးသူ၏ကိုယ်ရေးအကျဉ်း



ဒေါက်တာ ရန်နိုင်အေးသည် Mandaly Technological University (MTU) မှ Electronics အထူးပြုဖြင့် B.E. ဘွဲ့ ကို ၂၀၀၂ ခုနှစ် တွင် ရရှိခဲ့ပြီး၊ စက်ဘူးနိုင်ငံ Nanyang Technological University (NTU) မှ Computer Control and Automation ဘာသာရပ်ဖြင့် M.Sc. ဘွဲ့၊ Robotics နည်းပညာပိုင်းဖြင့် Ph.D. ဘွဲ့ တိုကို ၂၀၀၅ ခုနှစ် နှင့် ၂၀၁၆ ခုနှစ် တို့တွင် ရရှိ ခဲ့သည်။ NTU ရှိ Robotic Research Centreတွင် ၂၀၁၀ မှ စ၍ ၂၀၁၆ ထိ ၇ နှစ်ကြား နည်းပညာပိုင်းဆိုင်ရာ သုတေသနများ လုပ်ကိုင် ခဲ့သည်။ ယခု အခါ Sun Singapore System Pte. Ltd. တွင် Senior R&D Engineer အဖြစ် သုတေသန လုပ်ငန်း များ ဆောင်ရွက် လျက် ရှိသည်။

# မာတိကာ

စာရင်းသူ၏ကိုယ်ရေးအကျဉ်း	i
မာတိကာ	ii
<b>၁ နိတ်ဆက်</b>	၁
၁.၁ နှိပ်းယွဲမှုများ . . . . .	၁
၁.၁.၁ Arduino . . . . .	၁
၁.၁.၂ BeagleBoards . . . . .	၂
၁.၁.၃ Odroid-XU4 . . . . .	၄
၁.၁.၄ Orange Pi One . . . . .	၇
၁.၁.၅ Omega2+ . . . . .	၈
၁.၁.၆ သုံးသပ်ချက် . . . . .	၉
၁.၂ Raspberian ကိုတပ်ဆင်ခြင်း . . . . .	၁၀
၁.၃ Backup နှင့် Restore ပြည်ပခြင်း . . . . .	၁၂
၁.၃.၁ Linux . . . . .	၁၃
၁.၃.၂ Windows . . . . .	၁၄
၁.၃.၃ Raspberry Pi ပေါ်တွင်တိုက်ရှိက်ကူးခြင်း . . . . .	၁၅
၁.၄ Filesystem ကို ချွဲခြင်း . . . . .	၁၆
<b>၂ အခြေခံလုပ်ဆောင်မှုများ</b>	၁၉
၂.၁ Network ပြင်ဆင်ခြင်း . . . . .	၁၉
၂.၁.၁ Network ဆက်သွယ်မှုကိုစစ်ခြင်း . . . . .	၂၁

J.J VNC ဖြင့်အသုံးပြုခြင်း . . . . .	JJ
J.J.၁ VNC Server . . . . .	JJ
J.J.၂ VNC Client . . . . .	J၄
J.၃ SSH ဆက်သွယ်အသုံးပြုခြင်း . . . . .	J၅
J.၃.၁ Password ကို ပြောင်းခြင်း . . . . .	J၇
J.၃.၂ Password ကိုဖြုတ်ခြင်း . . . . .	J၈
J.၃.၃ User အသစ်ဖန်တီးခြင်း . . . . .	J၈
J.၃.၄ Sudoers . . . . .	J၉
J.၃.၅ User ဖျက်ခြင်း . . . . .	၃၀
J.၄ Internet ကို မျှဝေယူခြင်း . . . . .	၃၀
J.၄.၁ Windows Host PC တွင် ပြင်ဆင်ခြင်း . . . . .	၃၀
J.၄.၂ Raspberry Pi တွင် ပြင်ဆင်ခြင်း . . . . .	၃၂
J.၄.၃ Linux Host PC တွင် ပြင်ဆင်ခြင်း . . . . .	၃၃
J.၅ ဖိုင်များပေးပို့ရယူခြင်း . . . . .	၃၅
J.၅.၁ VNC . . . . .	၃၅
J.၅.၂ SFTP . . . . .	၃၇
J.၆ Sources list ကိုပြင်ဆင်ခြင်း . . . . .	၄၀
J.၇ C/C++ ပရိုဂရမ်ရေးသားခြင်း . . . . .	၄၀
J.၈ Linux Virtual Machine တစ်ခု တပ်ဆင်ခြင်း . . . . .	၄၃
J.၉ Cross Compilation Tool Chain . . . . .	၄၇
J.၁၀ QEMU - Quick EMULATOR . . . . .	၅၀
J.၁၁ Code::Blocks IDE တပ်ဆင်ခြင်း . . . . .	၅၁
J.၁၁.၁ Code::Blocks Project နမူနာ . . . . .	၅၅
<b>၃ Input/Output များ</b>	<b>၆၁</b>
၃.၁ Digital IO . . . . .	၆၁
၃.၁.၁ C++ ဖြင့်သုံးခြင်း . . . . .	၆၄
၃.၁.၂ Light sensor ဖြင့်အလင်းရောင်ကိုအာရုံခံခြင်း . . . . .	၇၀
၃.၁.၃ Relay ဖြင့်ပါဝါအဖွင့်အပိတ်ထိန်းချုပ်ခြင်း . . . . .	၇၁

၃.၁၅ Light Activated Switch . . . . .	၇၂
၃.၂၂ Analog input များကိုဖတ်ခြင်း . . . . .	၇၃
၃.၂၃ Analog output ထုတ်ခြင်း . . . . .	၇၆
၃.၂၄ Wiring Pi ကို အသုံးပြုခြင်း . . . . .	၇၆
၃.၂၄.၁ Pins များ . . . . .	၇၆
၃.၂၄.၂ Digital IO များသုံးခြင်း . . . . .	၇၆
၃.၂၄.၃ Wiring Pi ကို source မှ build လုပ်၍ တပ်ဆင်ခြင်း . . . . .	၇၈
၃.၂၄.၄ Analog Output သုံးခြင်း . . . . .	၈၀
<b>၄ Serial Bus များ</b>	<b>၈၃</b>
၄.၁ I2C . . . . .	၈၃
၄.၁.၁ i2c-tools . . . . .	၈၅
၄.၁.၂ C++ ဖြင့်သုံးခြင်း . . . . .	၈၈
၄.၁.၃ Gyroscope ကိုအသုံးပြုခြင်း . . . . .	၉၉
၄.၂ SPI . . . . .	၉၅
၄.၂.၁ Shell . . . . .	၉၅
၄.၂.၂ C++ ဖြင့်သုံးခြင်း . . . . .	၉၈
၄.၂.၃ Gyroscope ကိုအသုံးပြုခြင်း . . . . .	၁၀၄
၄.၃ Analog အဝင်များကို ဖတ်ခြင်း . . . . .	၁၀၀
၄.၄ Analog အထွက် နှစ်မျိုး . . . . .	၁၀၈
၄.၄.၁ Digital to Analog Converter . . . . .	၁၀၉
၄.၄.၂ PWM အထွက်များ . . . . .	၁၂၂
<b>၅ စက်အမြင်</b>	<b>၁၃၀</b>
၅.၁ နောက်ခံအကြောင်းအရာ . . . . .	၁၃၀
၅.၂ ဖွဲ့စည်းပုံ . . . . .	၁၃၁
၅.၃ တပ်ဆင်ခြင်း . . . . .	၁၃၂
၅.၃.၁ ရှိထားရန်လိုအပ်သည့် packages များ . . . . .	၁၃၂
၅.၃.၂ apt ဖြင့်တပ်ဆင်ခြင်း . . . . .	၁၃၃
၅.၃.၃ Source မှ build လုပ်ခြင်း . . . . .	၁၃၄

၅.၃.၄	GCC ၊ CMake တို့ဖြင့် အသုံးပြုခြင်း . . . . .	၁၃၂
၅.၄	ကင်မရာကိုသုံးခြင်း . . . . .	၁၃၈
၅.၅	Real-time Face Detection . . . . .	၁၄၂
၅.၆	RaspiCam . . . . .	၁၄၆
၅.၆.၁	OpenCV နှင့် သုံးခြင်း . . . . .	၁၄၈
၅.၆.၂	CMake ဖြင့် Build လုပ်ခြင်း . . . . .	၁၅၀
၅.၇	Smart Surveillance ကင်မရာပြုလုပ်ခြင်း . . . . .	၁၅၀
<b>၆</b>	<b>GUI application များရေးသားခြင်း</b>	<b>၁၅၉</b>
၆.၁	wxWidgets ကိုတပ်ဆင်ခြင်း . . . . .	၁၆၀
၆.၂	wxWidgets ကို source မှ build လုပ်ခြင်း . . . . .	၁၆၂
၆.၃	OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း . . . . .	၁၆၃
၆.၃.၁	Code::Blocks . . . . .	၁၆၇
၆.၄	Autostart . . . . .	၁၇၃
<b>၇</b>	<b>Serial Port</b>	<b>၁၇၇</b>
၇.၁	UART . . . . .	၁၇၇
၇.၁.၁	Start Bit . . . . .	၁၇၇
၇.၁.၂	Baud Rate . . . . .	၁၇၈
၇.၁.၃	Data Bits . . . . .	၁၇၈
၇.၁.၄	Parity Bit . . . . .	၁၇၈
၇.၁.၅	Stop Bit . . . . .	၁၇၈
၇.၂	RS-232 . . . . .	၁၇၉
၇.၂.၁	Handshaking . . . . .	၁၈၀
၇.၃	Hardware များ . . . . .	၁၈၂
၇.၃.၁	Windows တွင်အသုံးပြုခြင်း . . . . .	၁၈၄
၇.၃.၂	Linux တွင်အသုံးပြုခြင်း . . . . .	၁၈၆
၇.၄	RPi ၏ serial port ကိုသုံးခြင်း . . . . .	၁၈၈
၇.၄.၁	Linux Console . . . . .	၁၉၉
၇.၄.၂	Serial Interface . . . . .	၁၉၁

၇.၅ Programming Serial Port . . . . .	၁၉၂
၇.၅.၁ Linux နှင့် Windows အတွက် Serial Class . . . . .	၁၉၃
၇.၅.၂ Console . . . . .	၁၉၄
၇.၅.၃ GUI . . . . .	၁၉၅
၇.၆ Byte Stuffing . . . . .	၂၀၃
<b>၈ Database</b>	<b>၂၀၆</b>
၈.၁ တပ်ဆင်ခြင်း . . . . .	၂၀၇
၈.၁.၁ Major Release Version ကို ရွေးချယ်ခြင်း . . . . .	၂၁၀
၈.၂ Source မှ တပ်ဆင်ခြင်း . . . . .	၂၁၀
၈.၂.၁ လိုအပ်သည့်အရာများ . . . . .	၂၁၀
၈.၂.၂ Preconfiguration setup . . . . .	၂၁၁
၈.၂.၃ Installation . . . . .	၂၁၂
၈.၂.၄ Postinstallation setup . . . . .	၂၁၃
၈.၂.၅ Startup . . . . .	၂၁၄
၈.၃ Root Password ကို reset လုပ်ခြင်း . . . . .	၂၁၇
၈.၄ Configuration . . . . .	၂၁၈
၈.၅ MySQL server ကို စမ်းသပ်ခြင်း . . . . .	၂၂၀
၈.၅.၁ mysqladmin . . . . .	၂၂၀
၈.၅.၂ mysql . . . . .	၂၂၀
၈.၆ Installing MySQL Connector/C++ from source . . . . .	၂၂၂
၈.၆.၁ Build Tools . . . . .	၂၂၂
၈.၆.၂ ရယူခြင်း . . . . .	၂၂၂
၈.၆.၃ Configuring . . . . .	၂၂၃
၈.၆.၄ Building . . . . .	၂၂၄
၈.၆.၅ Installing . . . . .	၂၂၄
၈.၆.၆ Connector C++ Application များ build လုပ်ခြင်း . . . . .	၂၂၅
<b>၉ Internet of Things</b>	<b>၂၃၀</b>
၉.၁ UDP . . . . .	၂၃၀

၉.၂ TCP . . . . .	J၅၀
၉.၂.၁ TCP Server . . . . .	J၅၀
၉.၂.၂ TCP Client . . . . .	J၆၂
၉.၃ FTP Server တပ်ဆင်ခြင်း . . . . .	J၇၇
၉.၄ Web Server တပ်ဆင်ခြင်း . . . . .	J၇၇
၉.၄.၁ PHP ကို အသုံးပြုခြင်း . . . . .	J၇၈
၉.၅ Google Charts . . . . .	J၇၉
၉.၅.၁ Chart လိုင်ဘရီထည့်ခြင်း . . . . .	J၈၂
၉.၅.၂ ဒေတာများပြင်ဆင်ခြင်း . . . . .	J၈၃
၉.၅.၃ Chart ကို ဆွဲခြင်း . . . . .	J၈၃
၉.၆ D3.js . . . . .	J၈၄
၉.၇ Email ပိုခြင်း . . . . .	J၈၈
၉.၇.၁ ဖိုင်ကိုပိုခြင်း . . . . .	J၉၉
၉.၇.၂ C++ ဖြန့်ပိုခြင်း . . . . .	J၉၀
<b>၁၀ အချိန်</b>	<b>J၉၂</b>
၁၀.၁ Real Time Clock အသုံးပြုခြင်း . . . . .	J၉၂
၁၀.၁.၁ I2C ဆက်သွယ်မှု . . . . .	J၉၃
၁၀.၁.၂ I2C ဆက်သွယ်မှု . . . . .	J၉၄
၁၀.၁.၃ RTC ကို setup လုပ်ခြင်း . . . . .	J၉၆
၁၀.၂ NTP Server . . . . .	J၉၉
၁၀.၂.၁ Basic Time Commands . . . . .	၃၀၀
၁၀.၂.၂ ntpd ပြောင်းသုံးခြင်း . . . . .	၃၀၂
၁၀.၂.၃ ntpd ကို ပုံစံသွင်းခြင်း . . . . .	၃၀၃
၁၀.၃ NTP Client . . . . .	၃၀၅
၁၀.၃.၁ Linux . . . . .	၃၀၅
၁၀.၃.၂ Windows . . . . .	၃၀၈
၁၀.၄ Watchdog . . . . .	၃၀၈
၁၀.၅ Crontab ကိုသုံးခြင်း . . . . .	၃၁၄

၁၀.၅.၁ Crontab တွင်လုပ်ငန်းသတ်မှတ်ခြင်း . . . . .	၃၃၅
<b>၁၁ Sensors</b>	<b>၃၁၈</b>
၁၁.၁ Temperature sensor ဖြင့်အပူချိန်ကိုအာရုံခြင်း . . . . .	၃၁၈
၁၁.၂ Accelerometer သုံးခြင်း . . . . .	၃၂၂
၁၁.၂.၁ SPI ဖြင့်သုံးခြင်း . . . . .	၃၂၂
၁၁.၂.၂ I2C ဖြင့်သုံးခြင်း . . . . .	၃၂၆
၁၁.၃ Energy Monitor ပြုလုပ်ခြင်း . . . . .	၃၂၀
၁၁.၃.၁ Current တိုင်းတာရန်ပြင်ဆင်ခြင်း . . . . .	၃၂၀
၁၁.၃.၂ Bias ပေးခြင်း . . . . .	၃၂၂
၁၁.၃.၃ Power တိုင်းတာခြင်း . . . . .	၃၂၂
၁၁.၃.၄ Signal Processing . . . . .	၃၂၆
၁၁.၃.၅ Calibration . . . . .	၃၂၇
၁၁.၃.၆ Schematic circuit . . . . .	၃၂၉
၁၁.၃.၇ C++ ဖြင့်သုံးခြင်း . . . . .	၃၆၀
၁၁.၃.၈ Web Interface . . . . .	၃၆၆
<b>၁၂ Actuators</b>	<b>၃၆၉</b>
၁၂.၁ Servo motor ကိုထိန်းချုပ်ခြင်း . . . . .	၃၆၉
၁၂.၂ DC motor များသုံးခြင်း . . . . .	၃၇၆
၁၂.၂.၁ C++ ဖြင့်သုံးခြင်း . . . . .	၃၇၈
၁၂.၃ Stepper motor များသုံးခြင်း . . . . .	၃၆၀
၁၂.၃.၁ Wave drive . . . . .	၃၆၄
၁၂.၃.၂ Full step drive . . . . .	၃၆၄
၁၂.၃.၃ Half step drive . . . . .	၃၆၆
၁၂.၃.၄ C++ ဖြင့်သုံးခြင်း . . . . .	၃၆၆
၁၂.၃.၅ Stepper Motor Driver များသုံးခြင်း . . . . .	၃၇၃
<b>A Code Listing</b>	<b>၃၇၈</b>
၁၁.၁ Minimal wxWidgets sample . . . . .	၃၇၈

၁.၂	Alpha channel ပါရီသည့် ပုံရိပ် များကို အသုံးပြုသည့် နမူနာ minimal.cpp . . . . .	၃၈၃
၁.၃	Serial.h . . . . .	၃၈၅
၁.၄	frame.h . . . . .	၄၀၃
B	စကားလုံးဖွင့်ဆိုချက်များ	၄၀၉

## အခန်း ၁

# မိတ်ဆက်

Raspberry Pi ([raspberrypi.org](http://raspberrypi.org)) က ခရက်ဒစ် ကုဒ် အရွယ် သေးငယ်တဲ့ ဂွန်ပျူးတာ ဘုတ် (single board computer) လေး တစ်ခု ဖြစ်ပြီး Computer Science နဲ့ ပတ်သက်တဲ့ ပညာရေး တွေ အတွက် အထောက် အကူ အနေ နဲ့ရော၊ ဝါသနာ ရှင်တွေ၊ developer တွေ အလွယ်တကူ အသုံးပြုဖို့ အတွက် ရော ရည်ရွယ် ဒီဇိုင်း လုပ်ပြီး ဈေး သက်သက် သာသာ ထုတ်လုပ် ထားတာ ဖြစ်ပါ တယ်။

Raspberry Pi တွေကို ထုတ်လုပ် လာတာ မျိုးဆက် သုံးခု မြောက် ရှုံးပြု ဖြစ်ပြီး၊ ARM CPU နဲ့ on-chip GPU တွေ ပါဝင် တဲ့ Broadcom SoC တွေကို အသုံးပြု ထား ပါတယ်။ Raspberry Pi အတွက် operating system အနေနဲ့ Debian based Linux distribution တစ်ခု ဖြစ်တဲ့ Raspbian ကို အသုံးပြု နိုင်ပြီး၊ တစ်ခြား အသုံးပြု နိုင်တဲ့ third party operating system တွေကို လည်း သူရဲ့ [Downloads](#) ဝက်ဘ် စာမျက်နှာ မှာ တွေ့နိုင် ပါတယ်။

## ၁.၁ နှိုင်းယူဉ်မှုများ

### ၁.၁.၁ Arduino

Raspberry Pi နဲ့ Arduino ဘာကွာလဲ? လို့ မေးရင် တော့ သူတို့ နှစ်ခု ရဲ့ ရည်ရွယ်ချက် က မတူဘူး လို့ ဆိုရ ပါမယ်။

Arduino ([arduino.cc](http://arduino.cc)) က microcontroller ကို အခြေခံ တာ ဖြစ်ပြီး electronic ဆားကစ် ပစ္စည်း တွေကို လွယ်လွယ် ကူကူ တိုက်ရိုက် ထိန်းကျောင်း ဖို့ သင့်တော် ပါတယ်။ တွက်ချက် ဖို့ power အများကြီး မလို့ operating system တွေ မပါပဲ ပါဝါ ဖွင့်လိုက် တာနဲ့ ချက်ချင်း တိုက်ရိုက် အသုံး

## ပြနိုင် ပါတယ်။

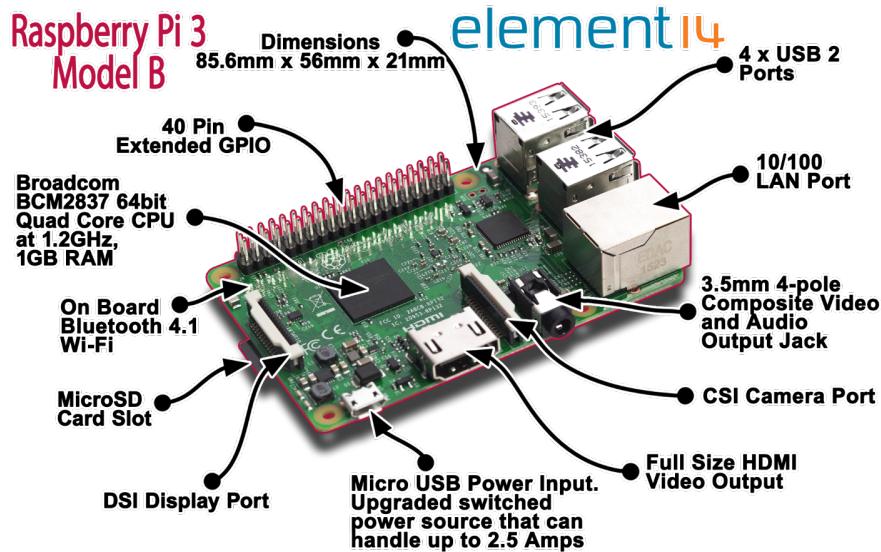
Raspberry Pi ကတေသာ တွက်ချက်မှု အများကြီး ပိုအားကောင်း တဲ့ microprocessor ကို သုံးပြီး ပိုမို ရှုပ်ထွေး အဆင့် မြင့်တဲ့ image processing လိုမျိုး တွေကို သုံးတဲ့ embedded ပရောဂျက် တွေ အတွက် သင့်တော် ပါတယ် [Dah15]။ Operating system အမျိုးမျိုး တပ်ဆင် နိုင်ပြီး Debian Linux ကို အခြေခံ ထားတဲ့ Raspbian ကို Raspberry Pi Foundation က တရားဝင် ထောက်ပံ့ ပေး ပါတယ်။

## ၁.၁.၂ BeagleBoards

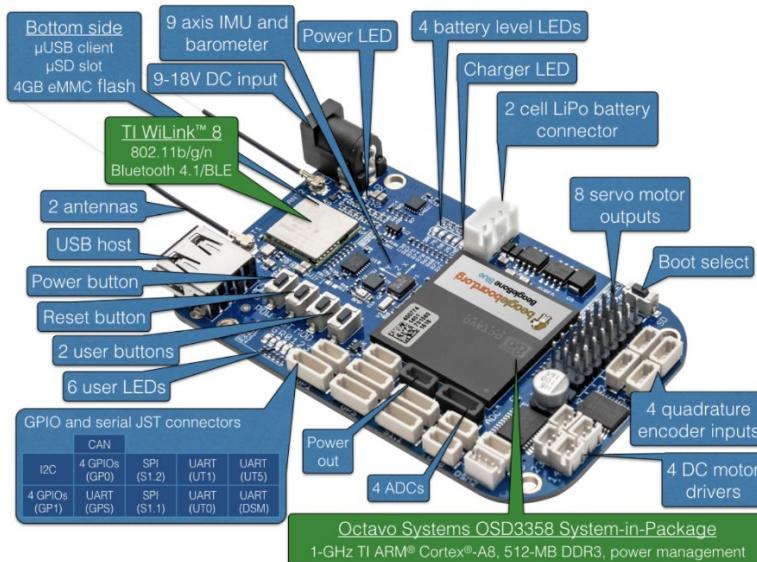
Single Board Computer တွေကို လူ အများ ကြား အလွယ် တကူ စွေးပေါ်ပေါ် နဲ့ သုံးဖို့ ပထမ ဆုံး အောင်မြင်စွာ ဖန်တီး ထုတ်လုပ်နိုင် ခဲ့တဲ့ Raspberry Pi က တစ်ခြား BeagleBone Black တို့လို single board computer တွေထက် အရင် ပေါ် တာပါ။ စွေး အနေနဲ့ လည်း Raspberry Pi က တော်တော် သက်သာ ပါတယ်။

ပထမ ဆုံး box ကို ဖွင့်ပြီး စ အသုံးပြု ရတဲ့ အခါ မှာတေသာ အခါ BeagleBone Black က နဲ့ ပို အဆင်ပြေ ပါတယ် [Leo14]။ Raspberry Pi က USB ကြိုး တခါထဲ ပါမလာ တဲ့ အပြင်၊ သူကို စတင် လုပ်ဆောင်ဖို့ μSD Card လိုပြီး၊ Operating System ကို အဲဒီထဲမှာ setup လုပ်ဖို့ လို ပါသေးတယ်။ BeagleBone Black စတဲ့ Beagleboards တွေ ကတေသာ eMMC ပါပြီး၊ ပါလာတဲ့ USB ကြိုးကို ကွန်ပျူးတာနဲ့ ဆက်လိုက် တာနဲ့ စပြီး သုံးလို ရပါပြီ။ တခြား အီလက် ထရောနစ် ပစ္စည်း တွေနဲ့ ဆက်သွယ် အသုံးပြု ဖို့ interface တွေခြင်း ယူဉ်ရင်လည်း BeagleBoards တွေက အထောက် အပံ့ ပိုများ တာကို တွေ့ရ ပါတယ် (ပုံ ၁.၁ နှင့် ယေား ၁.၁)။

အင်တာနက် နဲ့ ချိတ်ဆက် အသုံးပြု ရတဲ့ IoT ပရောဂျက် တွေ အတွက် တော့ နှစ်ခုလုံး က အဆင် ပြေကြ ပေမယ့် Multimedia နဲ့ ပတ်သက်တဲ့ လုပ်ဆောင်မှု တွေမှာတေသာ Raspberry Pi က ပိုကောင်း ပါတယ်။ Raspberry Pi မှာ HDMI output | camera port | ၃.၅ မမ audio jack စော့ တွေ ပါပြီး BeagleBone မှာ မပါ ပါဘူး။ Graphics processing တွေမှာလည်း Raspberry Pi က ပိုပြီး စွမ်းဆောင်ရည် မြင့်မား ပါတယ်။



(a) Raspberry Pi 3 Model B Technical Specifications



(b) BeagleBone Blue Technical Specifications

၁.၁: Raspberry Pi 3 Model B နဲ့ BeagleBone Blue နှင့်ယုံမှု။

ဧယား ၁.၁: Raspberry Pi 3 Model B နှင့် BeagleBone Blue တို့အား နိုင်းယူလိမ့်။

ပစ္စည်း	Raspberry Pi 3 Model B	BeagleBone Blue
Processor	Broadcom BCM2837 4 core, 64 bit, 1.2 GHz	Octavo Systems OSD3358 1 core, 32 bit, 1 GHz
RAM	1 GB LPDDR2	512 MB DDR3
Storage	$\mu$ SD Card Slot	4 GB on-board eMMC & $\mu$ SD Card Slot
Communication	Wifi, Bluetooth, Ethernet	Wifi, Bluetooth, CAN
USB	4 USB host,	1 USB hosts, 1 mini USB client
UART	1	5
GPIO	24 (shared on 40 pins header)	8
ADC	NA	4
Peripherals	Camera port, HDMI, audio jack	9 axis IMU, Barometer, 2 cell (2S) LiPo battery charger, Battery connector, 8 servo motor outputs, 4 quadrature encoder inputs, 4 DC motor drivers, 2 user buttons, 6 user LEDs

### ၁.၁.၃ Odroid-XU4

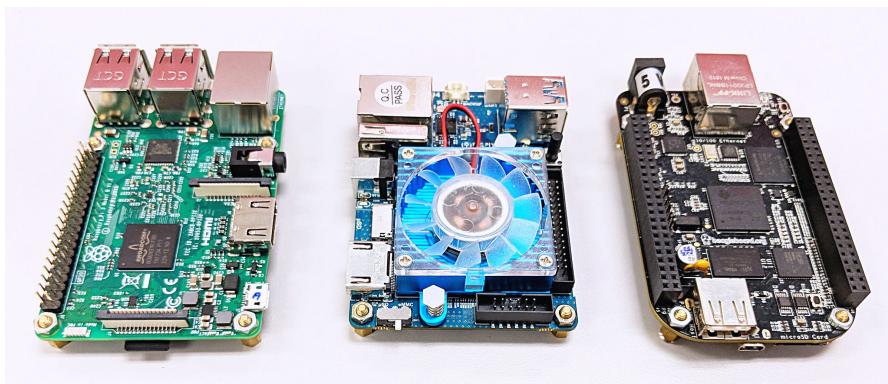
Odroid-XU4 က ကိုရီးယား ကုမ္ပဏီ hardkernel.com က ထုတ်ထဲ single board computer တစ်ခု ဖြစ်ပြီး လုပ်ဆောင် နိုင်မှု စွမ်းရည် တော်တော် မြင့်မား ပါတယ်။ Exynos5422 Cortex-A15 2Ghz နဲ့ Cortex-A7 Octa core CPU တွေ ပါရှိပြီး၊ LPDDR3 RAM က 2Gbyte ပါရှိ ပါတယ်။ Mali-T628 MP6 GPU လည်း ပါပြီး၊ storage အနေ နဲ့ လည်း SD card အပြင် ပိုမို မြန်ဆန် တဲ့ eMMC module ကို စိတ်ကြိုက် အချယ်အစား တပ်ဆင် အသုံးပြု နိုင် ပါတယ်။ Network interface အနေနဲ့ လည်း Gigabit Ethernet ပါရှိ တဲ့ အတွက် မြန်ဆန် တဲ့ ဒေတာ ပေးပို့ မှု ရရှိ နိုင် ပါတယ်။ HDMI interface နဲ့ USB 3.0 port နှစ်ခု၊ USB 2.0 port တစ်ခု လည်း ပါ ပါတယ်။ ဈေးက \$59 ဖို့လို့ ဈေးကြီး တဲ့ ပါပါတယ်။

Odroid အတွက် Operating systems တွေ အနေ နဲ့ ဆိုရင် လည်း Ubuntu Mate နဲ့ Android အပြင် အခြား third party OS အမျိုးမျိုး ကို သုံးလို့ ရပါတယ်။ Odroid-XU4 ကို သူရဲ့ Ubuntu Mate နဲ့ သုံးကြည့်တဲ့ အခါ desktop ကွန်ပျူးတာ ကို သုံးနေ ရ သလို လွယ်ကူ အဆင်ပြေ တာကို တွေ့ရ ပါတယ်။

ကြီးမား တဲ့ heat sink နဲ့ cooling fan တစ်ခါတဲ့ ပါလာ တာက လည်း ကြမ်းတမ်း တဲ့ ပတ်ဝန်းကျင် တွေမှာ များပြား တဲ့ တွက်ချက် မှုတွေ အတွက် သုံးတဲ့ အခါ ခံနိုင် ရည် မြင့်မား စေ ပါတယ်။ လက်တွေ

ပြင်ပ က outdoor car parking တွေမှာ video processing တွေ အများ အပြား လုပ်ဆောင် တဲ့ application တစ်ခု အတွက် 24/7 ယူလွှာသုံး ကြည့်တဲ့ အခါ Raspberry Pi 3B က crash မကြာ ခက်ဖြစ်ပြီး မလုပ် ဆောင်နိုင် ပေမယ့် Odroid-XU4 က ဘာမှ ပြသနာ မရှိ ပဲ လုပ်ဆောင် နှိုင်တာ ကို တွေ့ရ ပါတယ်။ ဒါပေမယ့် အဲဒီလို heavy သုံး စရာ မလိုတဲ့ သာမန် လုပ်ငန်း တွေ အတွက်၊ ဈေးနှုန်း၊ community support စော့ တွေ အားလုံး ခြုံကြည့် တဲ့ အခါ မှာတော့ Raspberry Pi က ပိုပြီး လူကြိုက် များ တာကို တွေ့ရ ပါတယ်။

ပုံ ၁.၂ မှာ Raspberry Pi 3B ၊ Odroid-XU4 နဲ့ BeagleBone Black တို့၏ အချင်း အစား တွေကို နှိုင်းယူလို ဖော်ပြု ထား ပါတယ်။

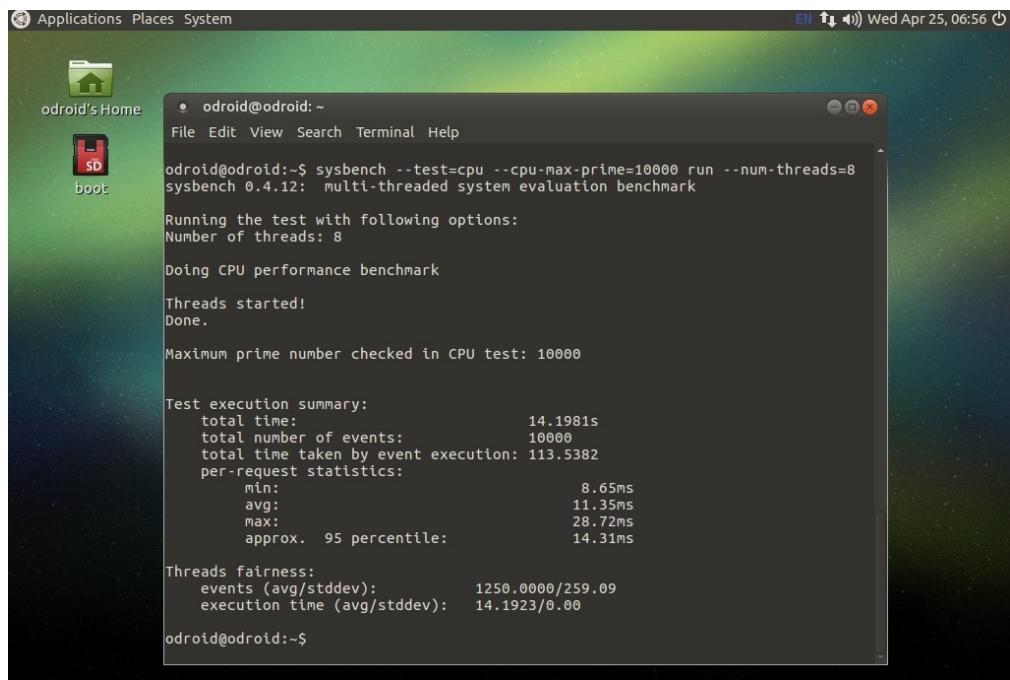


ပုံ ၁.၂: Raspberry Pi 3B ၊ Odroid-XU4 နဲ့ BeagleBone Black တို့၏ အချင်း အစား နှိုင်းယူလို များ

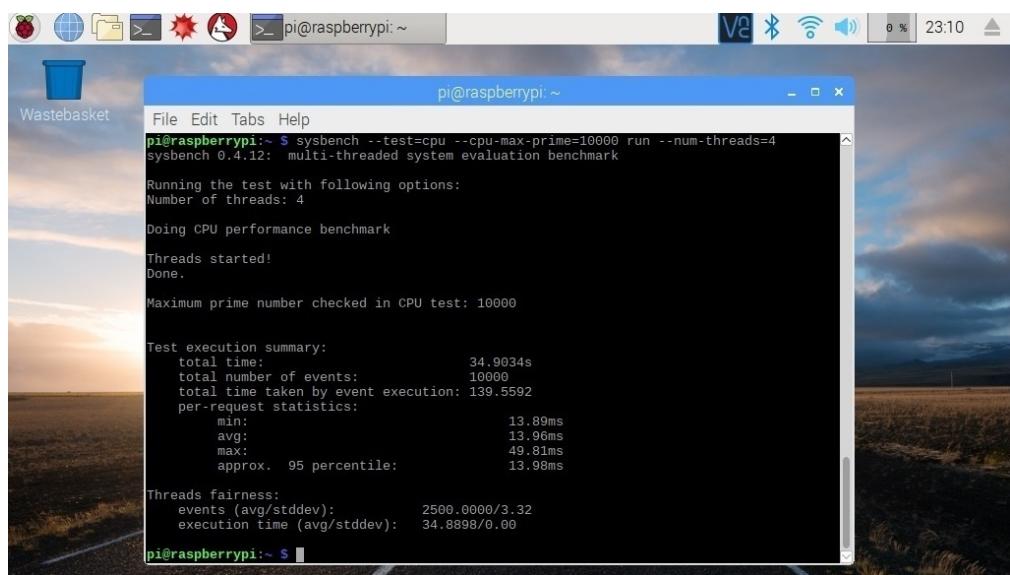
Odroid-XU4 ၊ Raspberry Pi 3B နဲ့ BeagleBone Black တို့၏ CPU လုပ်ဆောင်မှု တွေကို နှိုင်းယူလို ကြည့်ဖို့ အတွက် sysbench ဆိုတဲ့ application ကို အောက်ပါ အတိုင်း သုံးကြည့် ပါမယ်။

```
$ sudo apt install sysbench
$ sysbench --test=cpu --cpu-max-prime=10000 run --num-threads=8
```

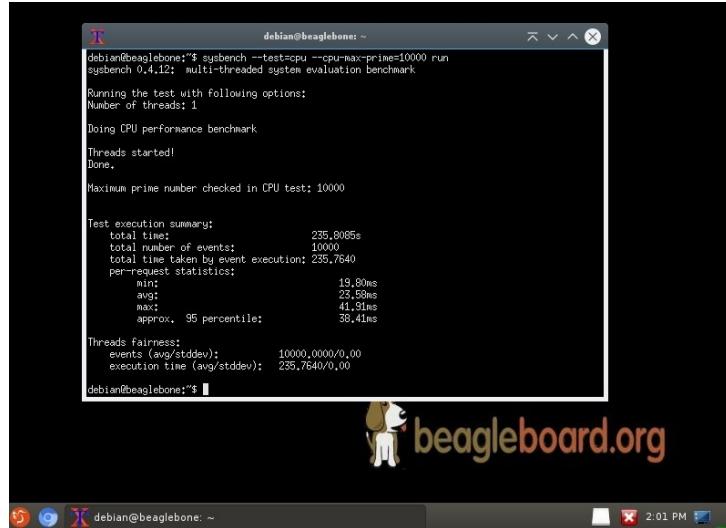
အကြွေးဆုံး ဂဏန်း ၁၀၀၀၀ အထိ ရှိတဲ့ prime number တွေကို တွေကိုကြည့်တဲ့ လုပ်ဆောင်မှု ကို ရှိသမှု CPU core တွေ သုံးပြီး တွေက်ကြည့် တဲ့ အခါ Odroid-XU4 က 14 s နဲ့ အမြန်ဆုံး (ပုံ ၁.၃)၊ Raspberry Pi 3B က 35 s ကြာပြီး (ပုံ ၁.၄)၊ BeagleBone Black က 236 s နဲ့ အနေးဆုံး (ပုံ ၁.၅) ဖြစ်တာ ကို တွေ့ရ ပါတယ်။



ပုံ ၁.၃: Odroid-XU4 ဒါ sysbench ရလဒ်။



ပုံ ၁.၄: Raspberry Pi 3B ဒါ sysbench ရလဒ်။



ပုံ ၁.၅: BeagleBone Black ၏ sysbench ရလဒ်။

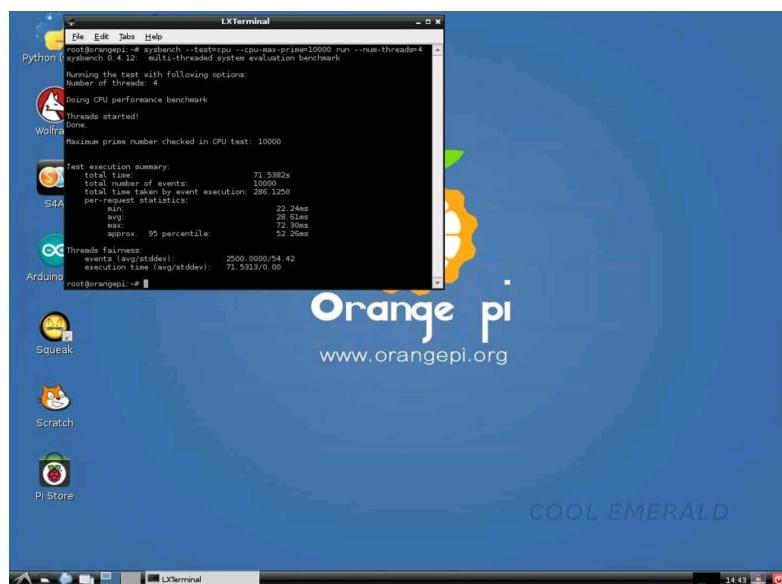
### ၁.၁.၄ Orange Pi One

ကုန်ကျ စရိတ် ကို ဦးစားပေး လိုတဲ့ လုပ်ငန်း တွေ အတွက် ဆိုရင်တော့ Orange Pi One ဆိုတဲ့ \$10 လောက်ပဲ ရှိတဲ့ Single Board Computer လေးက တော်တော် သင့်တော် ပါတယ်။ ပုံ ၁.၆ မှာ ပြထား သလို အချင်း အစား က Raspberry Pi ထက် သေးပြီး၊ လုပ်ဆောင် နိုင်မှ စွမ်းရည် ကလည်း မဆိုးတာ ကို တွေ့ရ ပါတယ်။



ပုံ ၁.၆: အချင်း အစား ကျေးသက်သာသော Orange Pi One ။

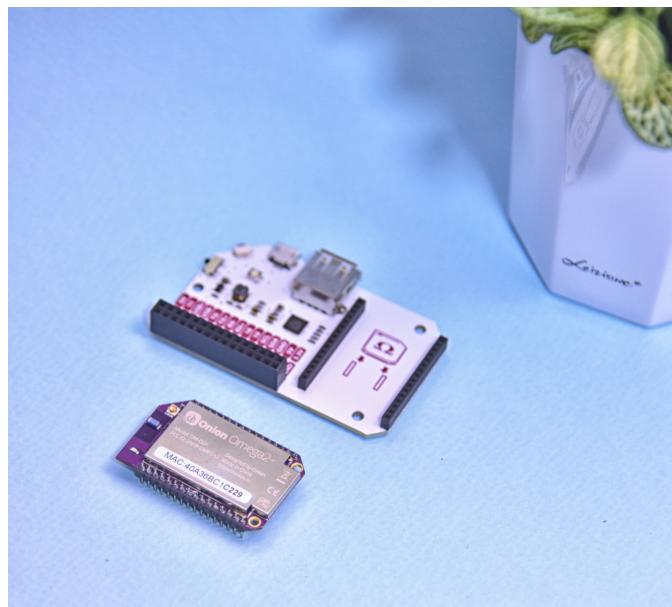
ခုနက အကြီးဆုံး ဂဏေန်း ၁၀၀၀၀ အထိ ရှိတဲ့ prime number တွေကို တွက်ကြည့်တဲ့ sysbench လုပ်ဆောင်မှု အတွက် 72 s ပဲ ကြာ တာကို တွေ့ရ ပါတယ် (ပုံ ၁.၇)။ မှာယူ တဲ့ အခါမှာ လည်း AliExpress က ဖွန်လိုင်း ဝယ် တဲ့ အခါ ၁၀ ရက် အတွင်း ရောက်လာ ပါတယ်။ Debian စတဲ့ OS တော်တော် များများ လည်း ရွှေးချယ် စရာ ရှိ တာကို တွေ့ရ မှာပါ။ ဒါပေမယ့် တစ်ခု ခဲ့ အတွက် သိမြှို့လိုလာရင် community support က မရှိ သလောက် ပဲမို့ အဲဒါ က Raspberry Pi နဲ့ ယွင်းတဲ့ အခါ ကြီးမား တဲ့ အားနည်း ချက် လို့ ပြောစရာ ပါ။



ပုံ ၁.၇: Orange Pi One ၏ sysbench ရလဒ်။

## ၁.၁.၅ Omega2+

Omega2+ ကတော့ တော်တော် လေး သေးငယ် ပြီး \$13 ပဲ ရှိလို့ ဈေးလည်း သက်သာ တဲ့ SBC လေးပါ။ IoT အတွက် ဦးတည် ထားပြီး၊ တွေ့ခြား hardware expansions တွေလည်း အများ ကြီးရှိ ပါတယ်။ တော်တော်လေး ကောင်းမွန် တဲ့ website support လည်း ရှိပါတယ်။ MT7688 580MHz MIPS CPU သုံးထားပြီး 128MB Memory ပါရှိသလို့ Wifi, Ethernet တွေ ပါရှိ တာကို တွေ့ရ ပါတယ်။ GPIO 12 ခု ပါရှိပြီး၊ SPI, I2C, UART စတာ တွေကို လည်း သုံးနိုင် ပါတယ်။ အရမ်း သေးငယ် တဲ့ အတွက်၊ စ စမ်းသပ်ဖို့ အတွက် ဆိုရင်၊ ပါဝါ ပေးတာ ကစ အဆင် ပြေအောင် သူအတွက် Expansion Dock ကိုပါ ဝယ်ဖို့ လို ပါလိမ့် မယ်။



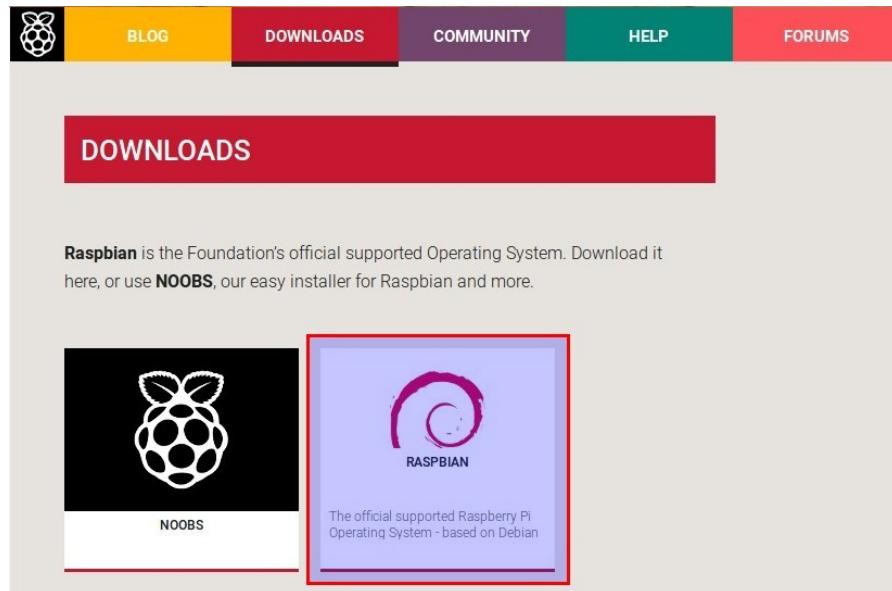
ပုံ ၁.၈: Omega2+ နှင့် သူအတွက် Expansion Dock ။

## ၁.၁.၆ သုံးသပ်ချက်

နောက်ဆုံး အနေနဲ့ သုံးသပ် ရမယ် ဆိုရင်တော့ hardware တိုက်ရိုက် သုံးတာမျိုး၊ real-time အတိအကျ လုပ် ဆောင်မှု မျိုး တွေ အတွက် Arduino တိုင်း microcontroller တွေ တိုက်ရိုက် သုံးတာ က ကောင်းမွန် အဆင် ပြေ ပါတယ်။ Raspberry Pi ကတော့ ယေဘုယျ လုပ်ဆောင် မှု အများစုံ အတွက် အဆင်ပြေ ပြီး ဈေး နဲ့ ရနိုင်တဲ့ တွက်ချက်မှု စွမ်းအား ကောင်းသလို community support တွေ ကောင်းပါ တယ်။ Robotic လုပ်ငန်း တွေလို မျိုး hardware တွေ အများကြီး ထိန်းချုပ် ဖို့ လိုတာ၊ hardware interface တွေ အများကြီး နဲ့ ရှုပ်ထွေး တဲ့ တွက်ချက် နှင့်မှု တွေလည်း လိုတဲ့ နေရာ တွေ မှာတော့ BeagleBoards တွေက အဆင် အပြေ ဆုံးပါ။ ကွန်ပျူးတာ စွမ်းဆောင် နှင့်မှု များများ ပို လိုအပ် ပြီး industry တွေမှာ heavy သုံးမယ် ဆိုရင် Odroid-XU4 ကလည်း သင့်တော် ပါတယ်။ ဈေးနှုန်း သက်သာ ဖို့ အရေးကြီး ပြီး ရှုပ်ထွေး အဆင့်မြင့် တဲ့ လုပ်ဆောင်မှု တွေလည်း လိုအပ် ရင် တော့ Orange Pi One နဲ့ သင့်တော်မှု ရှိပါ တယ်။ အရွယ် အစား compact ဖြစ်ဖို့ လိုတယ်၊ ချိတ်ဆက် မှု တွေ အများကြီး သုံးချင်တယ် ဆိုရင် Omega2+ ကိုလည်း သုံးနိုင် ပါတယ်။ အားလုံးက Debian Linux ကို အခြေခံ တဲ့ OS တွေ သုံးလို ရတာမို့ ဒါ စာအုပ် မှာ ဈေးနွေး ထားတဲ့ အကြောင်း အရာ တွေက အဲဒီ SBC တွေ အကုန် လုံးအတွက် အသုံး တည့် နိုင် ပါတယ်။

## ၁.J Raspberian ကိုတပ်ဆင်ခြင်း

Raspberry Pi အတွက် SD Card ကို နောက်ဆုံးပေါ် software image နဲ့ တပ်ဆင် ဖို့ အတွက် official Raspberry Pi Downloads page ([raspberrypi.org/download/](http://raspberrypi.org/download/)) ကို သွားပြီး ပုံ ၁.၉ မှာ ပြထားသလို Raspbian ပေါ်မှာ ကလစ် နှိပ်ပြီး '2017-09-07-raspbian-stretch.zip' အစ ရှိတဲ့ နောက်ဆုံး image ကို ယလိုက် ပါမယ်။



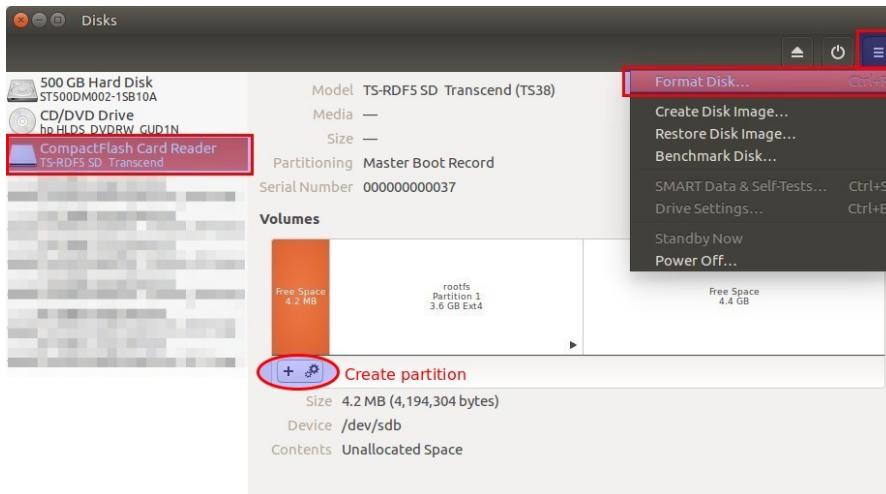
ပုံ ၁.၉: Getting Raspberian



ပုံ ၁.၁၀: Disks application

နောက် တစ်ခါ <https://etcher.io/> ကို သွားပြီး Etcher ကို ရယူ တပ်ဆင် လိုက်ပါမယ်

[ras17b]။ ပြီးတဲ့ အခါ  $\mu$ SD card ကို ကွန်ပျူတာ ရဲ့ Card Reader ဒါမှ မဟုတ် USB adapter သုံးပြီး တပ်ဆင် လိုက် ပါမယ်။ Ubuntu Linux ရဲ့ Disks ဆိုတဲ့ application (ပုံ ၁.၁၀) ကို သုံးပြီး card ကို format လုပ်နိုင် ပါတယ်။ Windows တို့ Mac တို့ ပေါ်မှာ တော့ SD Association's website က SD Memory Card Formatter ကို သုံးပြီး format လုပ် နိုင်ပါတယ်။



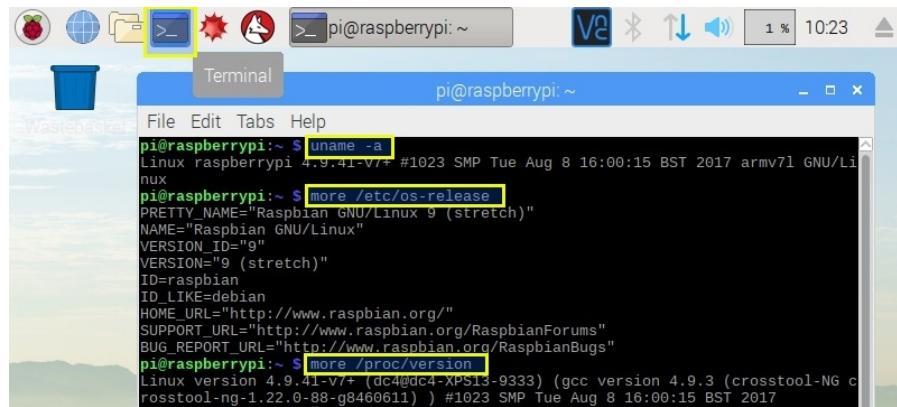
ပုံ ၁.၁၀: Format လုပ်ခြင်း နှင့် Partition create လုပ်ခြင်း။

Etcher application ကို ဖွင့်လိုက် တဲ့ အခါ  $\mu$ SD Card ကို အလို အလျောက် ရွေးချယ် ပြသ နေပါ လိမ့်မယ်။ လိုအပ် ရင် ရေးမယ့် ကဒ် ကို ပြောင်း သတ်မှတ် ပေးနိုင် ပါတယ်။ Source file အတွက် ခုန် က download လုပ် ထားတဲ့ '2017-09-07-raspbian-stretch.zip' ကို ရွေးပေး လိုက် ပါမယ်။ ပြီးတဲ့ အခါ ပုံ ၁.၁၂ မှာ ပြထားတဲ့ Flash ဆလုတ် ကို နှိမ်ပြီး  $\mu$ SD Card ပေါ်ကို software image ထည့်သွင်း နိုင် ပါတယ်။



ပုံ ၁.၁၂: Software image ကို  $\mu$ SD Card တွင် ထည့်သွင်းခြင်း။

Raspberry Pi မှာ ခုနက ရလာတဲ့ μSD Card ကို တပ်၊ monitor ထွေ၊ keyboard နဲ့ mouse ထွေ ချိတ်ဆက် ပြီး power ပေးလိုက် ရင် μSD Card ပေါ်က နောက်ဆုံး software ဟာရှင်းကို သုံးဖြီးစက်က တက် လာ ပါမယ်။ System information ထွေ ကြည့်ဖို့ အတွက် Task bar ပေါ်က Terminal ကို ဖွင့်ပြီး အောက်က command ထွေကို ရိုက်ထည့် ပြီး ကြည့်နိုင် ပါတယ် (စာရင်း ၁.၁ နှင့် ပုံ ၁.၁၃)။



ပုံ ၁.၁၃: System information များကို စစ်ခြင်း။

```

1 $ uname -a
2 $ more /etc/os-release
3 $ more /proc/version

```

စာရင်း ၁.၁: System အချက်အလက်များ ကြည့်ခြင်း။

## ၁.၃ Backup နှင့် Restore ပြုလုပ်ခြင်း

ကိုယ့် Raspberry Pi အတွက် ကိုယ်ဟာ ကိုယ် စိတ်တိုင်းကျ setup လုပ်ထားတဲ့ μSD Card ကို Image ဖိုင်ဗျား ယူပြီး backup လုပ် ထားတာ ကောင်းပါတယ်။ Backup လုပ် ဖို့ အတွက် Windows စက်တွေမှာ Win32 Disk Imager ကို သုံးလိုကြပြီး Linux နဲ့ Mac ထွေ အတွက် dd ဆိုတဲ့ command ကို သုံးနိုင် ပါတယ် [Tib17; ras17a]။

### ၁.၃.၁ Linux

နှမူနာ အနေနဲ့ Linux ပေါ်မှာ dd ကို သုံးပြီး backup လုပ်တဲ့ အကြောင်း ဖော်ပြ ပါမယ်။ စစ်ခြင်း μSD card ကို ကွန်ပျုံတာ မှာ မတပ်ခင်

```
$ df -h
```

ဆိုတဲ့ command ကို ရိုက်ထည့်ပြီး ရလာတဲ့ စာရင်း ကို μSD Card တပ်ပြီး အဲဒီ command ကို ပြန်သုံးတဲ့ အခါ ရလာတဲ့ စာရင်း နဲ့ နှိမ်းယူဉ် ကြည့် ပါမယ်။ ဥပမာ ကျွန်တော့ စက်မှာ /dev/sdb1 နဲ့ /dev/sdb2 ဆိုတဲ့ partition နှစ်ခု μSD Card အတွက် ပေါ်လာ တာကို တွေ့ရ ပါတယ် (ပုံ ၁.၁၄)။

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sdb1	42M	21M	21M	51%	/media/yan/boot
/dev/sdb2	15G	11G	3.0G	79%	/media/yan/b4ea8e46

ပုံ ၁.၁၄: SD Card ၏ path ကို စစ်ကြည့်ခြင်း။

Disk တစ်ခု လုံး ကူးချင် တာမူး အောက်မှာ ပြထားတဲ့ အတိုင်း dd ရဲ့ input အနေနဲ့ /dev/sdb လို့ ပေးလိုက် တဲ့ အခါ partition နှစ်ခု လုံးကို output အနေနဲ့ ပေးထားတဲ့ rpid.img ဆိုတဲ့ ဖိုင် အနေနဲ့ backup လုပ်ပေးပါ လိမ့်မယ်။ Optional အနေနဲ့ block size ကို 1M စသည်ဖြင့် ထည့် သတ်မှတ် နိုင် ပါတယ် (ပုံ ၁.၁၅)။

```
$ sudo dd if=/dev/sdb of=/home/yan/rpid.img bs=1M
```

```
yan@hp:~$ sudo dd if=/dev/sdb of=/home/yan/rpid.img bs=1M
15193+1 records in
15193+1 records out
15931539456 bytes (16 GB, 15 GiB) copied, 191.997 s, 83.0 MB/s
```

ပုံ ၁.၁၅: dd ကို အသုံးပြုခြင်း။

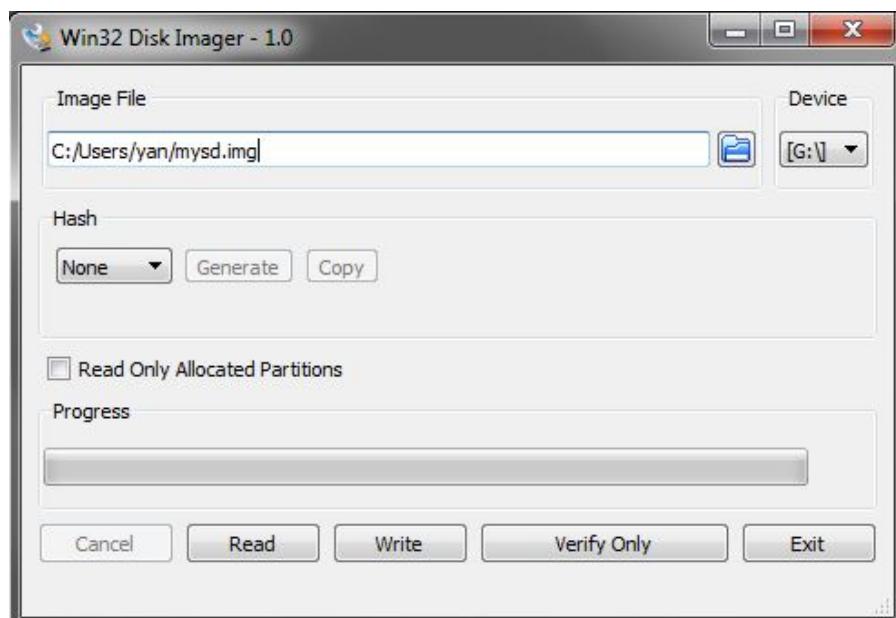
ရလာတဲ့ ဖိုင်ပေါ်မှာ ညာဖက် ကလစ် ကို နှိပ်ပြီး compress... ကို ရွေး၊ .xz နဲ့ သိမ်းရင် ပိုပြီး သေးငယ်တဲ့ ဖိုင်ကို ရနိုင် ပါတယ်။ သိမ်းထားတဲ့ image ဖိုင် ကို μSD Card ပေါ် ပြန်ထည့် ချင်ရင် အဲဒီ command ကို ပဲ input နဲ့ output ကို ပြောင်းပြီး ပြန်သုံး နိုင် ပါတယ်။ နောက် တစ်နည်း အနေနဲ့

အပိုင်း ၁.၂ က အတိုင်း Etcher ဆိုတဲ့ application ကို သုံးပြီး image ဖိုင်ကို ကူးရေး နှင့် ပါတယ်။

Etcher application ကို ဖွင့်လိုက် တဲ့ အခါ ဗုဒ္ဓာ ကျော် ရွှေးချယ် ပြုသ နေမှာ ဖြစ်ပြီး၊ လိုအပ်ရင် ရေးမယ့် ကုဒ် ကို ပြောင်း သတ်မှတ် ပေးနိုင် ပါတယ်။ Source file အတွက် ခုန် က ဖန်တီး ထားတဲ့ rpid.img ကို ကို ရွှေးပေး လိုက် ပါမယ်။ ပြီးတဲ့ Flash ခလုတ် ကို နှိပ်လိုက် ရင် ရှုက ပုံ ၁.၃ မှာ ပြထားတဲ့ နည်းတူ ဗုဒ္ဓာ ပေါ်ကို ကူးထည့် ပေး သွား မှာ ဖြစ် ပါတယ်။

## ၁.၃.၂ Windows

Windows စက်တွေ မှာတော့ Win32 Disk Imager ကို [sourceforge.net/projects/win32diskimager](http://sourceforge.net/projects/win32diskimager) ကနေ download လုပ်ပြီး တပ်ဆင် လိုက် ပါမယ်။ ပြီးတဲ့ အခါ သူကို ဖွင့်လိုက် ရင် ပုံ ၁.၁၆ မှာ ပြထား သလို တွေ့ရမှာ ဖြစ်ပါတယ်။

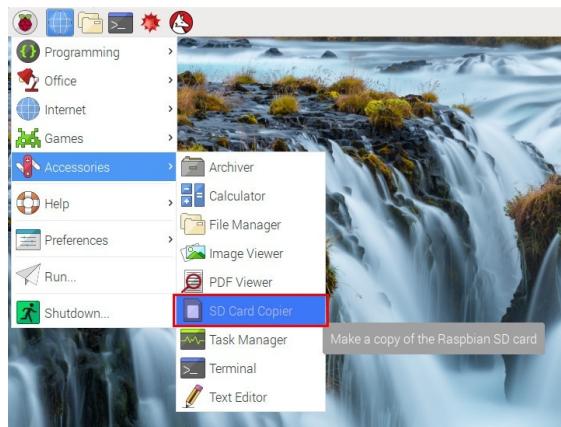


ပုံ ၁.၁၆: Win32 Disk Imager ကို အသုံးပြုခြင်း။

အဲဒီမှာ Read ခလုတ် ကို နှိပ်ရင် SD Card ကို သတ်မှတ် လိုက်တဲ့ image ဖိုင် အနေ နဲ့ ကူးယူ ပေးမှာ ဖြစ် ပါတယ်။ Write ခလုတ် ကို နှိပ်ရင် တော့ image ဖိုင်ကို SD Card ပေါ် ရေးပေး ပါလိမ့် မယ်။

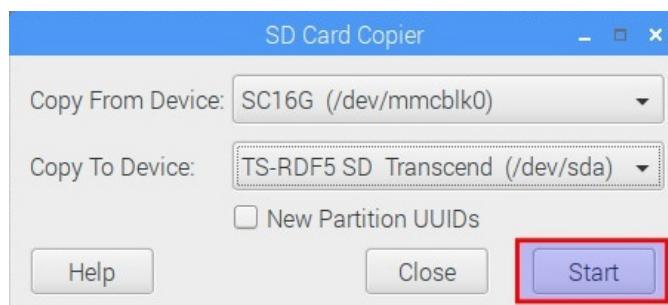
### ၁.၃.၃ Raspberry Pi ပေါ်တွင်တိုက်ရှိကူးခြင်း

SD card reader ကို Raspberry Pi ရဲ့ USB port မှာ တိုက်ရှိက် တင်ပြီး ကူးရင် လည်း ရပါတယ်။ SD card reader ကို တပ်ဖြီး တဲ့ အခါ ပုံ ၁.၁၇ မှာ ပြထား သလို Start Menu → Accessories → SD Card Copier ကို နိုပ်ပြီး ဖွင့်လိုက် ပါမယ်။



ပုံ ၁.၁၇: SD Card Copier ကို ဖွင့်ခြင်း။

ပြီးရင် Copy From Device ကို မူရင်း /dev/mmcblk0 နဲ့ Copy to Device မှာ အသစ် ပေါ်လာတဲ့ /dev/sda ကို ရွေးပေးပြီး Start လလှတ် ကို နိုပ် လိုက် ရင် လက်ရှိ ကိုယ့်စိတ်ကြိုက် setup လုပ်ထားတဲ့ Raspbian နဲ့ တထပ်ထည်း တူညီတဲ့ μSD ကော်ကို ကော်ပိ ကူးပေး သွားမှာ ဖြစ်ပါတယ်။



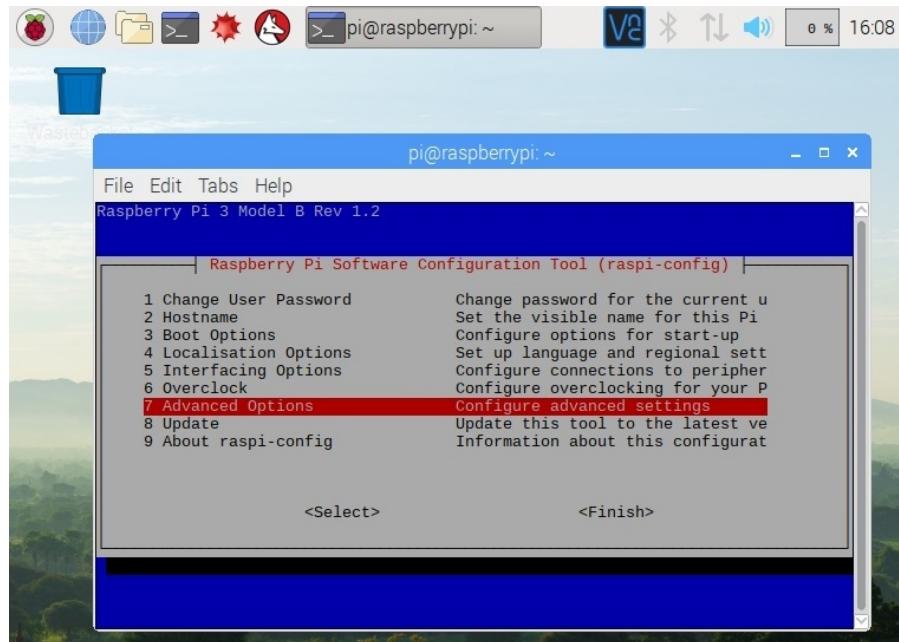
ပုံ ၁.၁၈: SD Card Copier ကို အသုံးပြုခြင်း။

## ၁.၄ Filesystem ကို ပျော်ခြင်း

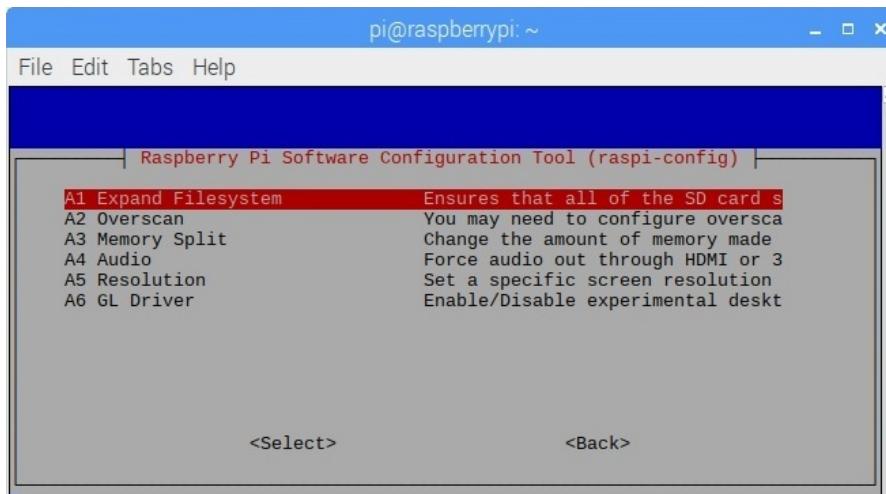
$\mu$ SD card ရဲ့ storage တွေ ကို အပြည့် အဝ သုံးဖို့ အတွက် သူရဲ့ filesystem ကို expand လုပ်ချင် ရင် အောက်က command နဲ့ Raspberry Pi Software Configuration Tool (raspi-config) ကို ဖွင့်စိုင် ပါတယ်။

```
$ sudo raspi-config
```

Configuration tool ပေါ်လာတဲ့ အခါ ပုံ ၁.၁၉ မှာ ပြထား သလို Advanced Options ကို ရွေးလိုက် ပါမယ်။ အဲဒီ နောက် ပေါ်လာတဲ့ options တွေ ထဲက Expand Filesystem ကို ရွေးလိုက် ရင် root partition ကို resize လုပ်ပေး သွားတာ ကို တွေ့ရ ပါလိမ့် မယ် (ပုံ ၁.၂၀ နှင့် ပုံ ၁.၂၁)။



ပုံ ၁.၁၉: Raspberry Pi Software Configuration Tool ကို ဖွင့်ခြင်း။



ဗုံး။ Advanced options တွင် filesystem ကို ချေရန် ရွေးချယ်ခြင်း။



ဗုံး။ Root partition ကို ချေခြင်း။

## အကိုးအကားများ

[Dah15] Oyvind Nydal Dahl. Raspberry Pi or Arduino? 2015. url: <https://www.build-electronic-circuits.com/raspberry-pi-or-arduino/>.

- [Leo14] Michael Leonard. How to Choose the Right Platform: Raspberry Pi or Beagle-Bone Black? 2014. url: <http://makezine.com/2014/02/25/how-to-choose-the-right-platform-raspberry-pi-or-beaglebone-black/>.
- [ras17a] raspberrypi.org. Backups. 2017. url: <https://www.raspberrypi.org/documentation/linux/filesystem/backup.md>.
- [ras17b] raspberrypi.org. Installing Operating System Images. 2017. url: <https://www.raspberrypi.org/documentation/installation/installing-images/README.md>.
- [Tib17] Tibor. How do I backup my Raspberry Pi? 2017. url: <https://raspberrypi.stackexchange.com/questions/311/how-do-i-backup-my-raspberry-pi>.

## အခန်း J

# အကြော်လုပ်ဆောင်မှုများ

### J.၁ Network ပြင်ဆင်ခြင်း

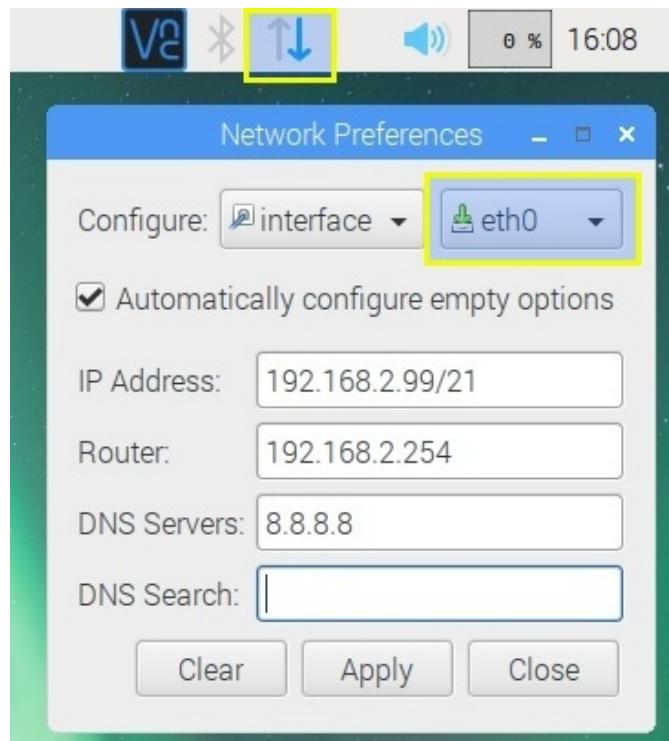
Raspberry Pi ကို network ချိတ်ဆက် အသုံးပြုဖို့ ပြင်ဆင် ပါမယ်။ Raspberry Pi မှာ Ethernet cable နဲ့ ဆက်ဖို့ connector ပါသလို wireless network adapter လည်း ပါတဲ့ အတွက် wireless lan နဲ့ လည်း ချိတ်ဆက် အသုံးပြု လိုရ ပါတယ်။

Wireless lan နဲ့ ချိတ်ဆက် ဖို့ အတွက် task bar ပေါ်က Wireless & Wired Network ဆိုတဲ့ အပေါ် အောက် မြား နှစ်ခု နဲ့ icon ကို ကလစ် နိုပ်ပြီး ဆက်ချင် တဲ့ SSID ကိုရွေးပြီး ချိတ်ဆက် နိုင် ပါတယ်။ အဲဒီ မှာ Wi-Fi ကို အပိတ် အဖွင့် လည်း လုပ်လို ရပါ တယ်။

ပုံမှန် အားဖြင့် Raspberry Pi ကို Dynamic Host Configuration Protocol (DHCP) သုံးပြီး IP address တစ်ခု အလို အလေ့ အလျောက် ရယူ အသုံး ပြုပြီး ဆက်သွယ် နိုင်အောင် စီမံ ထား ပါတယ်။ DHCP မရှိ လို IP address မသတ်မှတ် ရသေးရင် ဒါမှမဟုတ် static IP တစ်ခု ကိုယ်ဟာ ကိုယ် သတ်မှတ် ဖို့ လိုခဲ့ရင် Network ကို ပြန် configure လုပ်ဖို့ လို ပါမယ်။

အဲဒီ အတွက် ပုံ မှာ ပြထား သလို Task bar ပေါ်က Wireless & Wired Network icon ကို ညာဖက် ကလစ် နိုပ်ပြီး ပေါ်လာ တဲ့ context menu ထဲက Wireless & Wired Network Settings ကို နိုင် လိုက် ပါမယ်။ အဲဒီ မှာ Network interface တွေကို configure လုပ်ဖို့ Network Preferences ဆိုတဲ့ ဝင်းဒိုး ပေါ်လာ မှာ ဖြစ် ပါတယ်။ Interface နှစ်ခု ရှိနေမှာ ဖြစ်ပြီး eth0 က Ethernet interface နဲ့ wlan0 က wireless lan interface အတွက် ဖြစ် ပါတယ်။

နူးနာ အနေနဲ့ eth0 ကို static IP ဖြစ်အောင် သတ်မှတ် တာကို ပုံ J.၁ မှာ တွေ့နိုင် ပါတယ်။ အဲဒီ မှာ Gateway တွေ၊ Domain Name System (DNS) server တွေပါ တခါ တည်း သတ်မှတ် နိုင် ပါတယ်။



ပုံ J.၁: Static IP သတ်မှတ်ခြင်း။

ပိုကောင်း တာက တော့ သူရဲ့ media access control address (MAC address) ကို စစ်ကြည့်  
ဖိုး၊ Router မှာ သူအတွက် IP address တစ်ခု ကို reserved လုပ်တဲ့ နည်းပါ။ အဲဒါ ဆို firmware ကို  
ပြောင်းလိုက် ရင်တောင်၊ reserved လုပ်ထား တဲ့ network ကို လာ ဆက်လိုက် တိုင်း၊ သူရဲ့ သတ်မှတ်  
ထားတဲ့ IP address က မပြောင်းလဲ ပဲ အရင် အတိုင်း ပဲ ပြန်ရ နေမှာ ပါ။ စက်ရဲ့ MAC address ကို  
အောက်က အတိုင်း စစ်ကြည့် နိုင် ပါတယ် (ပုံ J.၂)။

```
$ cd /sys/class/net
$ ls
$ cat eth0/address
```

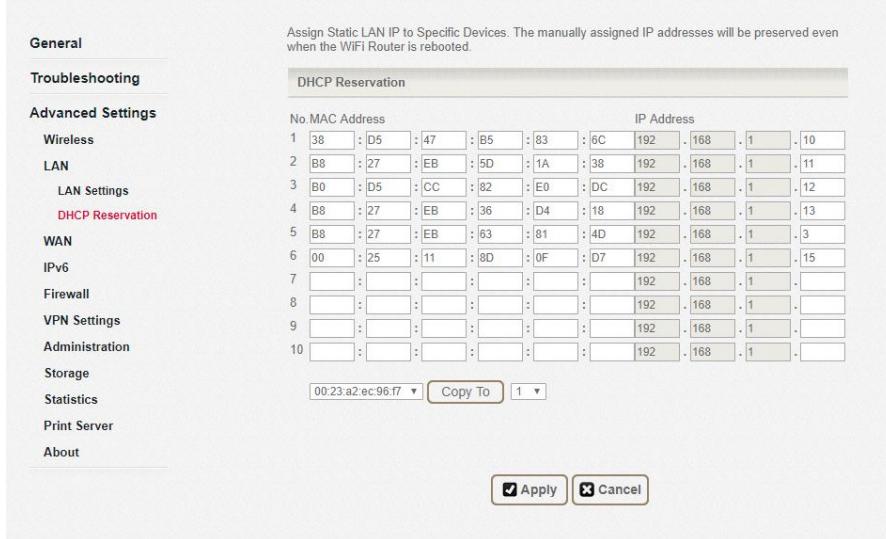
```
pi@raspberrypi:~ $ cd /sys/class/net
pi@raspberrypi:/sys/class/net $ ls
eth0  lo  wlan0
pi@raspberrypi:/sys/class/net $ cat eth0/address
b8:27:eb:fe:16:a8
```

ပုံ J.၂: MAC address ကို စစ်ခြင်း။

## J.C. NETWORK ပြင်ဆင်ခြင်း

JO

Router မှာ MAC address တွေကို reserved လုပ်တဲ့ နမူနာ တစ်ခု ကို ပုံ J.R မှာ ပြထား ပါတယ်။



ပုံ J.R: MAC address ကို reserved လုပ်ခြင်း။

## J.C. Network ဆက်သွယ်မှုကိုစစ်ခြင်း

Terminal မှာ ifconfig ဆိုတဲ့ command ကို ရိုက်ကြည့် လိုက်ရင် ပုံ J.C မှာလို သတ်မှတ် ထားတဲ့ IP address အတိုင်း ဖြစ်နေ တာကို တွေ့နိုင် ပါတယ်။

```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.98  netmask 255.255.248.0  broadcast 192.168.7.255
                ...
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
            loop  txqueuelen 1  (Local Loopback)
            RX packets 9 bytes 524 (524.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 9 bytes 524 (524.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ပုံ J.C: Static IP ကို စစ်ခြင်း။

အင်တာနက် ဆက်သွယ်မှု အဆင်ပြေ မပြေ စစ်ကြည့် ဖို့ အတွက် တော့ အောက် ကအတိုင်း ping

လုပ်ကြည့် နိုင်ပါတယ်။

```
$ ping google.com
```

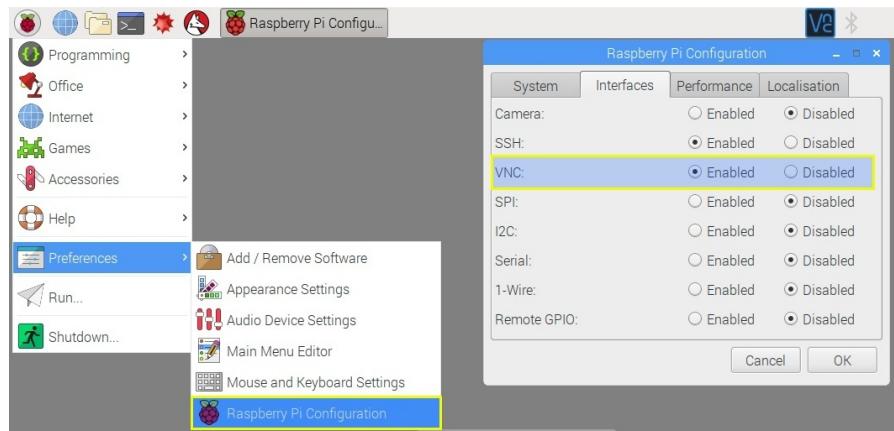
ပြီးတဲ့ အခါ ping လုပ်တာကို ရပို့ အတွက် ctrl+c ကို နှိပ်နိုင် ပါတယ်။ Raspberry Pi ရဲ့ network ဆက်သွယ်မှု အဆင်ပြေ တာနဲ့ သူကို keyboard တွေ၊ monitor တွေ ချိတ်ထား စရာ မလို တော့ပဲ၊ ကိုယ့်ရဲ့ Desktop ကွန်ပျူးတာ ကနေ VNC တို့၊ SSH တို့ သုံးပြီး ခုန် က သတ်မှတ် ခဲ့တဲ့ IP address ကို လုမ်းပြီး ဆက်သွယ် အသုံးပြု လိုရ ပါတယ်။

## J.J VNC ဖြင့်အသုံးပြုခြင်း

Virtual Network Computing (VNC) က တစ်ခြား ကွန်ပျူးတာ တစ်ခု ရဲ့ desktop ကို အဝေးက နေ လုမ်းသုံး လို ရတဲ့ စနစ် တစ်ခု ပါ။ အဲဒီ အတွက် အသုံးပြုခဲ့ မယ့် စက် မှာ VNC server ရှိဖို့ လိုပါတယ်။

### J.J.၁ VNC Server

Raspiberry Pi မှာ VNC server တပ်ဆင် ပြီးတဲ့ အခါ သူကို အဝေး ကနေ ပဲ desktop ကွန်ပျူးတာ ကနေ လုမ်းသုံး နိုင် ပါတယ်။ အဲဒီ အတွက် Application Menu→Preferences→Raspberry Pi Configuration ကို ဖွင့် လိုက် ပါမယ်။ ပေါ်လာ တဲ့ configuration ဝင်းဒိုး ရဲ့ Interfaces tab မှာ VNC ကို ပုံ J.၅ မှာ ပြထား သလို Enabled လုပ်နိုင် ပါတယ်။

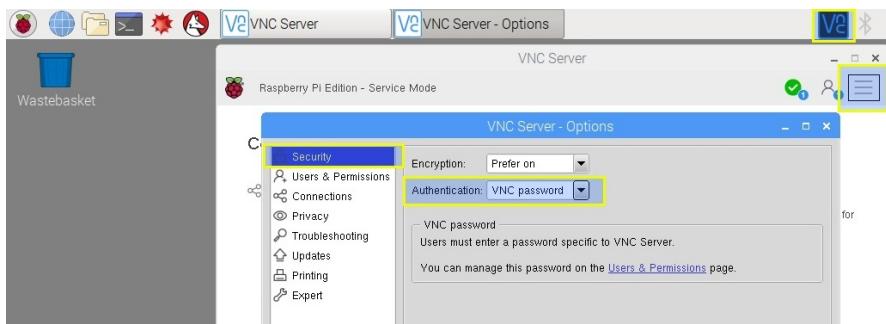


ပုံ J.၅: VNC ကို enabled လုပ်ခြင်း။

## J.J. VNC ဖြင့်အသုံးပြုခြင်း

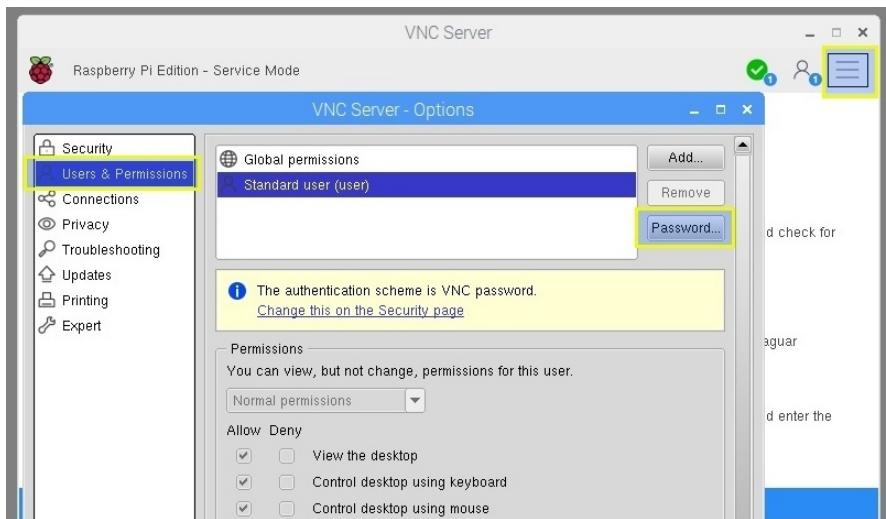
JR

အဲဒီ နောက် task bar ပေါ်က VNC icon ပေါ်ကို နှိပ်ပြီး VNC server ဝင်းဒုံး ကို ဖွင့်စိုင် ပါတယ်။ ပေါ်လာ တဲ့ ဝင်းဒုံး ရဲ့ Menu ကို နှိပ်ပြီး VNC Server - Options ဝင်းဒုံး ကို ပဲ ပြထား သလို ထပ်ဖွဲ့ ပါမယ်။ ပြီးတဲ့ အခါ Security ရဲ့ Authentication မှာ VNC password ကို ရွေ့နိုင် ပါတယ်။ အဲဒါ က VNC viewer တော်တော် များများ နဲ့ အဆင်ပြေ ပါတယ်။



ပဲ J.၆: VNC အတွက် authentication ကို သတ်မှတ်ခြင်း။

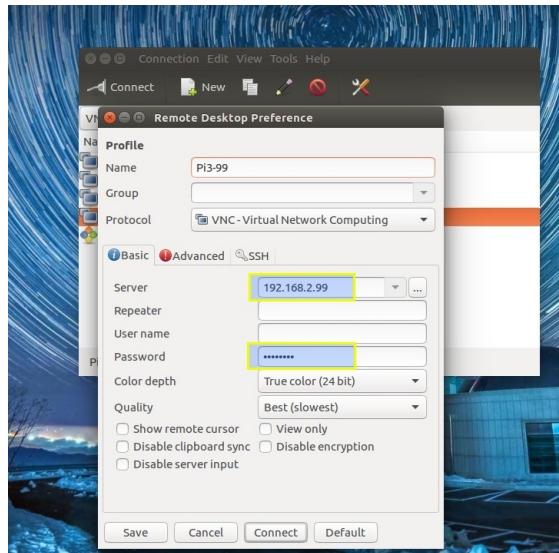
နောက် တစ်ခါ user & permissions ကို နှိပ်ပြီး VNC အတွက် password ကို သတ်မှတ်နိုင် ပါတယ် (ပဲ J.၇)။ အခု လို setup လုပ်ထား တဲ့ VNC server က Raspberry Pi စက်ဖွဲ့ တိုင်း အတူတူ ပွင့်လာ မှာ ဖြစ်တဲ့ အတွက်၊ Raspberry Pi ပါဝါ ပွင့်နေ တိုင်း လုမ်းပြီး ချိတ်ဆက် အသုံးနိုင် မှာ ဖြစ်ပါတယ်။



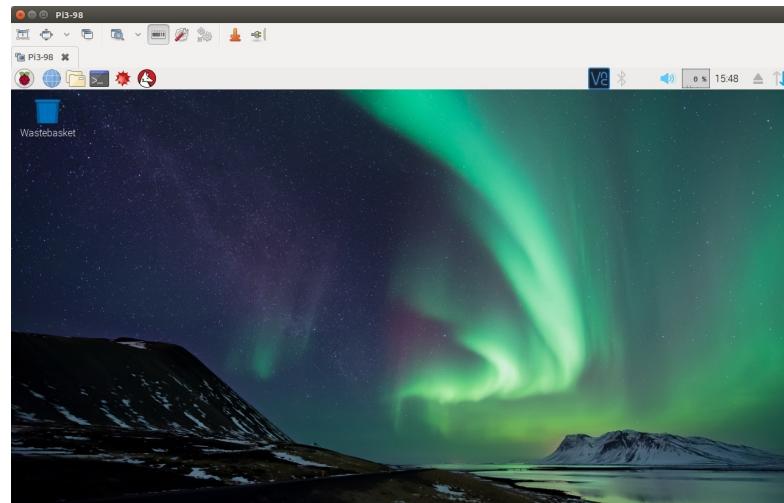
ပဲ J.၇: VNC အတွက် password သတ်မှတ်ခြင်း။

### J.J.J VNC Client

Raspberry Pi မှာ server ကို ဖွင့်ပြီး သွားရင် ကိုယ့်ရဲ့ client စက်မှာ vnc viewer တစ်ခုခု သံဃားပြီး လုမ်းချိတ်ဆက် အသုံးပြု လို ရပါပြီ။ Platform အမျိုးမျိုး အတွက် [RealVNC Viewer](#) စတာ တွေကို သုံးနိုင် ပါတယ်။



ပုံ J.၈: Remmina ဖြင့် VNC ချိတ်ဆက်ခြင်း။



ပုံ J.၉: Raspberry Pi ၏ desktop ကို VNC ဖြင့် အသုံးပြုခြင်း။

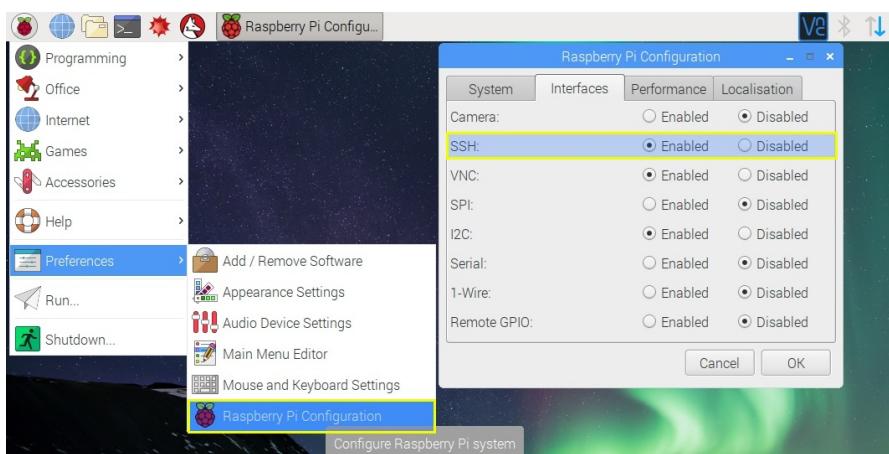
## J.2. SSH ဆက်သွယ်အသုံးပြုခြင်း

၂၅

Ubuntu Linux မှာ ပါလာ ပြီးသား ဖြစ်တဲ့ Remmina Remote Desktop Client သုံးတဲ့ နမူနာ ကို ပုံ J.၈ မှာ ပြထား ပါတယ်။ တခြား client တွေ အတွက် လည်း 192.168.2.99 အတိုင်း အတူတူ ဖြစ်ပြီး VNC ချိတ်ဆက် ပြီးတဲ့ အခါ Raspberry Pi ရဲ့ desktop ကို ကိုးဘုတ်၊ မောက်စ် တွေနဲ့ လှမ်းသုံး လို ရပါဖြီ (ပုံ J.၉)။

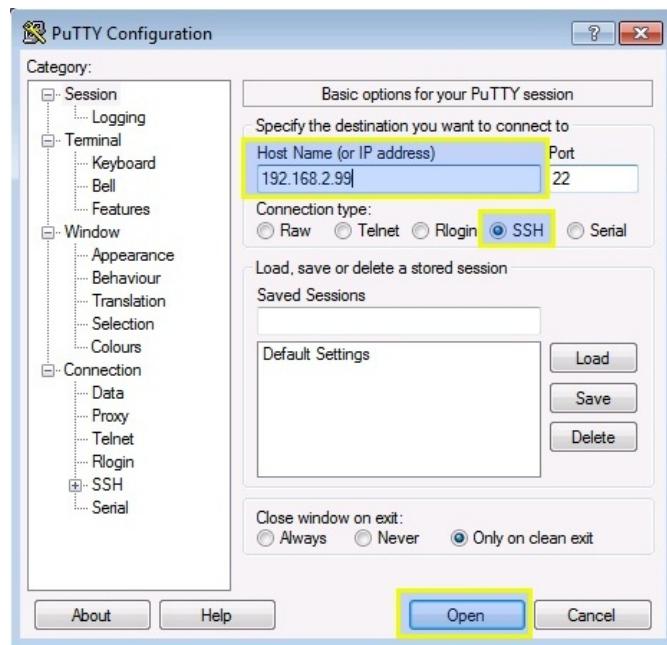
## J.2 SSH ဆက်သွယ်အသုံးပြုခြင်း

Secure SHell (SSH) ကို Raspberry Pi မှာ အသုံးပြု ဖို့ အတွက် Application Menu→Preferences→Raspberry Pi Configuration ကို ဖွင့် လိုက် ပါမယ်။ ပေါ်လာ တဲ့ configuration ဝင်းဒိုး ရဲ့ Interfaces tab မှာ SSH ကို ပုံ J.၁၀ မှာ ပြထား သလို Enabled လုပ်နိုင် ပါတယ်။



ပုံ J.၁၀: SSH ကို enabled လုပ်ခြင်း။

Raspberry Pi မှာ SSH ကို ဖွင့်ပြီး တဲ့အခါ သူကို desktop ကွန်ပျိုးတာ ကနေ လှမ်းပြီး ဆက်သွယ် အသုံးပြု ဖို့ အတွက် SSH client တစ်ခုကို အသုံးပြု နိုင် ပါတယ်။ Windows အတွက် ဆိုရင် PuTTY ကို <http://www.putty.org/> ကနေ ယူပြီး တပ်ဆင် အသုံးပြု လိုရ ပါတယ်။ ပြီးတဲ့ အခါ PuTTY ကို ဖွင့်ပြီး Raspberry Pi ရဲ့ IP address ဉာပမာ 192.168.2.99 ကို ပုံ J.၁၁ မှာ ပြထား သလို ထည့်ပြီး ဆက်သွယ် နိုင် ပါတယ် [Mon15]။



ပုံ၂.၁၁: PuTTY အသုံးပြု၍ ဆက်သွယ်ခြင်း။

Security alert ပေါ်လာရင် Yes ကို ရွေးလို ရပြီး Login as မှာ username နဲ့ password ကို ရှိက်ထည့် လိုက်ရင် ပုံ၂.၁၂ မှာလို Raspberry Pi ကို ဆက်သွယ် သွား တာကို တွေ့နှိုင် ပါတယ်။ Username က pi နဲ့ default password က raspberry ဖြစ် ပါတယ်။

```
pi@raspberrypi: ~
login as: pi
pi@192.168.2.99's password:
Linux raspberrypi 4.9.41-v7+ #1023 SMP Tue Aug 8 16:00:15 BST 2017 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Jan 13 11:28:13 2018 from 192.168.2.71
pi@raspberrypi: ~ $
```

ပုံ၂.၁၂: SSH လုပ်ဆောင်ပုံ။

Linux ဒါမှ မဟုတ် Mac ကို သုံးတယ် ဆိုရင် တော့ terminal ကို ဖွင့်ပြီး အောက်က စာရင်း၂.၁၃ မှာ ပြထား တဲ့ command ကို ထည့်ပြီး ဆက်သွယ် နိုင် ပါတယ် (ပုံ၂.၁၄)။ အဲဒီမှာ pi က username အတွက် ဖြစ်ပြီး Are you sure you want to continue connecting (yes/no)? လို မေးတဲ့ အခါ yes

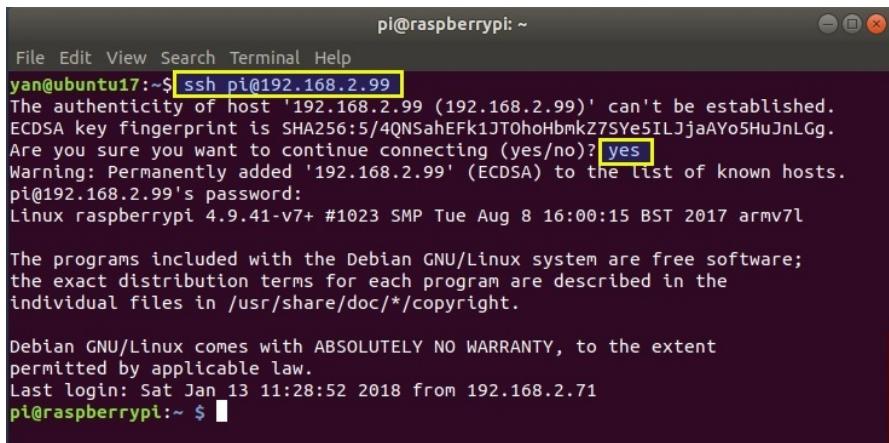
## ၂.၃. SSH ဆက်သွယ်အသုံးပြုခြင်း

၂၇

ကို ရှိက်ထည့် နှင့် ပါတယ်။ နောက်တစ်ခါ password တောင်း တဲ့ အခါမှာ raspberry ကို ထည့်ပေး နှင့် ပါတယ်။

```
1 $ ssh pi@192.168.2.99
```

စာရင်း ၂.၁၁: Username 'pi' အတွက် SSH ကိုသုံးချုံဆက်သွယ်ခြင်း။



ပုံ ၂.၁၃: Linux SSH session

Raspberry Pi ကို SSH နဲ့ လှမ်း ဆက်သွယ် တဲ့ အခါ password တွေ ပြန်ပြန် မထည့် ချင်ရင် စာရင်း ၂.၁၂ မှာ ပြထား သလို လုပ်ထား လိုရ ပါတယ်။ Passphrase တို့ ဘာတို့ တောင်းတဲ့ အခါ ဘာမှ မဝည့်ပဲ enter ပဲ နှိပ်ပြီး ဆက်သွား လို ရပါတယ်။

```
1 $ ssh-keygen  
2 $ ssh-copy-id root@192.168.2.92  
3 $ ssh-add
```

စာရင်း ၂.၂၂: SSH အတွက် ပြင်ဆင်ခြင်း။

## ၂.၃.၁ Password ကို ပြောင်းခြင်း

အခါ default username နဲ့ password က Raspberry Pi အားလုံး အတူတူ ဖြစ်တာ ကြောင့် security ရှိချင်ရင် အောက်က command သုံးပြီး ပြောင်းလို ရပါတယ်။

```
$ sudo passwd pi
```

တကယ်လို root user ကို enable လုပ်ချင်ရင် တော့

```
$ sudo passwd root
```

ကို ရိုက်ထည့် ပြီး password ကို နှစ်ခါ ပြန်ထည့် ပေးပါမယ်။ ပြီးရင်

```
$ sudo passwd -u root
```

ကို ရိုက်ထည့်ပြီး unlock လုပ်နိုင် ပါတယ်။ အဲဒီ နောက် SSH အတွက် အောက်က command သုံးပြီး /etc/ssh/sshd\_config ကို edit လုပ်နိုင် ပါတယ်။

```
$ sudo nano /etc/ssh/sshd_config
```

အဲဒီ config ထိုင် ထဲမှာ PermitRootLogin ကို အောက်ပါ အတိုင်း ပြင်နိုင် ပါတယ်။

```
PermitRootLogin yes
```

ပြီးရင် restart ပြန်လုပ်ပြီး root ကို သုံးနိုင် ပါတယ်။

### J.၃.J Password ကိုဖြဲတြဲခြင်း

User တစ်ယောက် ဥပမာ yan ရဲ့ password ကို ဖြဲတြဲချင်ရင် အောက်က အတိုင်း ဖြဲတြဲနိုင် ပါတယ်။

```
$ sudo passwd yan -d
```

### J.၃.၃ User အသစ်ဖန်တီးခြင်း

User အသစ် ဖန်တီး ချင်ရင် အောက်က command အတိုင်း သုံးပြီး ဖန်တီး နိုင် ပါတယ်။ အဲဒီ user အတွက် password ကို တစ်ခါ ထဲ တောင်းပါ လိမ့်မယ်။

```
$ sudo adduser yan
```

### ၂.၃.၄ Sudoers

ပုံမှန်အားဖြင့် pi ဆိုတဲ့ user က sudoer ဖြစ်ပါတယ်။ သူက command တွိကို root အနေနဲ့ run ချင်ရင် ရွှေမှာ sudo ခံပြီး run လို ရပါတယ်။ ဖန်တီးလိုက်တဲ့ user အသစ်ကို sudoer ဖြစ်စေ ချင်ရင်

```
$ sudo visudo
```

ကို သုံးပြီး ပြင်နိုင် ပါတယ်။ အဲဒီမှာ

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
```

ဆိုတဲ့ လိုင်းတွေ ကို လိုက်ရှာပြီး root ရဲ သတ်မှတ်ချက် ကို copy ကူးပြီး yan ဆိုတဲ့ user အတွက် password ထည့်စရာ မလိုပဲ sudo သုံးလို ရအောင် အောက် က အတိုင်း ထပ်ဖြည့် နှင့် ပါတယ်။

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
yan    ALL=NOPASSWD : ALL
```

သူရဲ နှင့် မူရင်း pi ဆိုတဲ့ user ကလည်း NOPASSWD လုပ် ထားတဲ့ အတွက် password ထည့်စရာ မလိုပဲ sudo သုံးလို ရ ပါတယ်။ ကိုယ့်စက် ကို security ပိုရှိ စေချင် ရင်တော့ pi ကို password တောင်း အောင် ပြင်လို ရပါတယ်။ အဲဒီ အတွက် sudoers ဖိုင်ထဲ မှာ

```
pi    ALL=(ALL:ALL) ALL
```

ဆိုတဲ့ စာကြောင်း ကို နောက်ဆုံး လိုင်းမှာ ထည့်ပေး နှင့် ပါတယ်။

### J.၃.၅ User ဖျက်ခြင်း

ရှိဖြိုးသား User ကို ဖျက် ချင်ရင် အောက်က command ကို သုံးပြီး ဖျက်နိုင် ပါတယ်။ အဲဒီမှာ -r ဆိုတဲ့ option က သူရဲ့ home အခန်း ကိုပါ ဖျက်ဖို့ ဆိုလို ပါတယ်။

```
$ sudo userdel -r yan
```

### J.၄ Internet ကို မျှဝေယူခြင်း

တစ်ခါ တစ်ရုံ မှာ Raspberry Pi ကို router နဲ့ ဖြစ်ဖြစ်၊ wireless hotspot နဲ့ ဖြစ်ဖြစ် တိုက်ရိုက် ချိတ်ပြီး အင်တာနက် သုံးဖို့ အတွက် အဆင် မပြော တဲ့ အခါ ရှိနိုင် ပါတယ်။ အဲဒီ အခါ desktop ကွန်ပျိုးတာ နဲ့ တိုက်ရိုက် ချိတ်ပြီး၊ သူရဲ့ အင်တာနက် ကို တဆင့် မျှယူ ပြီး သုံးနိုင် ပါတယ်။

ဥပမာ wireless အင်တာနက် သုံးနေ တဲ့ ကွန်ပျိုးတာ ကို network ကြိုး သုံးပြီး Raspberry Pi နဲ့ ဆက်လို ရပါတယ်။ အဲဒီ အခါ host ကွန်ပျိုးတာ နဲ့ Raspberry Pi အပြန် အလှန် ဆက်သွယ် လို ရပြီး ကွန်ပျိုးတာ ရဲ့ အင်တာနက် ဆက်သွယ်မှု ကို အဲဒီ network interface က တဆင့် Raspberry Pi ကို မျှဝေ အသုံးပြု ချင်ရင် လည်း သုံးလို ရပါ တယ်။ အဲဒီ လို မဟုတ်ပဲ အင်တာနက် မသုံးရင် တောင်မှ ကွန်ပျိုးတာ နဲ့ Raspberry Pi ကို network ကြိုးနဲ့ တိုက်ရိုက် ဆက်ပြီး VNC တို့ သုံးချင် တဲ့ အခါ မှာလည်း သုံးနိုင် ပါတယ်။

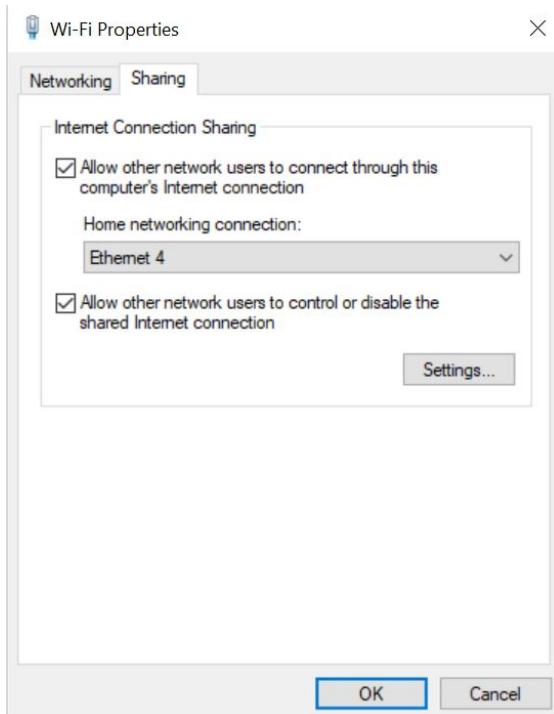
### J.၄.၁ Windows Host PC တွင် ပြင်ဆင်ခြင်း

Control Panel → Network and Internet → Network Connections ကို သွားပြီး ကြည့်လိုက် မယ် ဆိုရင် ပဲ J.၁၄ မှာ ပြထား သလို internet နဲ့ ချိတ်ထား တဲ့ wireless connection နဲ့ Raspberry Pi ကို ချိတ်ဆက် ထားတဲ့ wired connection တွေကို တွေ့နိုင် ပါတယ်။



ပဲ J.၁၄: Control Panel ရှိ Network Connections များ။

ပြီးရင် Windows ဂွန်ပျိုတာ ရဲ့ အင်တာနက် နဲ့ ချိတ်ဆက် ထားတဲ့ adapter ပေါ်မှာ ညာဘက် ကလစ် ကို နှိပ်ပြီး၊ သူရဲ့ Properties ကို ဖွင့်လိုက် ပါမယ်။ ပွင့်လာ တဲ့ Properties box မှာ Sharing tab ကို သွားပြီး ပုံ ၂၁၅ မှာ ပြထား သလို Allow other network users to connect through this computer's Internet connection ဆိုတဲ့ checkbox မှာ tick လုပ်၊ Home networking connection ဆိုတဲ့ dropdown box မှာ ခုန် က RPi နဲ့ ချိတ်ဆက် ထားတဲ့ connection ကို ရွေးပေးလိုက် ပါမယ်။

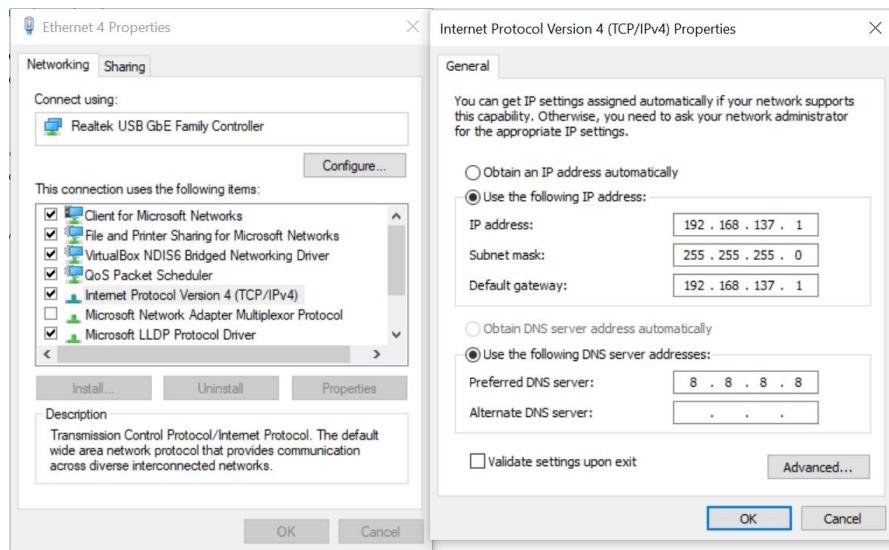


ပုံ ၂၁၅: Internet connection ကို Sharing လုပ်ခြင်း။

အဲဒီ လို sharing လုပ်လိုက် တဲ့ အခါ RPi နဲ့ ချိတ်ဆက် ထားတဲ့ connection ရဲ့ properties တွေက ပြောင်းသွား တာကို တွေ့ရ ပါတယ်။ ဒါကြောင့် RPi နဲ့ ချိတ်ဆက် ထားတဲ့ connection ကို ပြန် config-ure လုပ်ဖို့ အဲဒီ ပေါ်မှာ ညာဖက် ကလစ် နှိပ်ပြီး properties ကို ရွေးလိုက် ပါမယ်။ ပြီးရင် ပေါ်လာတဲ့ Properties box က Networking tab ကို သွား၊ Internet Protocol Version 4 ကို ရွေးပြီး၊ ညာဘက် အောက်နား က Properties ခလုတ် ကို နှိပ်လိုက် ပါမယ်။

နောက်ထပ် Internet Protocol Version 4 (TCP/IPv4) Properties ဆိုတဲ့ box တစ်ခု ထပ်ပေါ် လာမှာ ဖြစ်ပြီး၊ အဲဒီ မှာ Use the following IP address ဆိုတဲ့ option ကို ရွေးလိုက် ပါမယ်။ IP address ကို 192.168.137.1 လို့ သူမှာ ရှိနေ တဲ့ အတိုင်း ထားလိုက် ပါမယ်။ ကိုယ်ဟာ ကိုယ် စိတ်ကြိုက်

သတ်မှတ် လိုလည်း ရပါတယ်။ Subnet mask ကို 255.255.255.0 လို့ ဖြည့်ပြီး၊ Default gateway ကို လည်း 192.168.137.1 လို့ ထားလိုက် ပါမယ်။ Preferred DNS Server နေရာ မှာ 8.8.8.8 ကို သုံးနိုင် ပါတယ်။ ပြီးတဲ့ အခါ ပုံ J.၁၆ မှာ ပြထား အတိုင်း OK ကို နှိပ်လိုက် ပါမယ်။



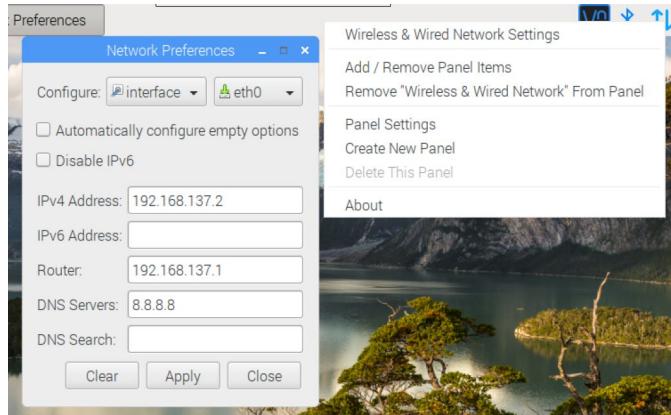
ပုံ J.၁၆: RPi အတွက် connection ကို ပြန် configure လုပ်ခြင်း။

### J.၄.J Raspberry Pi တွင် ပြင်ဆင်ခြင်း

Host PC မှာ ပြင်ဆင် ပြီးတဲ့ အခါ ပုံ J.၁၇ အတိုင်း Raspberry Pi ရဲ့ terminal မှာ ifconfig ဆိုတဲ့ command နဲ့ သူရဲ့ IP address စစ်ကြည့် နိုင်ပြီး 8.8.8.8 ကို ping လုပ်ကြည့် ပြီး အင်တာနက် ရ မရ စစ်ကြည့် နိုင် ပါတယ်။ မရ သေးရင် Raspberry Pi ရဲ့ connection မှာ လည်း အင်တာနက် ချိတ်ဆက်ဖို့ အတွက် ပြင်ဆင် ဖို့လို ပါတယ် [Cyn15]။

```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi: ~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.137.2 netmask 255.255.255.0 broadcast 192.168.137.255
              ether b8:27:eb:36:d4:18 txqueuelen 1000  (Ethernet)
              RX packets 5617 bytes 346949 (338.8 KiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 12204 bytes 14138005 (13.4 MiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ပုံ J.၁၇: RPi ၏ connection ကို ifconfig ဖြင့်စစ်ခြင်း။



ပုံ J.၁၈: RPi အတွက် connection ကို ပြန် configure လုပ်ခြင်း။

ပြီးတဲ့ အခါ 8.8.8.8 တို့ google.com တို့ကို ping လုပ် ကြည့်လိုက် ရင် အဆင်ပြေ သွားတာ ကို  
ပုံ J.၁၉ မှာ ပြထား သလို တွေ့နိုင် ပါတယ်။

```
pi@raspberrypi:~ $ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=41 time=7.12 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=41 time=6.08 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=41 time=6.54 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 6.089/6.585/7.126/0.429 ms
pi@raspberrypi:~ $ ping google.com
PING google.com (172.217.194.100) 56(84) bytes of data.
64 bytes from 172.217.194.100: icmp_seq=1 ttl=40 time=4.82 ms
64 bytes from 172.217.194.100: icmp_seq=2 ttl=40 time=6.32 ms
64 bytes from 172.217.194.100: icmp_seq=3 ttl=40 time=6.57 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 4.822/5.907/6.577/0.774 ms
```

ပုံ J.၁၉: Internet ဆက်သွယ်မှု ကို စစ်ဆေးရန် google.com ကို ping လုပ်ခြင်း။

## J.6.2 Linux Host PC တွင် ပြင်ဆင်ခြင်း

Linux PC ကို host အနေ နဲ့ သုံးတယ် ဆိုရင် လည်း internet ဆက်သွယ် ကို share လုပ်ပေး နိုင် ပါတယ်။ Ubuntu Linux ကို သုံးထား တဲ့ host နဲ့ နမူနာ ဖော်ပြ ပါမယ်။ RPi ကို ဆက်သွယ် လိုက်တဲ့ အခါ ကွန်ပျူးတာ မှာ ပေါ်လာတဲ့ network interface ရဲ့ IP address ကို ကြည့်ဖို့ အတွက် command prompt မှာ အောက်ပါ command ကို သုံးပြီး ကြည့်နိုင် ပါတယ်။

```
$ ifconfig
```

ပုံ J.J၁ မှာ ပြထား တဲ့ နမူနာ မှာ RPi အတွက် interface က enx00e04c68005a ဖြစ်ပြီး၊ wireless interface က wlp2s0 ဖြစ်တာ ကို တွေ့နိုင် ပါတယ်။ RPi အတွက် interface ရဲ့ IP address ကို လည်း 192.168.137.1 လို့ သတ်မှတ် နိုင် ပါတယ် ပုံ J.J၁။

```

yan@air:~$ ifconfig
enx00e04c68005a: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.15.98 netmask 255.255.255.0 broadcast 192.168.15.255
        inet6 fe80::20bc:e92a:384d:2bbf prefixlen 64 scopedid 0x20<link>
            ether 00:e0:4c:68:00:5a txqueuelen 1000 (Ethernet)
                RX packets 22 bytes 3387 (3.3 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 48 bytes 5175 (5.1 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

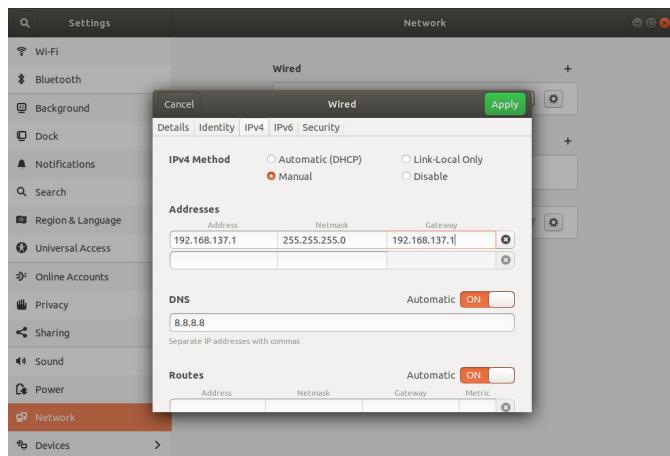
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopedid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
                RX packets 226 bytes 15710 (15.7 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 226 bytes 15710 (15.7 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.226 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::3b61:d493:a0aa:8184 prefixlen 64 scopedid 0x20<link>
            ether 28:37:37:24:38:5e txqueuelen 1000 (Ethernet)
                RX packets 1033 bytes 945950 (945.9 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 555 bytes 59126 (59.1 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
                device interrupt 17

yan@air:~$ route
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref    Use Iface
default         Singtel-ACPlus 0.0.0.0       UG    600    0      0 wlp2s0

```

ပုံ J.J၁: Linux တွင် network interface များကို စစ်ခြင်း။



ပုံ J.J၂: RPi အတွက် network interface တွင် IP address သတ်မှတ်ခြင်း။

ပြီးတဲ့ အခါ network interface ကို ပြန်ပိတ်၊ ပြန်ဖွင့် ပြီး စတင် ဖို့ လိုပါတယ်။ သတ်မှတ် တဲ့ အတိုင်း ရမရ ကို ifconfig နဲ့ ပြန်စစ် နိုင် ပါတယ်။ အဲဒါက host ကွန်ပျူးတာ ရဲ့ အင်တာနက် ရှိတဲ့ interface

'wlp2s0' ကို RPi အတွက် interface 'enx00e04c68005a' မှာ share လုပ်ဖို့ အတွက် အောက်ပါ command တွက် သုံးလိုက် ပါမယ် [Bol13]။

```
$ sudo iptables --table nat --append POSTROUTING --out-interface wlp2s0 -j MASQUERADE
$ sudo iptables --append FORWARD --in-interface enx00e04c68005a -j ACCEPT
$ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
```

အဲဒီ လို host ကွန်ပျိုတာ ဘက်မှာ configure လုပ်ပြီး RPi ကို လည်း အပိုင်းခဲ့ J.၄.J မှာ ဖော်ပြထားတဲ့ အတိုင်း ပြန် configure လုပ် ပြီးတဲ့ အခါ internet ဆက်သွယ်မှု ရရှိ သွားပါ လိမ့်မယ်။

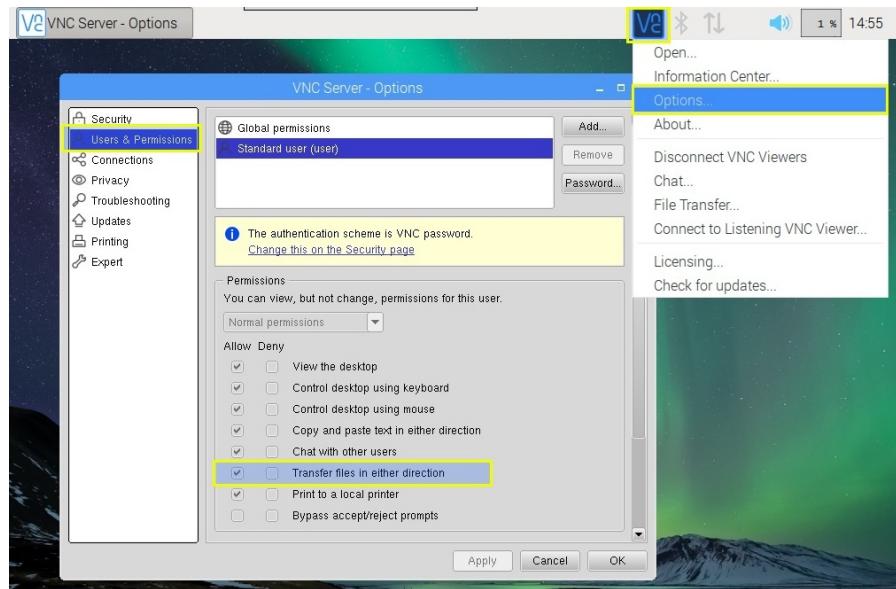
## J.၅ ဖိုင်များပေးပိုဂယူခြင်း

Raspberry Pi နဲ့ desktop ကွန်ပျိုတာ နဲ့ ဖိုင်တွေ အပြန် အလှန် ပေးပို့ ဖလှယ် တဲ့ အကြောင်း ဆွေးနွေးချင် ပါတယ်။

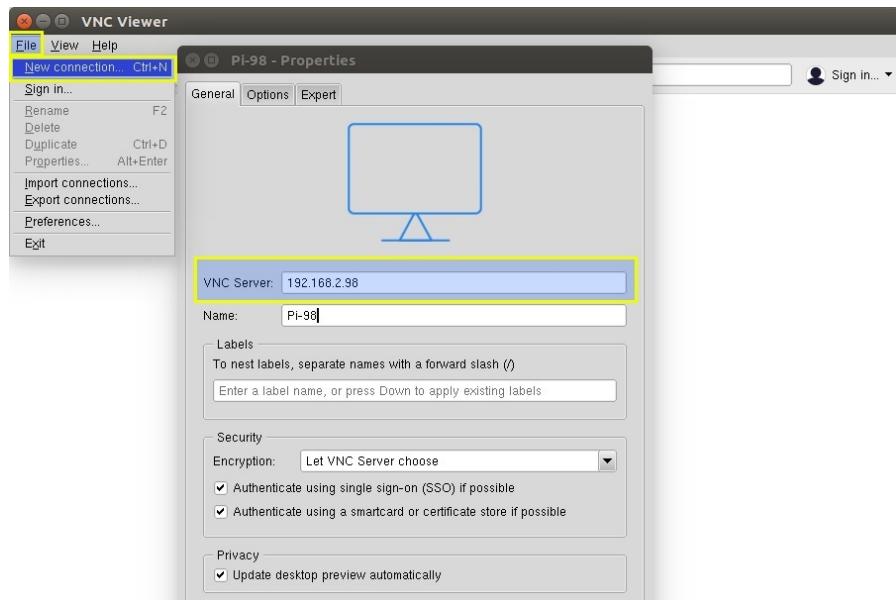
### J.၅.၁ VNC

တကယ်လို Raspberry Pi မှာ VNC server ကို setup လုပ် တုန်းက file transfer ကို နိုင် အတိုင်း ခွင့်ပြုထားရင် (ပုံ J.၂၂)၊ ပြီးတော့ VNC client ကို RealVNC Viewer သုံးမယ် ဆိုရင် file တွေကို Raspberry Pi ရဲ့ Real VNC server နဲ့ အပြန် အလှန် ပြောင်းပို့ နိုင် ပါတယ်။ RealVNC Viewer နဲ့ Raspberry Pi ရဲ့ VNC server ကို လုမ်း ဆက်သွယ် ပုံကို ပုံ J.၂၃ မှာ ပြထား ပါတယ်။

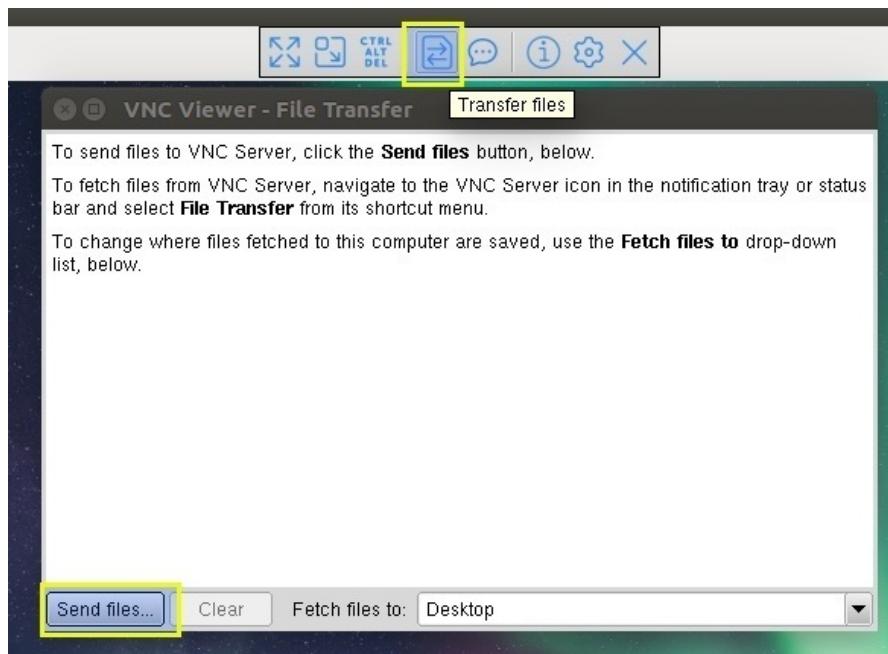
Desktop ကွန်ပျိုတာ ကနေ VNC server ရှိတဲ့ Raspberry Pi ဆီ ဖိုင်တွေ ပို့ချင်ရင် mouse pointer ကို VNC viewer ရဲ့ ထိပ်နား လေးကို ရွှေ့လိုက် တဲ့ အခါ ပေါ်လာတဲ့ tool box ထဲက transfer file ဆိုတဲ့ icon ကို နှိပ်ပြီး ပေါ်လာတဲ့ ဝင်းဒိုး ထဲက Send files ... ဆိုတဲ့ ခလုတ် ကို နှိပ်ပြီး ပို့လို့ ရပါတယ် (ပုံ J.၂၄)။



ဦး J.JJ: VNC server တွင် ဖိုင်ပေးပို့ခြင်းကို ခင့်ပြုခြင်း။

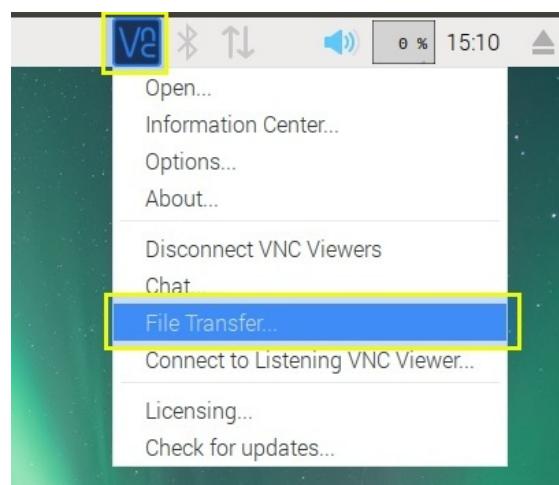


ဦး J.JR: Real VNC viewer ဖြင့် Raspberry Pi ကို လှမ်း၍ ဆက်သွယ် ချိတ်ဆက်ခြင်း။



ံ ၂.၂၃: Real VNC viewer ဖြင့် ဖိုင်များ ပိုခြင်း။

VNC server ဆိုက ဖိုင်တွေ ကို လှမ်းယူ ချင်ရင်တော့ task bar ပေါ်က VNC icon ပေါ်မှာ ညာဖက် ကလစ် နိုပ်ပြီး ပေါ်လာ တဲ့ shortcut menu ထဲက File Transfer ... ကို ရွေးနိုင် ပါတယ် (ံ ၂.၂၄)။



ံ ၂.၂၄: Real VNC viewer ဖြင့် ဖိုင်များ ယူခြင်း။

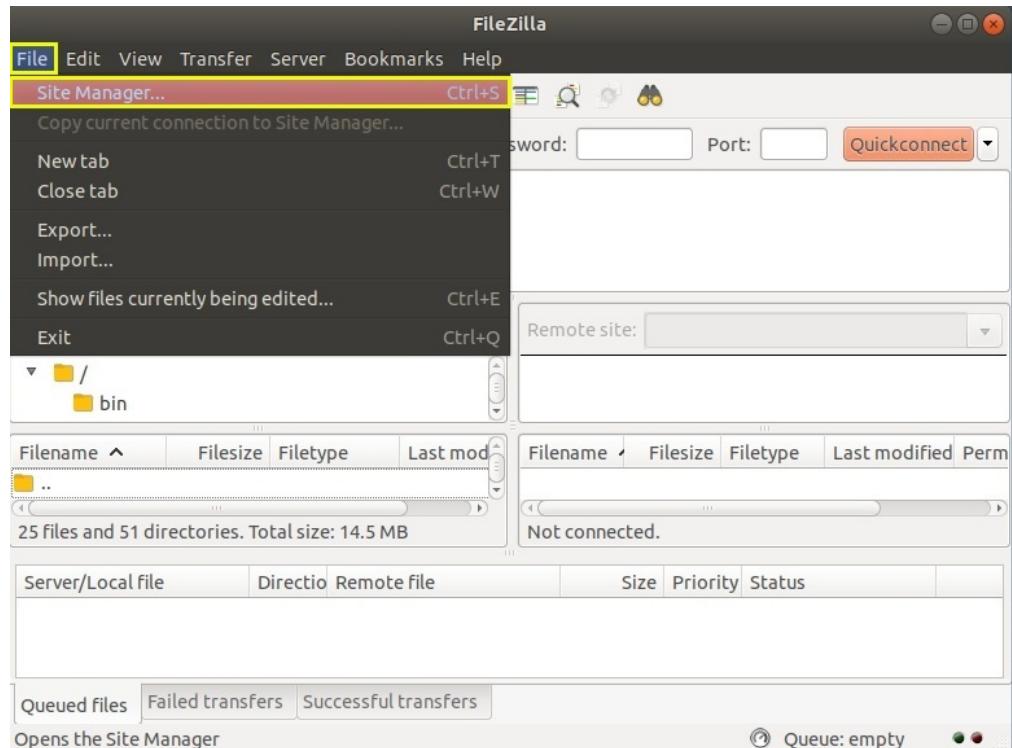
## J.၂.J SFTP

Raspberry Pi မှာ SSH ကို ဖွင့်ထား တယ် ဆိုရင် SSH File Transfer Protocol (SFTP) ကို သုံးပြီး ဖိုင်တွေ ပေးပို့ လိုလည်း ရပါတယ်။ နမူနာ အနေ နဲ့ scp command ကို ကွန်ပျိုးတာ terminal ကနေ သုံးပြီး test.cpp ဆိုတဲ့ ဖိုင်ကို Raspberry Pi သိ လှမ်းပိုတဲ့ ပုံစံ တစ်ခု ကို အောက်မှာ ပြထား ပါတယ် (ပုံ J.၂.၆)။

```
$ scp test.cpp pi@192.168.2.98:/home/pi/
```

```
yan@ubuntu17:~$ scp test.cpp pi@192.168.2.98:/home/pi/
pi@192.168.2.98's password:
test.cpp                                         100%   107     63.1KB/s   00:00
yan@ubuntu17:~$
```

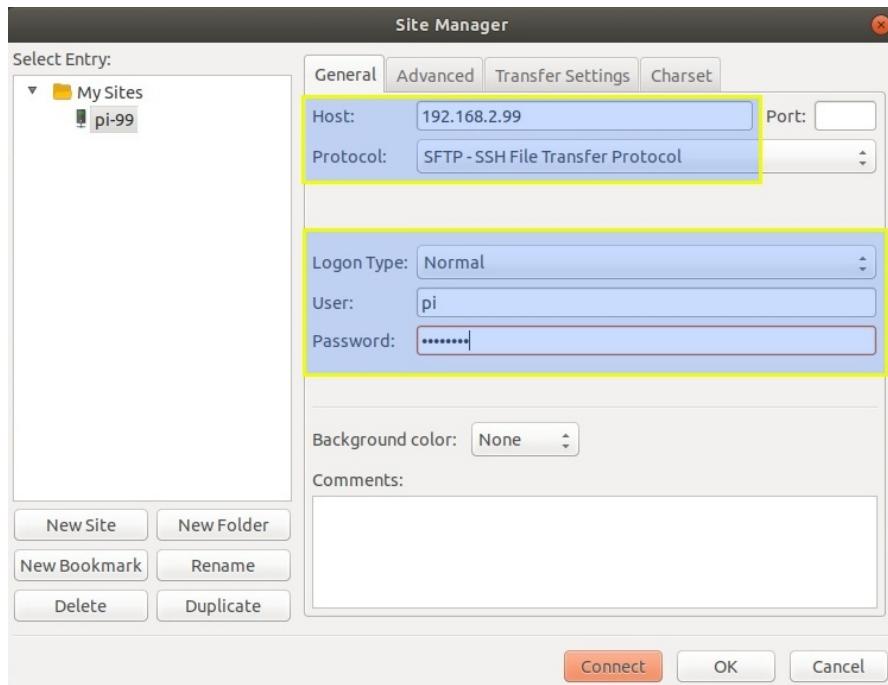
ပုံ J.၂.၆: scp ဖြင့် ဖိုင်ပြခြင်း။



ပုံ J.၂.၇: FileZilla ကို သုံးခြင်း။

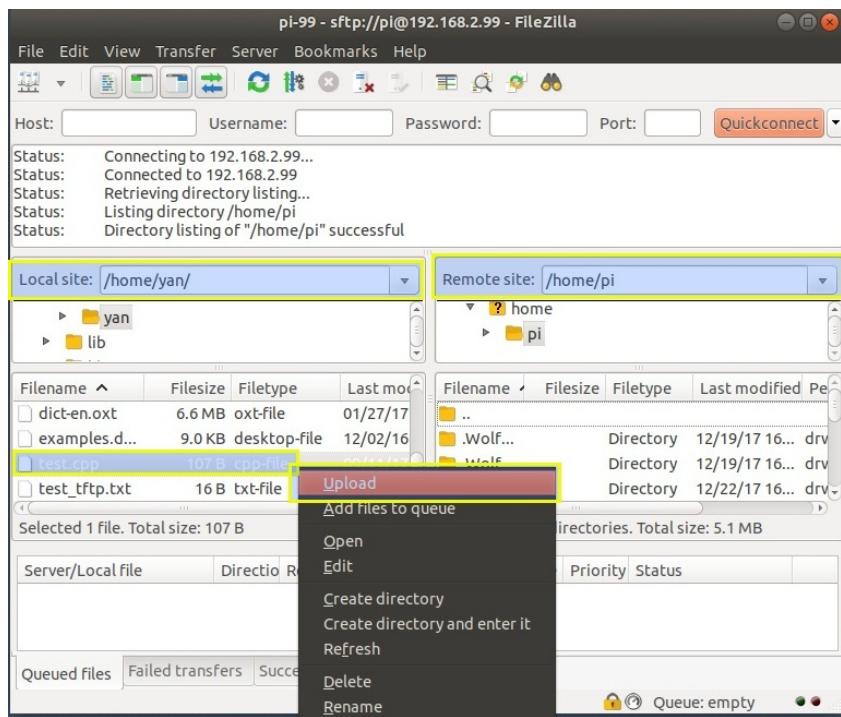
ပိုပြီး အဆင်ပြေ တဲ့ နည်းကတေသာ FileZilla စတဲ့ ftp client တွေကို သုံးတဲ့ နည်း ဖြစ်ပါ တယ်။ FileZilla ကို တပ်ဆင်ပြီး ဖွင့်ပြီး တဲ့ အခါ ပုံ J.J9 မှာ ပြထား သလို Site menu→Site Manager... ကို နိုင် ပါမယ်။

Site Manager ဝင်းဒီး ပေါ်လာ တဲ့ အခါ host အတွက် IP address ၊ protocol အတွက် SFTP တွေကို ရွေး၊ Logon type ကို normal နဲ့ username ၊ password တွေ ထည့်ပြီး တဲ့ အခါ connect ကို နိုင်ပြီး ဆက် သွယ်နိုင် ပါတယ် (ပုံ J.J9)။



ပုံ J.J9: SFTP ဖြင့် ဆက်သွယ်ခြင်း။

ဆက်သွယ် ပြီးတဲ့ အခါ local site နဲ့ remote site တွေကို သတ်မှတ်၊ ဖိုင်တွေ၊ အခန်း တွေ ပေါ်မှာ ညာဖက် ကလစ် နိုင်ပြီး ပိုတာ၊ လက်ခံတာ၊ ပျက်တာ တွေကို လွယ်လွယ် ကူကူ၊ လျင်လျင် မြန်မြန် လုပ်နိုင် ပါပြီ (ပုံ J.J9)။



ဗုံး J.၂၆: FileZilla ဖြင့် ဖိုင်များ ပေးပို့ ရယူခြင်း။

## J.၆ Sources list ကိုပြင်ဆင်ခြင်း

Raspberry Pi မှာ လို အပ်တဲ့ software packages တွေကို apt ဆိုတဲ့ application ကို သုံးပြီး တပ်ဆင် လို ရပါတယ်။ Software packages တွေကို သူတို့ ရဲ့ source ကနေ မရယူ နိုင်ပဲ ပြဿနာ တက်နေ ရင် sources list ကို ပြင်ဆင် ဖို့ လိုပါတယ်။ အဲဒီ အတွက် /etc/apt/sources.list ဆိုတဲ့ ဖိုင် ကို nano သုံးပြီး edit လုပ်တဲ့ အခါ word wrap ကြောင့် new line မတော်တစာ ပါဘား တာ မျိုး မဖြစ်အောင် အောက်က command ကို သုံးလိုက် ပါမယ်။

```
$ sudo nano -w /etc/apt/sources.list
```

Sources list ဖိုင် ပွင့်လာ တဲ့ အခါ

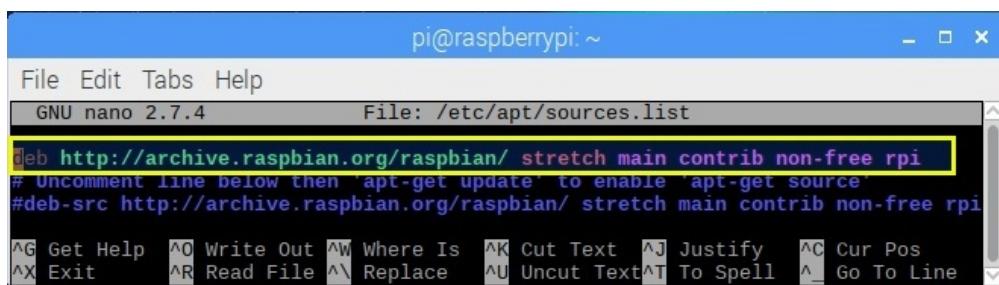
```
deb http://mirrordirector.raspbian.org/raspbian/ stretch main contrib non-free rpi
```

ဆိုတဲ့ စာကြောင်းကို

```
deb http://archive.raspbian.org/raspbian/ stretch main contrib non-free rpi
```

ဆိုပြီး ပဲ J.၃၀ မှာ ပြထား သလို အစားထိုး နှင့် ပါတယ်။ အဲဒီ နောက် Ctrl+o နဲ့ ပြင်ရေး လိုက်ပြီး၊ Ctrl+x ကို နိုင်ပြီး nano ကနေ exit လုပ်လိုက် ပါမယ်။ ပြီးတဲ့ အခါ terminal မှာ အောက်က command ကို သုံးပြီး update လုပ်တဲ့ အခါ apt နဲ့ software packages တွေ ရယူ တပ်ဆင် ဖို့ အဆင်ပြု သွားတာ ကို တွေ့ရ ပါတယ်။

```
$ sudo apt update
```



ပဲ J.၃၀: sources.list ကို ပြင်ဆင်ခြင်း။

## J.7 C/C++ ပရိုဂရမ်ရေးသားခြင်း

Raspberry Pi ကို SSH နဲ့ ဆက်ပြီး တဲ့ အခါ ရိုးရှင်း တဲ့ C/C++ ပရိုဂရမ် နမူနာ လေး တစ်ခု ရေးကြည့် ပါမယ်။ ပရိုဂရမ် လေး ကို nano သုံးပြီး ရေးထို့ SSH terminal မှာ စာရင်း J.၃ က command ကို ရိုက်ထည့် လိုက်ပါမယ် [Eli13]။

```
1 $ nano eg.cpp
```

စာရင်း J.၃: Nano သုံး၍ C ပရိုဂရမ် ရေးသားခြင်း။

ပြီးတဲ့ အခါ စာရင်း J.၄ မှာ ပြထားတဲ့ eg.cpp ဆိုတဲ့ နမူနာ C ပရိုဂရမ် ကို ရေးထည့် ပြီး၊ Ctrl+X နဲ့ ပိတ်၊ သိမ်းဖို့ အတွက် Y ကို နိုင်ပြီး တဲ့ အခါ၊ လက်ရှိ ဖိုင် နာမည် eg.cpp နဲ့ပဲ သိမ်းဖို့ Enter ကို နိုင်နိုင် ပါတယ်။

```

1 #include<iostream>
2 #include<stdio.h>
3 using namespace std;
4 int main(){
5     cout<<"Example C/C++ program."<<endl;
6     for(int i=0;i<10;i++){
7         printf("%d \n",i);
8     }
9     return 0;
10 }
```

စာရင်း ၂.၄: ရီးရှင်းသောနမူနာ C/C++ ပရိုကရမ်။

အဲဒီနောက် ပရိုကရမ် ကို build လုပ်ပြီး run ဖို့ စာရင်း ၂.၅ က command တွက် terminal မှာ ရိုက်ထည့် ပါမယ်။ ပရိုကရမ် run လိုက်တဲ့ အခါ terminal မှာ ပေါ်လာတဲ့ output တွက် ပုံ ၂.၃၁ မှာ ပြထား သလို တွေ့နှိုင် ပါတယ်။

```

1 $ g++ eg.cpp -o eg
2 $ ./eg
```

စာရင်း ၂.၅: C ပရိုကရမ်ကို build လုပ်၍ run ခြင်း။

```

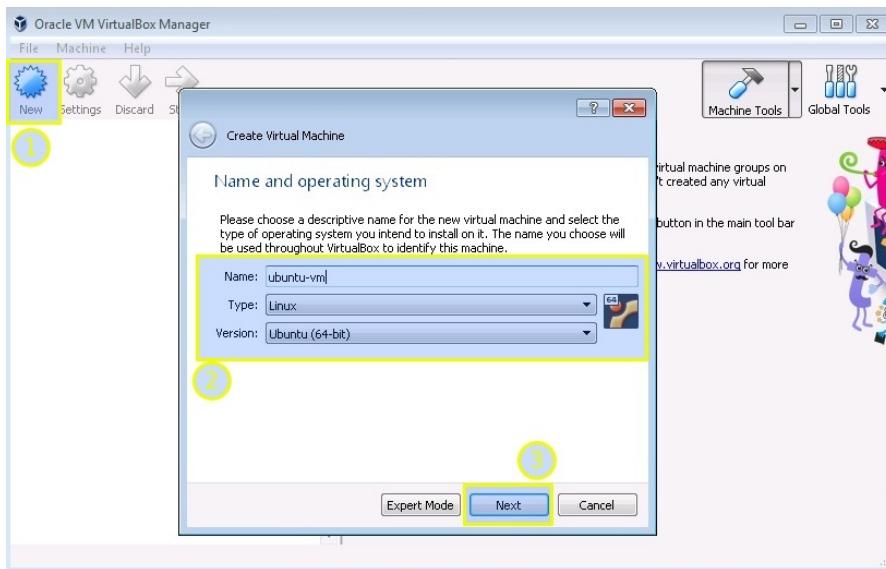
Terminal File Edit View Search Terminal Help
pi@raspberrypi:~/egcpp $ nano eg.cpp
pi@raspberrypi:~/egcpp $ g++ eg.cpp -o eg
pi@raspberrypi:~/egcpp $ ls
eg eg.cpp
pi@raspberrypi:~/egcpp $ ./eg
Example C/C++ program.
0
1
2
3
4
5
6
7
8
9
pi@raspberrypi:~/egcpp $
```

ပုံ ၂.၃၁: eg.cpp ပရိုကရမ်၏ output ကို တွေ့ရပုံ။

## J.၉ Linux Virtual Machine တစ်ခု တပ်ဆင်ခြင်း

Raspbian က Debian Linux ကို အခြေခံ ထားတာ ဖြစ်တဲ့ အတွက်၊ သူလို ပဲ Debian ကို အခြေခံ ထားတဲ့ Ubuntu တို့ Linux Mint တို့လို operating system တွေ တပ်ဆင် ထားတဲ့ desktop ကွန်ပျူးတာ တစ်လုံး ရှိရင် application တွေကို ဖန်တီးဖို့ စွမ်းဆောင်ရည် ပိုမြင့်တဲ့ Integrated Development Environment (IDE) တွေ သုံးတာ၊ နောက် အပိုင်း မှာ ဆွေးနွေး ထား သလို cross compilation tool chain တွေ၊ emulator တွေ သုံးတာ မျိုး အတွက် ပိုမြြိုး အဆင်ပြေ ပါတယ်။ အဲဒီ အတွက် နောက်ထပ် ကွန်ပျူးတာ အပို့ မရှိရင် Windows စက်ပေါ် မှာပဲ Linux virtual machine တစ်ခု တပ်ဆင် အသုံးပြု လို ရပါတယ်။

နမူနာ အနေနဲ့ free application တစ်ခု ဖြစ်တဲ့ Oracle ရဲ့ VirtualBox ([virtualbox.org](http://virtualbox.org)) ကို သုံးပြီး Ubuntu ([ubuntu.com](http://ubuntu.com)) ကို တပ်ဆင် အသုံးပြု ပါမယ်။ Windows hosts အတွက် VirtualBox နောက်ဆုံး ဗားရှင်း ကို [download](#) လုပ်ပြီး တပ်ဆင် ပါမယ်။ ပြီးတဲ့ အခါ ပုံ J.၂၂ မှာ ပြထား သလို New ကို နှိပ်ပြီး Virtual Machine အသစ် တစ်ခု ကို ဖန်တီး နိုင် ပါတယ်။ စက်နာမည် နဲ့ အမျိုးအစား Linux အတွက် version ကို သင့်တော် သလို ရွေးနိုင် ပါတယ်။

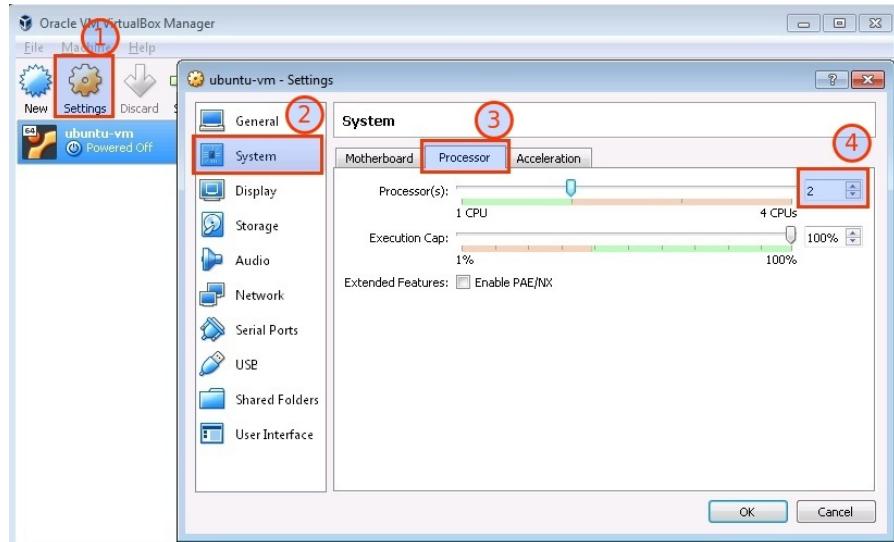


ပုံ J.၂၂: Virtual machine တစ်ခု ဖန်တီးခြင်း။

Memory အရွယ် အစား ကို လက်ရှိ နောက်ဆုံး Ubuntu 17 အတွက် ရဲ့ အနည်းဆုံး လိုအပ်ချက် ဖြစ်တဲ့ 2 GB ရွေးပြီး၊ Create a virtual hard disk ကို ရွေးပြီး virtual hard disk တစ်ခု ဖန်တီး။

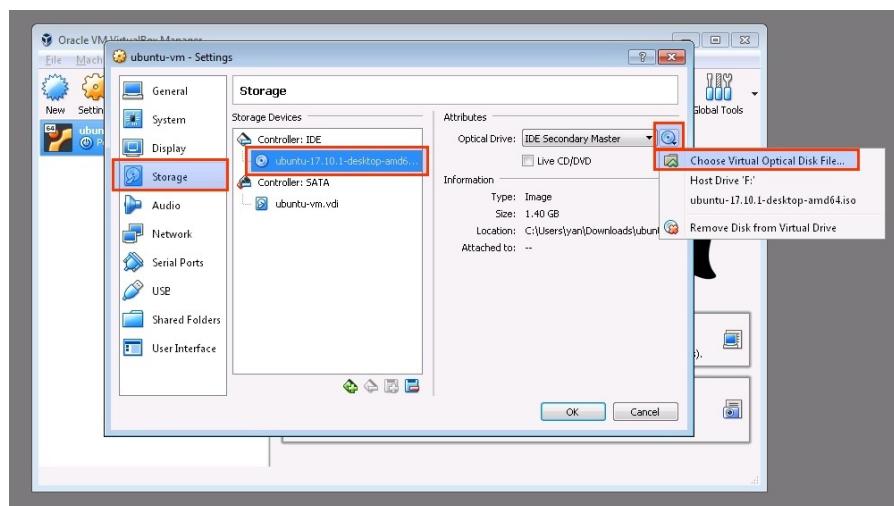
အမျိုးအစား နဲ့ အရွယ် (အနည်းဆုံး 25 GB) ကို သင့်တော် သလို ရွေးနိုင် ပါတယ်။

ဖန်တီး လိုက်တဲ့ virtual machine ရလာ တဲ့ အခါ သူကို select လုပ် ပြီး Settings ကို နှိပ်လိုက် ပါမယ်။ ပေါ်လာတဲ့ ဝင်းဒီး က system → processor မှာ အနည်းဆုံး 2 ကို ရွေးနိုင် ပါတယ် (ပုံ J.၃၃)။



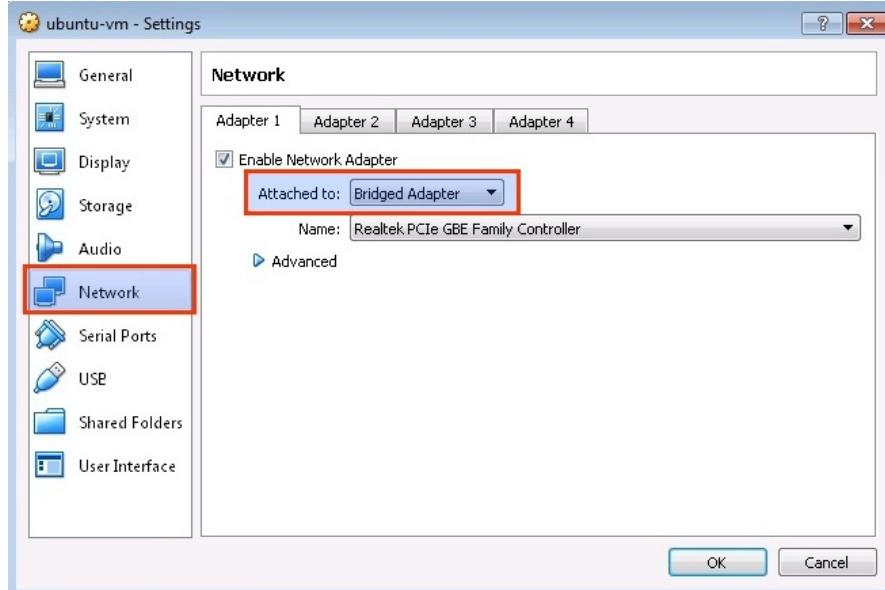
ပုံ J.၃၃: Virtual machine တစ်ခု ဖန်တီးခြင်း။

Storage က Optical disk အတွက် ပုံ မှာ ပြထား သလို Utuntu website ကနေ download လုပ် လိုက်တဲ့ image ဖိုင်ကို ရွေးပေး နိုင် ပါတယ်။

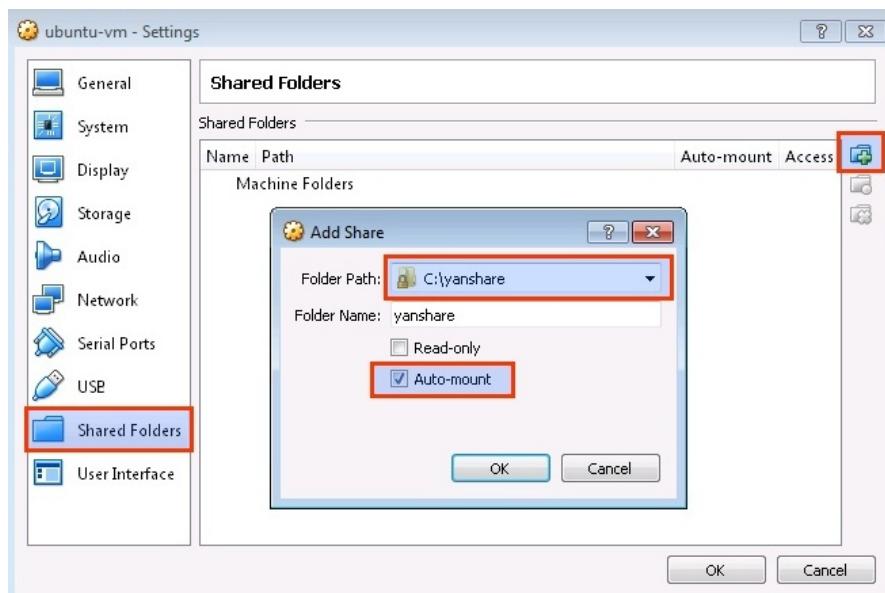


ပုံ J.၃၄: တပ်ဆင်မည့် OS ၏ image ကို သတ်မှတ်ခြင်း။

Virtur machine အတွက် network adapter မှာ Bridge Adapter ကို ရွေးပေး မယ်ဆို ရင် host machine တို့ Raspberry Pi တို့နဲ့ network တစ်ခု ထဲမှာ အတူ ရှိသွား ပါမယ် (ပုံ ၂၃၅)။

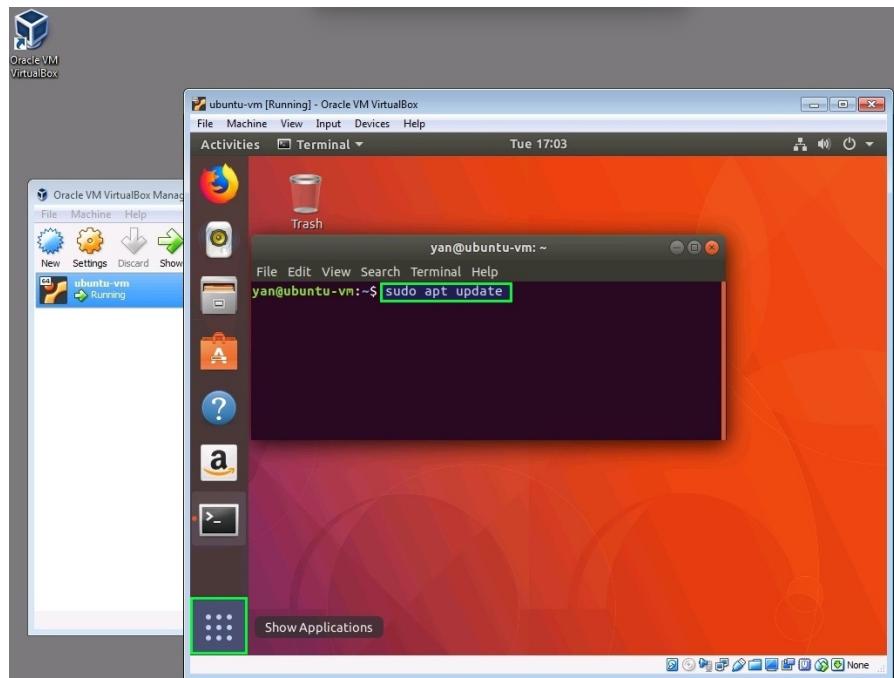


ပုံ ၂၃၅: Network adapter ကိုသတ်မှတ်ခြင်း။



ပုံ ၂၃၆: Share folder ကိုသတ်မှတ်ခြင်း။

Host machine နဲ့ ဖိုင်တွေ အပြန် အလှန် share လုပ်ဖို့ share folder တစ်ခု ကို သတ်မှတ် ပါမယ်။ အဲဒီ အတွက် ပုံ J.၃၆ အတိုင်း share folders ကို သွား Adds new shared folder ကို နှိပ် ပြီး folder ကို သတ်မှတ်၊ auto mount လုပ်ဖို့ သတ်မှတ် တာတွေ လုပ်နိုင် ပါတယ်။ အားလုံး စိတ်ကြိုက် သတ်မှတ် ပြီးတဲ့ အခါ OK ကို နိုပ်၊ Start ကို နှိပ်ပြီး virtual machine ကို စတင် နိုင် ပါတယ်။

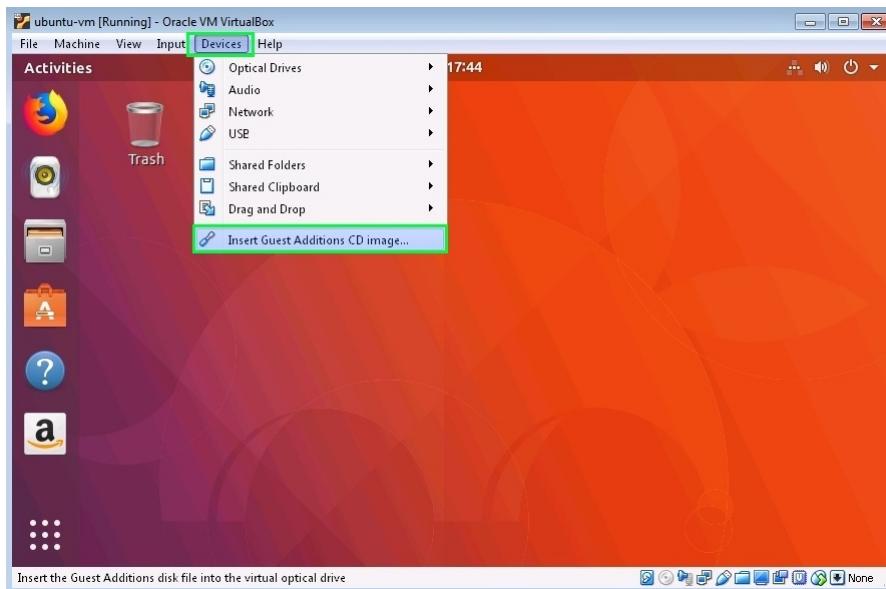


ပုံ J.၃၇: လိုအပ်သော software packages များတပ်ဆင်ခြင်း။

Virtual machine ကို run လိုက်ပြီး၊ စက်ပွင့် လာတဲ့ အခါ သူ optical disk အတွက် သတ်မှတ် ထားတဲ့ Ubuntu ကို အလိုက်သင့် တပ်ဆင် နိုင်ပါတယ်။ Ubuntu ကို တပ်ဆင် ပြီးတဲ့ အခါ ပုံ J.၃၇ မှာ ပြထား သလို Show Applications ကို နှိပ်ပြီး၊ Terminal ကို ရှာဖွင့် လိုက် ပါမယ်။ ပြီးရင် အောက်က command တွေကို ရိုက်ထည့် ပြီး လိုအပ် တဲ့ software packages တွေကို တပ်ဆင် လိုက် ပါမယ်။

```
$ sudo apt update
$ sudo apt install build-essential
```

အဲဒီနောက် ပုံ J.၃၈ မှာ ပြထား သလို Virtual machine ဝင်းဒိုးမှာ Devices menu ကို နှိပ်ပြီး၊ Insert Guest Additions CD image... ကို နှိပ်ပြီး တပ်ဆင် လိုက် ပါမယ်။



ံ J.၃၈: Guest additions ကို တပ်ဆင်ခြင်း။

Share folder ကို သုံးနိုင် အောင် Virtual machine ရဲ့ terminal ကို ဖွံ့ဖြိုး အောက်ပါ အတိုင်း username နေရာ မှာ သတ်မှတ် ထားတဲ့ နာမည် ပြောင်းထည့် ပြီး ရိုက်ထည့် နိုင် ပါတယ်။

```
$ sudo usermod -a -G vboxsf username
```

Virtual machine ကို reboot လုပ်ပြီးတဲ့ အခါ share folder က desktop ပေါ်မှာ mount လုပ်ထား တာကို တွေ့နိုင် ပြီး သုံးဖို့ အဆင် သင့်ဖြစ် နေတဲ့ virtual machine ကို ရပါ လိမ့်မယ်။

## J.၇ Cross Compilation Tool Chain

အပိုင်း J.၇ က နှမူနာ C++ ပရိုဂရမ် မှာ code ကို ရေးတာရော၊ execute လုပ်တာရော က Raspberry Pi ပေါ်မှာ ပဲ လုပ်တာပါ။ အဲဒါက ပရိုဂရမ် ကြီးလာ တဲ့ အခါ ရေးရာ ပြန်စစ်ရ တာ သိပ် အဆင် မပြေ ပါဘူး။ ဒါကြောင့် desktop computer ပေါ်မှာ ပဲ ပရိုဂရမ် ကို ရေးပါ မယ်။ compile လုပ်ရင် လည်း code ရေးတဲ့ desktop ပေါ်မှာ တစ်ခါထည်း လုပ်တာ က ပိုပြီး အဆင်ပြေ မြန်ဆန် ပါတယ်။ အဲဒီ အတွက် Ubuntu OS ပေါ်မှာ ARM architecture တွေ အတွက် compiler လုပ်ပေးမယ့် cross compiler ကို အရင် ထည့်ဖို့ လို ပါတယ်။ ပြီးရင် Code::Blocks IDE နဲ့ တွဲပြီး အသုံးပြု မှာပါ။

စစချင်း desktop ကွန်ပျူးတာ ထဲမှာ ARM စက်တွေ အတွက် compiler ကို install လုပ်ပါမယ်။

မတူညီ တဲ့ စက် architecture တွေရဲ့ လိုင်ဘရဲ့ တွေကို စက်တဲ့ ထဲမှာ တပ်ဆင် ဖို့အတွက် MultiArch [Wik17b] ကို သုံးပါမယ်။ လက်ရှိ စက်ရဲ့ အာခါ တက်ချာ ကို အောက်က command သုံးပြီး ကြည့်နိုင် ပါတယ်။

```
$ dpkg --print-architecture
```

Floating point နံပါတ် တွေကို တွက်ချက်ဖို့ hardware အထောက် အပံ့ ပါတဲ့ armhf အာခါ တက်ချာ ကိုထည့်ဖို့ အတွက် အောက်က ပထမ command သုံး ထည့်နိုင်ပြီး၊ စက်ထဲမှာ ရှိတဲ့ တွေး အာခါ တက်ချာ တွေကို အောက်က ဒုတိယ command နဲ့ ကြည့်နိုင် ပါတယ်။

```
$ sudo dpkg --add-architecture armhf
$ dpkg --print-foreign-architectures
```

အဲဒီ နောက် မှာတော့ အောက်က အတိုင်း arm အတွက် cross compiler ကို တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt-get install crossbuild-essential-armhf
```

```
yan@ubuntu17: /usr/bin
File Edit View Search Terminal Help
arm-linux-gnueabihf-ranlib      x86_64-linux-gnu-gold
arm-linux-gnueabihf-readelf    x86_64-linux-gnu-gprof
arm-linux-gnueabihf-size        x86_64-linux-gnu-ld
arm-linux-gnueabihf-strings     x86_64-linux-gnu-ld.bfd
arm-linux-gnueabihf-strip       x86_64-linux-gnu-ld.gold
cpans5.26-x86_64-linux-gnu     x86_64-linux-gnu-nm
i686-linux-gnu-pkg-config      x86_64-linux-gnu-objcopy
perl5.26-x86_64-linux-gnu      x86_64-linux-gnu-objdump
watchgnupg                      x86_64-linux-gnu-pkg-config
x86_64-linux-gnu-addr2line      x86_64-linux-gnu-python2.7-config
x86_64-linux-gnu-ar             x86_64-linux-gnu-python-config
x86_64-linux-gnu-as             x86_64-linux-gnu-ranlib
x86_64-linux-gnu-c++filt       x86_64-linux-gnu-readelf
x86_64-linux-gnu-cpp           x86_64-linux-gnu-size
x86_64-linux-gnu-cpp-5          x86_64-linux-gnu-strings
x86_64-linux-gnu-cpp-6          x86_64-linux-gnu-strip
x86_64-linux-gnu-cpp-7          x86_64-pc-linux-gnu-pkg-config
yan@ubuntu17:/usr/bin$ arm-linux-gnueabihf-g++ --version
arm-linux-gnueabihf-g++ (Ubuntu/Linaro 7.2.0-6ubuntu1) 7.2.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
yan@ubuntu17:/usr/bin$
```

ံ ၂၃၉: Cross compiler တပ်ဆင် ထားမှု။

ထည့်သွင်းတပ်ဆင် ပြီးတဲ့ အခါ terminal မှာ အောက်အတိုင်း ရိုက်ထည့်ပြီး စစ်ကြည့်နိုင်ပါတယ် (ပုံ J.၃၉)။

```
$ cd /usr/bin
$ ls *gnu*
$ arm-linux-gnueabihf-g++ --version
```

ထည့်လိုက် တဲ့ cross compiler ကို စမ်းကြည့် ဖို့ အတွက် ခုနက eg.cpp ဆိုတဲ့ နမူနာ ပရိုဂရမ် လေး ကိုပဲ အောက် က command နဲ့ Ubuntu desktop ကွန်ပျိုးတာ ရဲ့ terminal မှာ compile လုပ်ကြည့် ရင်လည်း ရတာ ကို တွေ့ရ ပါမယ်။ သူကို run ကြည့် လိုက်တဲ့ အခါ မှာတော့ လက်ရှိ စက်က amd64 အာမီ တက်ချာ မို့လို့ run လို့ မရ တာကို ပုံ J.၄၀ မှာ ပြထား သလို တွေ့နိုင် မှာပါ။

```
$ arm-linux-gnueabihf-g++ eg.cpp -o eg
```

```
yan@ubuntu17:~/egcpp$ arm-linux-gnueabihf-g++ eg.cpp -o eg
yan@ubuntu17:~/egcpp$ ./eg
bash: ./eg: cannot execute binary file: Exec format error
```

ပုံ J.၄၀: Cross compiler ကိုစမ်းကြည့်ခြင်း။

အဲဒီ ပရိုဂရမ် ကိုပဲ စာရင်း J.၆ မှာ ပြထားတဲ့ အတိုင်း Raspberry Pi ပေါ်ကို ကူးထည့်ပြီး run ကြည့်တဲ့ အခါ စက်နဲ့ architecture ကိုက်ညီ တဲ့ အတွက် အလုပ် လုပ်တာကို တွေ့ရ ပါမယ် (ပုံ J.၄၁)။

```
1 $ scp eg pi@192.168.1.98:/home/pi/
2 $ ssh pi@192.168.1.98
3 $ ./eg
```

စာရင်း J.၆: Cross compile လုပ်ထား သည့် ပရိုဂရမ် ကို Raspberry Pi ပေါ်သို့ ကူးထည့်ခြား run ခြင်း။

```

pi@raspberrypi: ~
File Edit View Search Terminal Help
yan@ubuntu17:~/egcpp$ scp eg pi@192.168.1.98:/home/pi/
pi@192.168.1.98's password:
eg                                         100% 8964      1.5MB/s   00:00
yan@ubuntu17:~/egcpp$ ssh pi@192.168.1.98
pi@192.168.1.98's password:
Linux raspberrypi 4.9.41-v7+ #1023 SMP Tue Aug 8 16:00:15 BST 2017 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Feb 6 16:22:38 2018
pi@raspberrypi:~ $ ./eg
Example C/C++ program.
0
1
2
3
4
5
6
7
8
9
pi@raspberrypi:~ $ 

```

ပုံ J.၄၁: Cross compile လုပ်ထားသည့် binary ကို Raspberry Pi ပေါ်တွင် run ခြင်း။

## J.၂၀ QEMU - Quick EMUlator

ARM architecture ကို emulate လုပ်ပြီး arm program တွေကို amd64 ပေါ်မှာ run ကြည့်ဖို့ အတွက် QEMU - <https://www.qemu.org/> ကို သုံးပါ မယ်။ သူ့ကို တပ်ဆင် ဖို့ အတွက် အောက်ပါ အတိုင်း ရိုက်ထည့် ပါမယ်။

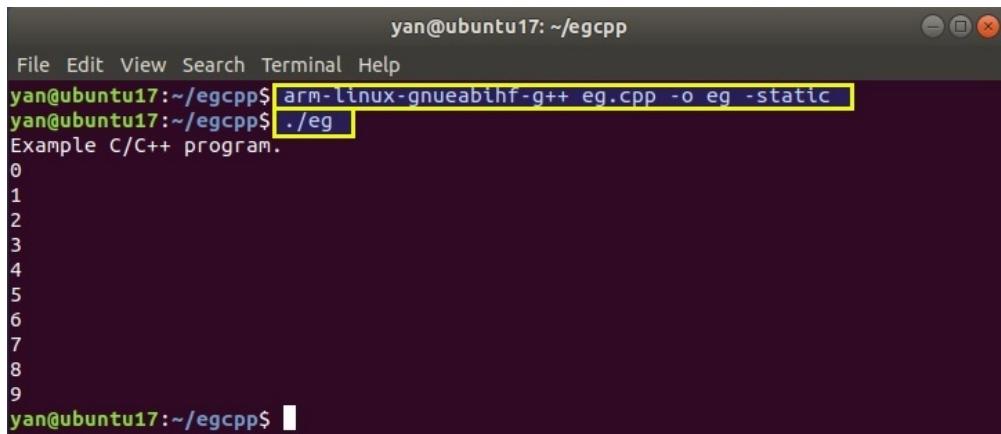
```
$ sudo apt-get install qemu
```

QEMU ထည့်ပြီး တဲ့ အခါ eg.cpp ကိုပဲ အောက်က အတိုင်း -static အနေနဲ့ compile လုပ်ပြီး တဲ့အခါ desktop ကွန်ပျူးတာ ပေါ်မှာ လည်း run ကြည့်လို့ ရသွား တာကို တွေ့ရ ပါမယ် (ပုံ J.၄၂)။

```
$ arm-linux-gnueabihf-g++ eg.cpp -o eg -static
$ ./eg
```

## J.၁၁. CODE::BLOCKS IDE တပ်ဆင်ခြင်း

၅၁



```
yan@ubuntu17:~/egcpp$ arm-linux-gnueabihf-g++ eg.cpp -o eg -static
yan@ubuntu17:~/egcpp$ ./eg
Example C/C++ program.
0
1
2
3
4
5
6
7
8
9
yan@ubuntu17:~/egcpp$
```

ံ J.၄၂: Emulator ဖြင့် run ခြင်း။

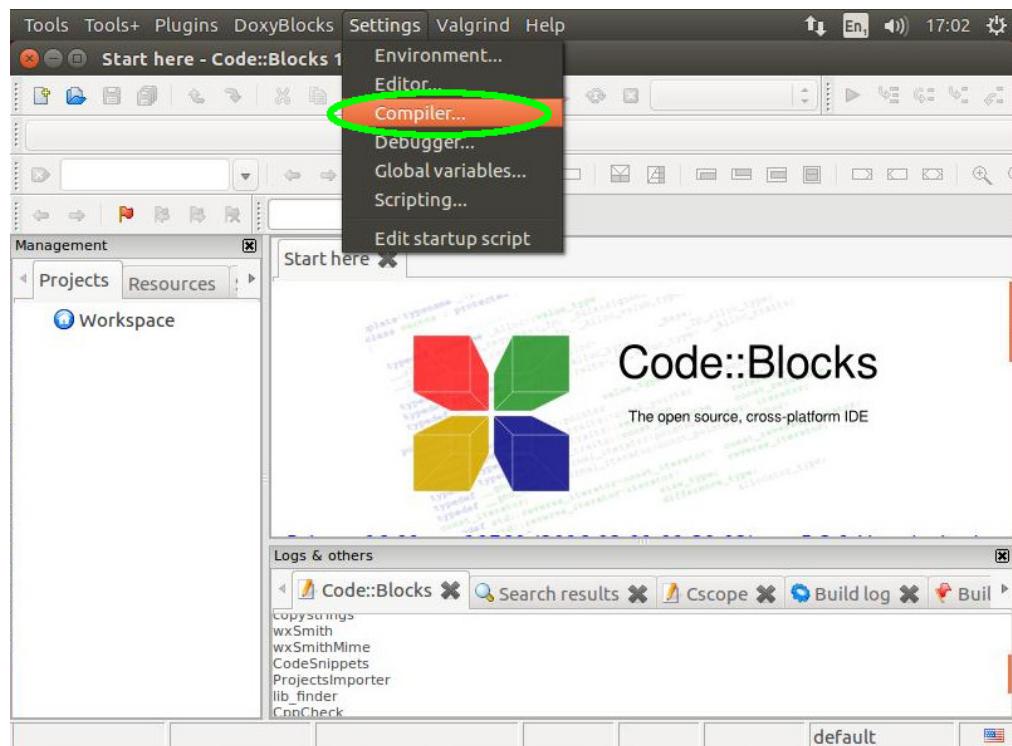
## J.၁၃ Code::Blocks IDE တပ်ဆင်ခြင်း

C/C++ ပရိုဂရမ် တွေ ရေးသားဖို့ အတွက် Code::Blocks ဆိုတဲ့ open source နဲ့ free ဖြစ်တဲ့ IDE ကို သုံးပါမယ်။ သူက extension တွေလည်းများ၊ စိတ်ကြိုက် configure လည်း လုပ်နိုင်ပြီး တော်တော် လည်း ခေတ်စား တဲ့ IDE တစ်ခု ပါ။ Cross platform GUI application တွေ အတွက် wxWidgets ကို သုံးမယ် ဆိုရင် လည်း အဆင်ပြေ လွယ်ကူ ပါတယ်။ Code::Blocks IDE ကို တပ်ဆင် ဖို့ အတွက် terminal မှာ စာရင်း [J.၇](#) အတိုင်း ရိုက်ထည့် နိုင် ပါတယ်။

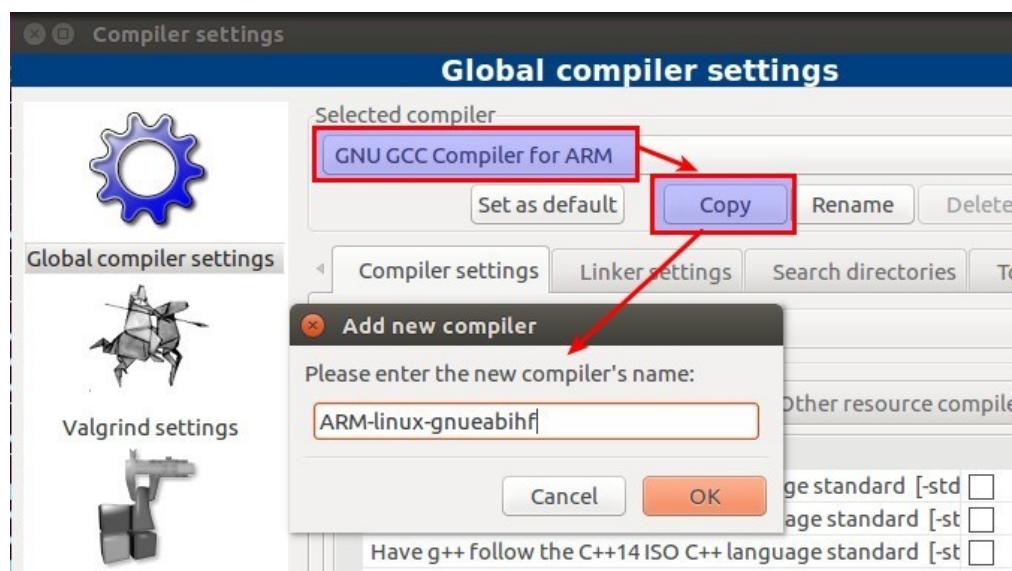
```
1 $ sudo apt-get install build-essential
2 $ sudo apt-get install gdb
3 $ sudo apt-get install libwxgtk3.0
4 $ sudo add-apt-repository ppa:damien-moore/codeblocks-stable
5 $ sudo apt-get update
6 $ sudo apt-get install codeblocks codeblocks-contrib
```

စာရင်း J.၇: Code Blocks တပ်ဆင်ခြင်း။

Code::Blocks ကို တပ်ဆင် ပြီးတဲ့ အပါ သူကို ဖွင့်လိုက်ပြီး ပံ့ J.၄၃ မှာ ပြထား သလို Settings menu → Compiler... ကို ဖွင့် လိုက် ပါမယ်။



ပုံ၂၂၃: Code::Blocks IDE တပ်ဆင် ပြင်ဆင်မှု။



ပုံ၂၂၄: Code::Blocks စုံင် Compiler အတွက် ပြင်ဆင်ခြင်း။

Compiler settings ဝင်းရှိုးပေါ်လာ တဲ့အခါ Global Compiler Settings ရဲ့ selected compiler မှာ GNU GCC Compiler for ARM ကို ရွေးလိုက် ပြီး၊ copy ကို နိုင်လိုက် ပါမယ်။ Compiler အသစ် ရဲ့ နာမည်ကို ARM-linux-gnueabihf အစ ရှိတဲ့ နာမည် တစ်ခု ပေးနိုင် ပါတယ် (ပုံ J.၄၄)။

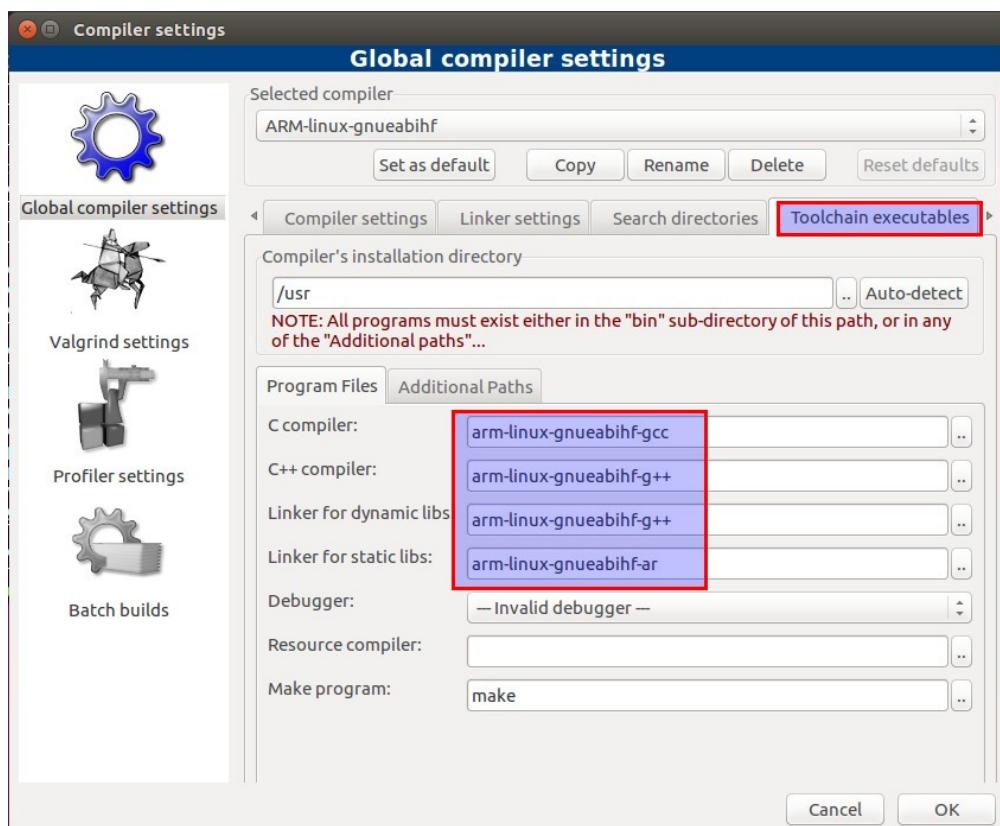
ပြီးတဲ့ အခါ Toolchain executables ဆိုတဲ့ tab ကို သွားပြီး၊ Program Files ဆိုတဲ့ tab မှာ အောက်က စာရင်း J.၈ နဲ့ ပုံ J.၄၅ မှာ ပြထား သလို အသီးသီး ဖြည့်နိုင် ပါတယ်။

```

1 arm-linux-gnueabihf-gcc
2 arm-linux-gnueabihf-g++
3 arm-linux-gnueabihf-g++
4 arm-linux-gnueabihf-ar

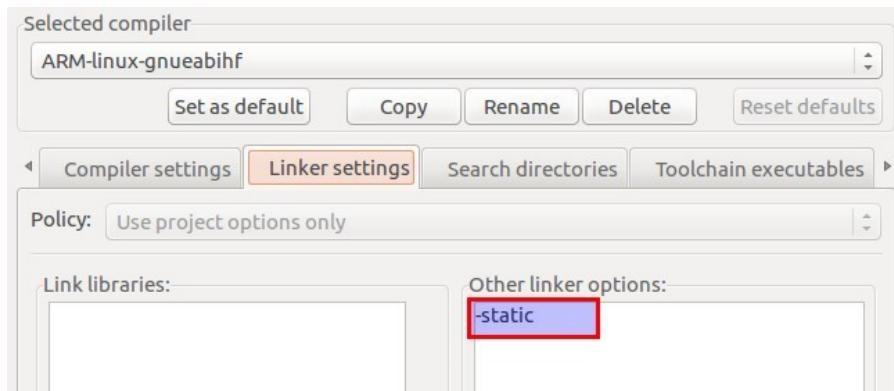
```

စာရင်း J.၈: Code Blocks တွင် compiler သတ်မှတ်ခြင်း။

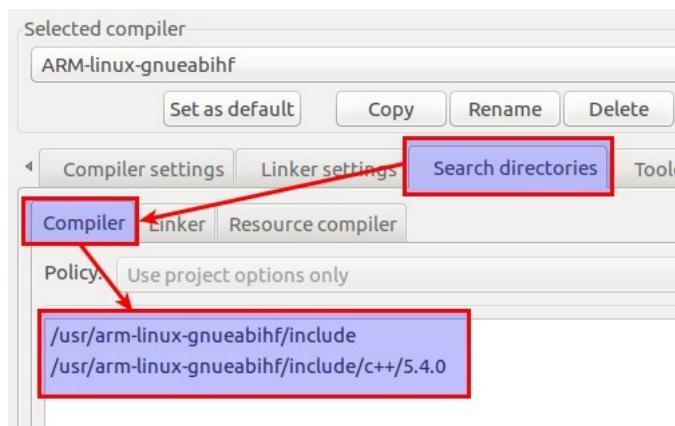


ပုံ J.၄၅: Code::Blocks တွင် Compiler program ဖိုင်များ သတ်မှတ်ခြင်း။

အဲဒီ နောက် Linker settings ဆိုတဲ့ tab ရဲ့ other linker options မှာ -static လို့ ဖြည့်လိုက် ပါမယ်။



ပုံ J.၄၆: Code::Blocks တွင် linker option သတ်မှတ်ခြင်း။



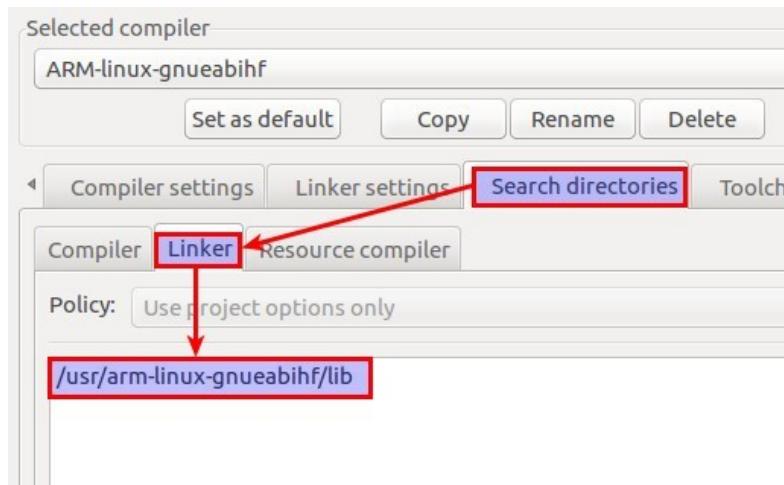
ပုံ J.၄၇: Code::Blocks တွင် include paths များ သတ်မှတ်ခြင်း။

နောက် တစ်ခါ ပုံ J.၄၇ မှာ ပြထား သလို Search directories ဆိုတဲ့ tab ထဲက compiler မှာ

```
/usr/arm-linux-gnueabihf/include
/usr/arm-linux-gnueabihf/include/c++/5.4.0
```

တိုကို ဖြည့်လိုက် ပါမယ်။

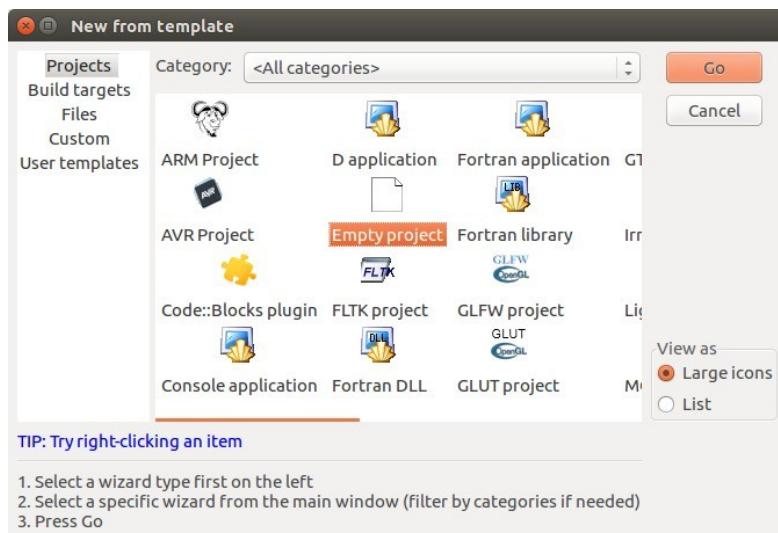
ခုနက လိုပဲ ပုံ J.၄၈ မှာ ပြထား သလို Search directories ဆိုတဲ့ tab ထဲက linker မှာ /usr/arm-linux-gnueabihf/lib ကို ဖြည့်လိုက် ပါမယ်။



ပုံ J.၄၈: Code::Blocks တွင် lib path သတ်မှတ်ခြင်း။

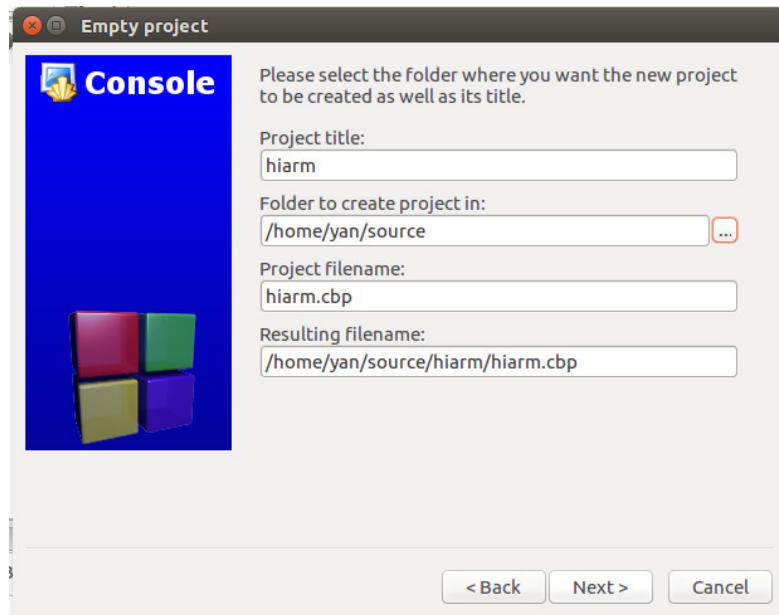
### J.၁၁.၁ Code::Blocks Project နမူနာ

Code::Blocks ထဲမှာ File menu → New → Project... ကို နိုင်ပြီး project အသစ် တစ်ခု ဖန်တီးလိုက်ပါမယ်။ ပေါ်လာ တဲ့ ဝင်းချွဲး မှာ Empty Project ကို ရွေးပြီး Go ကိုနိုင်ပြီးရင် Next ကို နိုင်ပြီး ရွေ့ဆက်သွား ပါမယ် (ပုံ J.၄၉)။

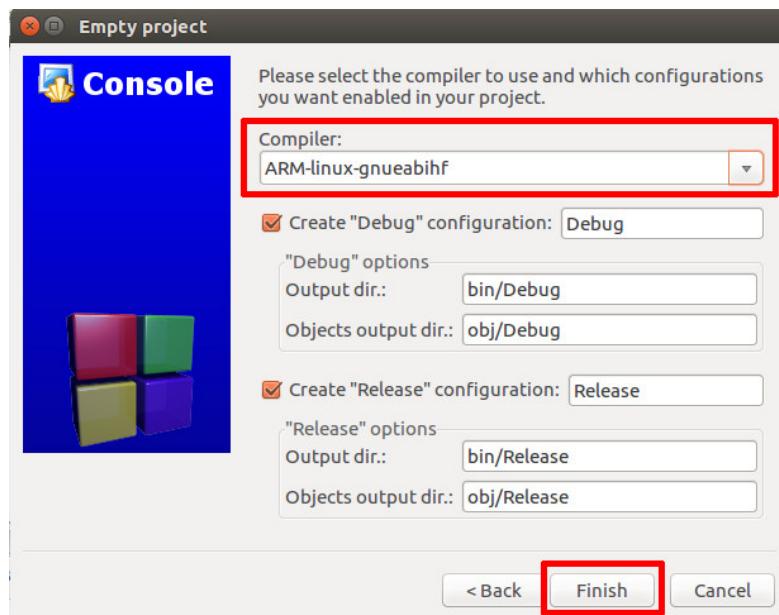


ပုံ J.၄၉: Code::Blocks တွင် Project တစ်ခု ဖန်တီး ခြင်း။

ပြီးတဲ့ အခါ ပရောဂျက် နာမည်၊ နေရာ တွေကို သတ်မှတ် ပြီးရင် Next ကို ထပ်နှိပ် ပါမယ် (ပုံ J.၅၀)။



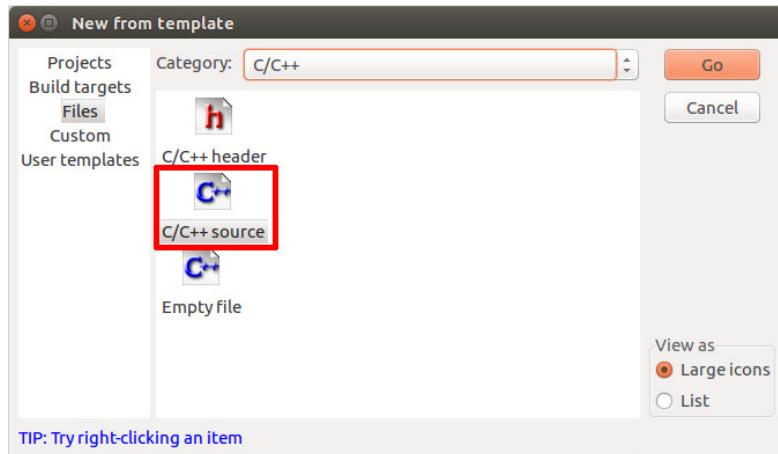
ပုံ J.၅၀: Project နာမည် နှင့် နေရာ များ သတ်မှတ် ခြင်း။



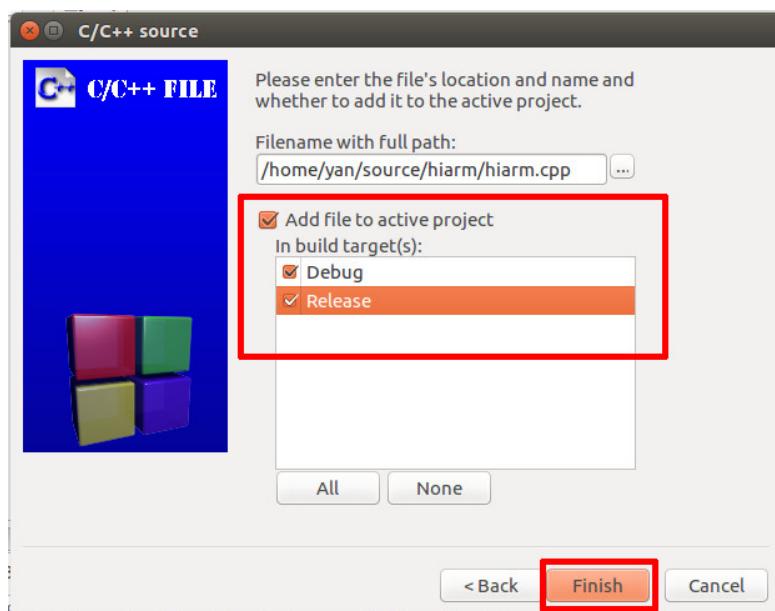
ပုံ J.၅၁: Project နာမည် နှင့် နေရာ များ သတ်မှတ် ခြင်း။

နောက် အဆင့် အနေနဲ့ compiler ကို ရွေးတဲ့ အခါ ပုံ J.၅၁ မှာ ပြထားသလို ခန်က အပိုင်းမှာ ဖန်တီး ခဲ့တဲ့ ARM-linux-gnueabihf ကို ရွေးလိုက် ပြီး Finish ခလုတ် ကို နှုပ်နိုင် ပါတယ်။

ပရောဂျက် ဖန်တီး လို ရလာ ပြီးတဲ့ အခါ File menu → New → File... ကို နှုပ်ပြီး ဖိုင် အသစ် တစ်ခု ဖန်တီး လိုက်ပြီး ပုံ J.၅၂ နဲ့ ပုံ J.၅၃ မှာ အဆင့်ဆင့် ပြထားသလို Active project ရဲ့ target တွေထဲ ကို ထည့်လိုက် ပါမယ်။



ပုံ J.၅၂: C++ ဖိုင် အသစ် ဖန်တီး ခြင်း။



ပုံ J.၅၃: Active project နှင့် target များ တွင်ထည့်ခြင်း။

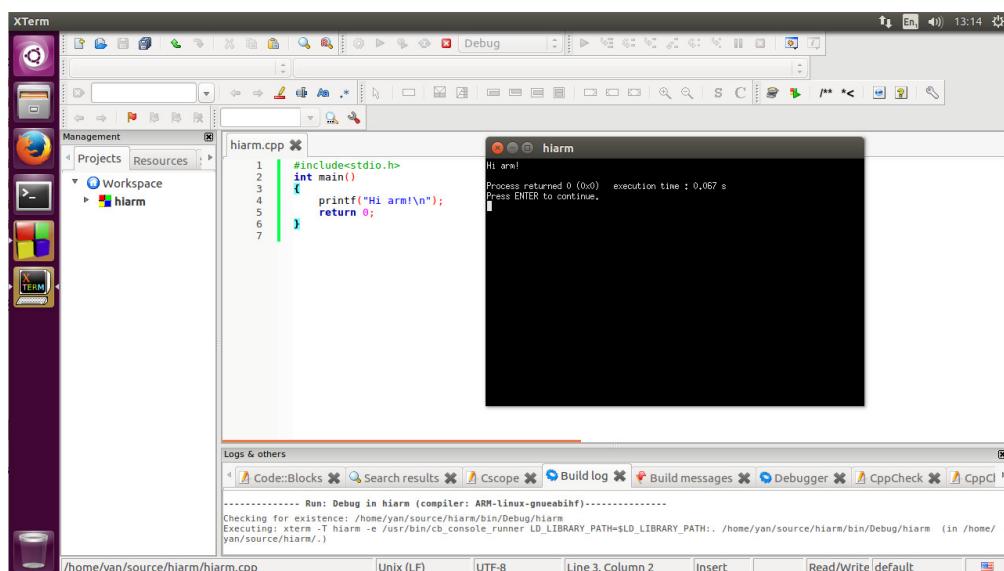
အဲဒီနာက် မှာ စာရင်း [၂.၉](#) မှာ ပြထားတဲ့ စမ်းသပ် ကြည့်ဖို့ နမူနာ C/C++ ပရိုဂရမ် လေး ရေးလိုက် ပါမယ်။ ပြီးတဲ့ အခါ F9 ခလုပ် နိုဝင်ပြီး Build and run လုပ်လိုက်ရင် ပုံ [၂.၅၄](#) မှာ ပြထား တဲ့ အတိုင်း ပရိုဂရမ် ရဲ့ output ကို တွေ့ နှင့် ပါတယ်။ နာက် တဆင့် အနေနဲ့ ပရောဂျက် အခန်းထဲက build လုပ်လိုက် တဲ့ configuration ပေါ် မူတည် ပြီး bin/Debug ဒါမှ မဟုတ် bin/Release အခန်း ထဲက executable application ကို Raspberry Pi ပေါ် ကူးထည့် ပြီး run ကြည့်တဲ့ အခါ မှာလည်း ရလဒ် တူတာ ကို တွေ့နှင့် ပါတယ်။

```

1 #include<stdio.h>
2 int main()
3 {
4     printf("Hi arm!\n");
5     return 0;
6 }
7

```

စာရင်း ၂.၉: Code::Blocks ဘွင် စမ်းသပ်ရန် နမူနာ C/C++ ပရိုဂရမ်။



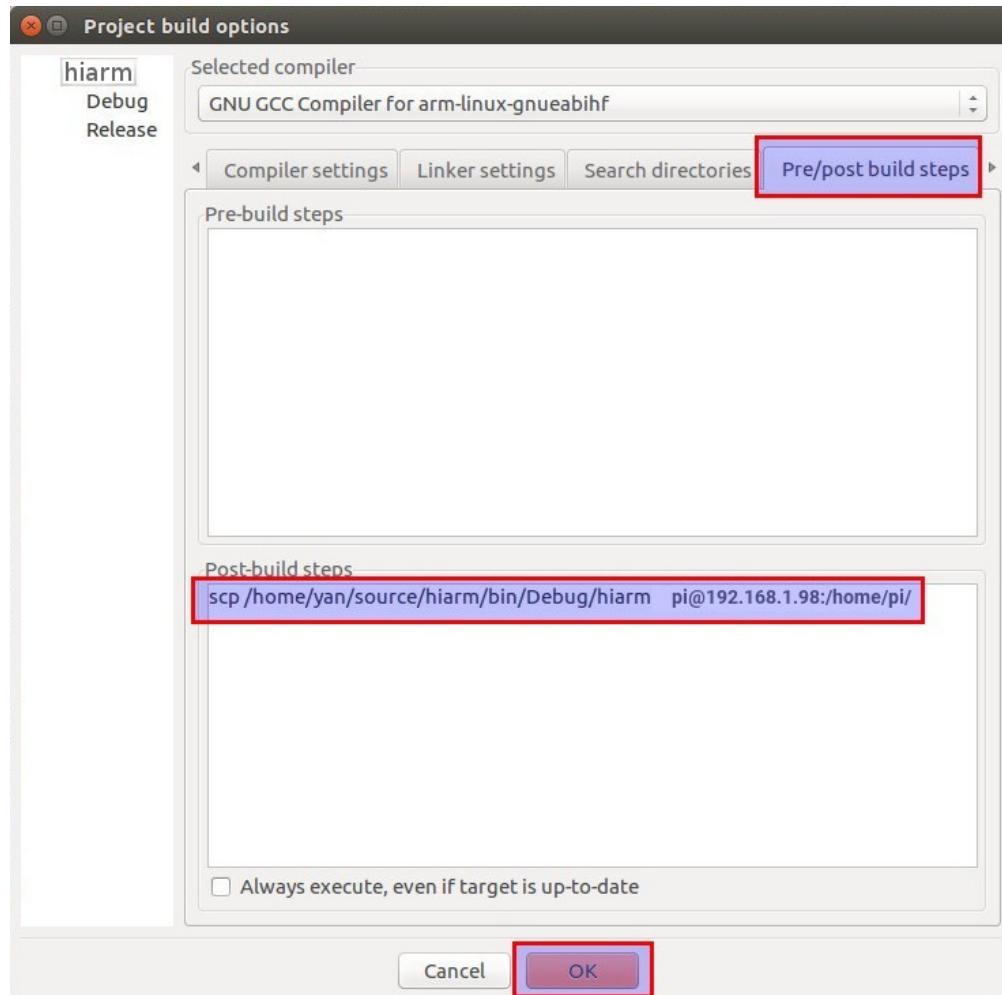
ဦး၂.၅၄: Build and Run ပြလုပ်ခြင်း။

Program ကို Build လုပ်ပြီး တိုင်း Raspberry Pi ပေါ်ကို အလို အလျောက် ကူးပေး စေခဲ့ရင်

Project menu → Build Options... ကို နှိပ်ပြီး Pre/post build steps tab အောက် ၂ Post-build steps များ

```
scp /home/yan/source/hiarm/bin/Debug/hiarm pi@192.168.1.98:/home/pi/
```

စတဲ့ command တွေကို ဖြည့်ထားနိုင် ပါတယ် (ပုံ J.၅၅)။



ပုံ J.၅၅: Post-build step များ သတ်မှတ်ခြင်း။

## အကိုးအကားများ

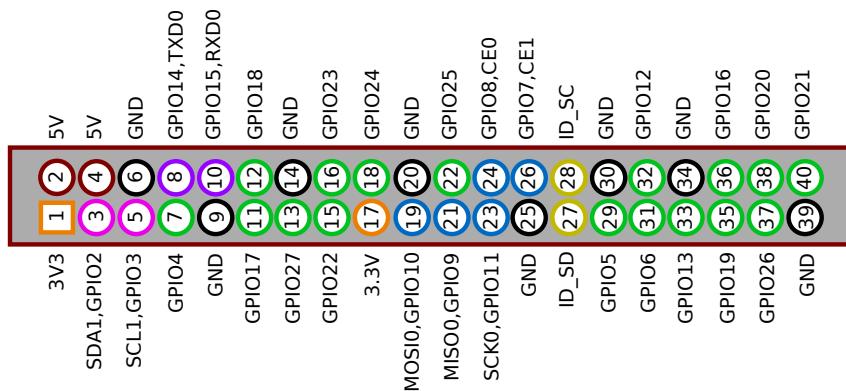
- [Bol13] Evan Boldt. BeagleBone Internet over USB only. 2013. url: <http://robotic-controls.com/learn/beaglebone/beaglebone-internet-over-usb-only>.
- [Cyn15] Cynic. How to Connect a Beaglebone Black to the Internet via USB. 2015. url: <https://ofitselso.com/BeagleNotes/HowToConnectBeagleboneBlackToTheInternetViaUSB.php>.
- [Eli13] Elinux. Beagleboard: C/C++ Programming. 2013. url: [http://elinux.org/Beagleboard:C/C%2B%2B\\_Programming](http://elinux.org/Beagleboard:C/C%2B%2B_Programming).
- [Mon15] Simon Monk. SSH with Windows and PuTTY. 2015. url: <https://learn.adafruit.com/ssh-to-beaglebone-black-over-usb/ssh-with-windows-and-putty>.
- [Wik17b] Bebian Wiki. Multiarch HowTo. 2017. url: <https://wiki.debian.org/Multiarch/HOWTO>.

## အခန်း ၃

# Input/Output များ

### ၃.၁ Digital IO

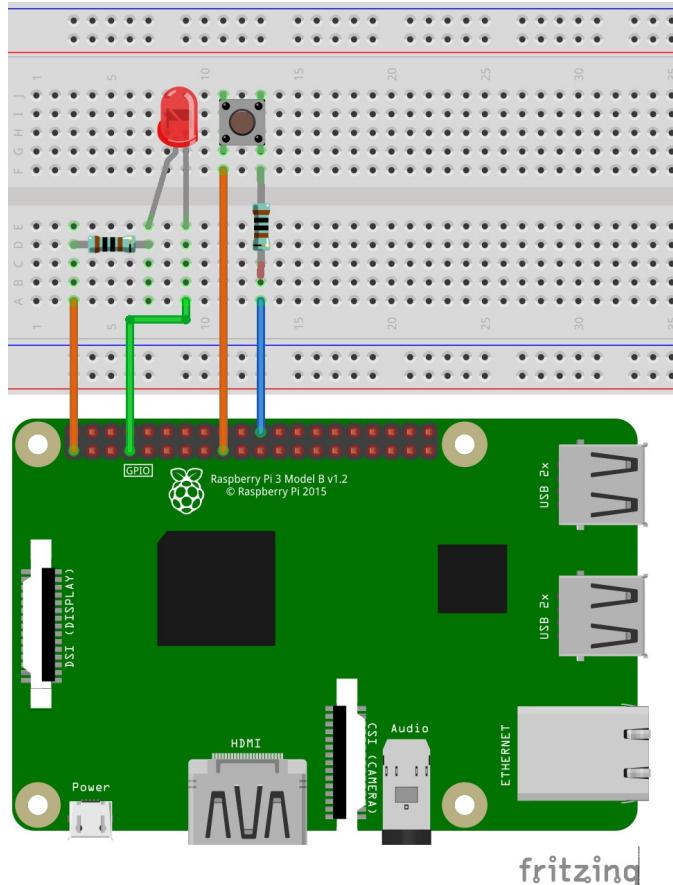
GPIO တွေကို ထိန်းချုပ် တဲ့ နယူနာ အနေနဲ့ Raspberry Pi ရဲ့ GPIO header မှာ ရှိတဲ့ pin တွေကို အသုံး ဖြော်ပြီး LED မီးသီး တစ်လုံး ကို digital output အနေနဲ့ ထိန်းချုပ် တာရယ်၊ ခလုတ် တစ်ခု နိုပ်မ နှင့် ကို digital input အနေနဲ့ ဖတ်တာရယ် ကို လုပ်ဆောင် ကြည့် ပါမယ်။ ပုံ ၃.၁ မှာ Raspberry Pi ရဲ့ 40 pin GPIO header ကို ဖော်ပြ ထားပြီး GPIO header ကို အောက်ဘက် က ကြည့်လိုက်ရင် လေးထောင့်ပုံ ဖြစ်နေ တဲ့ pin က နံပါတ်တစ် ဖြစ် ပါတယ်။



ပုံ ၃.၁: Raspberry Pi ၏ GPIO header ရှိ pin များ ကို အပေါ်စီး မှ မြင်ရပုံ။

Digital input နဲ့ output တွေကို စမ်းသပ် အသုံးပြု ကြည့်ဖို့ အတွက် LED မီးလုံး တစ်လုံး ကို pin

7 | push button တစ်ခုကို pin 22 နဲ့ ပုံ ၃.၂ မှာ ပြထားသလို ဆက်သွယ်လိုက် ပါမယ်။ ပုံ ၃.၁ မှာ ကြည့်လိုက်တဲ့ အခါ pin 7 က GPIO4 ဖြစ်ပြီး pin 22 က GPIO25 ဖြစ်တာ ကို တွေ့ရမှာ ဖြစ် ပါတယ်။



ပုံ ၃.၂: Digital input/output များ စမ်းသပ်ရန် ဆက်သွယ်ပုံ။

အသုံးပြုဖို့ အဆင့် သင့်ရှုတဲ့ pin တွေကို စစ်ကြည့်ဖို့ အတွက် စက်ရဲ့ /sys/class/gpio အခန်း ထဲကို သွားကြည့် နိုင် ပါတယ်။ ပုံမှန် အားဖြင့် GPIO တွေက disable ဖြစ်နေ တာမို့ သူတို့ ကို စာရင်း ၃.၁ က အတိုင်း export လုပ်ပြီး enable လုပ်ဖို့ လို ပါတယ်။ ဥပမာ GPIO\_4 ကို enable လုပ်ချင် ရင် /sys/class/gpio အခန်း ထဲက export ဆိုတဲ့ ဖိုင်ထဲကို 4 ရေးထည့် လိုက်တာပါ။

```
$ cd /sys/class/gpio
$ ls
$ echo 4 > export
$ ls
```

Enable လုပ်ပြီးတဲ့ အခါ /sys/class/gpio အခန်းထဲမှာ enable လုပ်လိုက်တဲ့ gpio ရဲ့ နာမည်နဲ့ အခန်း တစ်ခု ပေါ်လာတာ တွေ့နိုင် ပါတယ်။ ပြီးတဲ့ အခါ GPIO တွေရဲ့ direction ကို အဝင် သိမဟုတ် အထွက် စသဖြင့် သတ်မှတ် နိုင်ပါ တယ်။ အဲဒီ အထွက် သူရဲ့ အခန်းထဲက direction ဆိုတဲ့ ဖိုင်ထဲကို in သိမဟုတ် out ကို ရေးနိုင် ပါတယ်။ ဥပမာ GPIO\_4 ကို output သတ်မှတ် မယ် ဆိုရင် gpio4 အခန်းထဲက direction ဆိုတဲ့ ဖိုင်မှာ out ကို ရေးနိုင် ပါတယ်။

```
$ cd gpio4
$ echo out > direction
```

အဲဒီ နောက်မှာ အဝင် ဆိုရင် သူရဲ့ တန်ဖိုးကို value ဆိုတဲ့ ဖိုင်မှာ ဖတ်ကြည့် နိုင်ပြီး၊ အထွက် ဆိုရင် 0 သိမဟုတ် 1 တန်ဖိုး တွေကို value ဆိုတဲ့ ဖိုင်မှာ ရေးထည့်ပြီး output ထဲတ် ပေးနိုင် ပါတယ်။ စာရင်း ၃.၁ က command တွေကို ပုံ ၃.၃ က အတိုင်း ထည့်ပြီး စမ်းသပ် ကြည့်တဲ့ အခါ LED မီးလုံး အဖွင့် အပိတ် ဖြစ်သွား တာကို တွေ့နိုင် ပါတယ်။

```
1 $ cd /sys/class/gpio
2 $ ls
3 $ echo 4 > export
4 $ ls
5 $ cd gpio4
6 $ ls
7 $ more direction
8 $ echo out > direction
9 $ more direction
10 $ more value
11 $ echo 1 > value
12 $ echo 0 > value
```

စာရင်း ၃.၁: GPIO များကို ကြည့်ခြင်း။

```

pi@raspberrypi:~ $ cd /sys/class/gpio
pi@raspberrypi:/sys/class/gpio $ ls
export gpiochip0 gpiochip100 gpiochip128 unexport
pi@raspberrypi:/sys/class/gpio $ echo 4 > export
pi@raspberrypi:/sys/class/gpio $ ls
export gpio4 gpiochip0 gpiochip100 gpiochip128 unexport
pi@raspberrypi:/sys/class/gpio $ cd gpio4
pi@raspberrypi:/sys/class/gpio/gpio4 $ ls
active_low device direction edge power subsystem uevent value
pi@raspberrypi:/sys/class/gpio/gpio4 $ more direction
in
pi@raspberrypi:/sys/class/gpio/gpio4 $ echo out > direction
pi@raspberrypi:/sys/class/gpio/gpio4 $ more direction
out
pi@raspberrypi:/sys/class/gpio/gpio4 $ more value
0
pi@raspberrypi:/sys/class/gpio/gpio4 $ echo 1 > value
pi@raspberrypi:/sys/class/gpio/gpio4 $ echo 0 > value

```

ပုံ ၃.၃: GPIO pin ကို Linux shell မှ ကြည့်ရှု ထိန်းချုပ်ခြင်း။

Enable လုပ်ထားတဲ့ GPIO တစ်ခု ကို ပြန်ပြီး ဖယ်ချင် ရင်တော့ unexport ကို အောက်က အတိုင်း သုံးနိုင်ပါတယ်။

```
echo 4 > unexport
```

### ၃.၁.၁ C++ ဖြော်သုံးခြင်း

C++ နဲ့ GPIO တွေကို ထိန်းချုပ် ဖို့ အတွက် ရိုးရှင်းတဲ့ နမူနာ class လေး တစ်ခု ကို `ce_io.h` (စာရင်း ၃.၂) မှာ ပြထား ပါတယ်။ စာရင်း ၃.၁ က device file တွေကို ရေးတဲ့ ဖတ်တဲ့ အလုပ် ကို C++ ရဲ့ file stream နဲ့ ပြောင်းပြီး ရေးတာ၊ ဖတ်တာ လုပ် လိုက်တာ ပါပဲ။

```

1 //File: ce_io.h
2 //Description: Controlling GPIO pins
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye
6
7 #ifndef CE_IO_H
8 #define CE_IO_H
9

```

```
10 #include <string>
11 #include <fstream>
12 #include <sstream>
13 #define IOPATH "/sys/class/gpio/"
14 using namespace std;
15 typedef enum {INPUT=0,OUTPUT=1} iodir;
16 typedef enum {LOW=0,HIGH=1} digitalval;
17
18 class CE_IO{
19     string fpath;
20     int gpioNumber;
21 public:
22     CE_IO();
23     CE_IO(int gpiono,iodir mode);
24     ~CE_IO();
25     int Begin(int gpiono,iodir mode);
26     digitalval Read();
27     int Write(digitalval b);
28     int Export();
29     int SetDirection(iodir mode);
30     template <typename T>
31         string ToString(T Number);
32 };
33
34 template <typename T>
35 string CE_IO::ToString(T a)
36 {
37     ostringstream ss;
38     ss << a;
39     return ss.str();
40 }
41
42 CE_IO::CE_IO()
43 {
44 }
```

```
46
47 CE_IO::CE_IO(int gpiono,iodir mode)
48 {
49     Begin(gpiono,mode);
50 }
51
52 int CE_IO::Begin(int gpiono,iodir mode)
53 {
54     gpioNumber=gpiono;
55     fpath = IOPATH;
56     fpath+="gpio"+ToString(gpiono)+"/value";
57     if(Export()<0) return -1;
58     if(SetDirection(mode)<0) return -1;
59     return 0;
60 }
61
62 int CE_IO::Export()
63 {
64     ofstream wfile;
65     int r=-1;
66     string epath=IOPATH;
67     epath+="export";
68     wfile.open(epath.c_str());
69     if (wfile.is_open()) {
70         wfile << gpioNumber;
71         r=0;
72     }
73     wfile.close();
74     return r;
75 }
76
77 int CE_IO::SetDirection(iodir mode)
78 {
79     ofstream wfile;
80     int r=-1;
81     string dirpath=IOPATH;
```

```
82     dirpath+="gpio"+ToString(gpioNumber) +"/direction";
83     wfile.open(dirpath.c_str());
84     string modestr[2]={"in","out"};
85     if (wfile.is_open()) {
86         wfile << modestr[mode];
87         r=0;
88     }
89     wfile.close();
90     return r;
91 }
92
93
94 CE_IO::~CE_IO()
95 {
96     ofstream wfile;
97     string epath=IOPATH;
98     epath+="unexport";
99     //unexport the gpio
100    wfile.open(epath.c_str());
101    if (wfile.is_open()) {wfile << gpioNumber;}
102    wfile.close();
103 }
104
105 digitalval CE_IO::Read()
106 {
107     ifstream rfile;
108     string str;
109     digitalval rdv=LOW;
110     rfile.open(fpath.c_str());
111     if (rfile.is_open()) {
112         getline(rfile,str);
113     }
114     else {
115         str="";
116     }
117     rfile.close();
```

```

118     if(str=="1") rdv=HIGH;
119
120 }
121
122 int CE_IO::Write(digitalval b)
123 {
124
125     ofstream wfile;
126
127     int r=-1;
128
129     wfile.open(fpath.c_str());
130
131     if (wfile.is_open()) {
132
133         wfile << b;
134
135         r=0;
136
137     }
138
139     wfile.close();
140
141     return r;
142
143 }
144
145 #endif

```

စာရင်း ၃.J: ce\_io.h

အဲဒီ class ကို အသုံးပြုတဲ့ C++ ပရိုဂရမ် နမူနာ gpiomain.cpp ကို စာရင်း ၃.၃ မှာ ပြထား ပါတယ်။ ပုံ ၃.J က ဆက်သွယ်မှု ပုံစံ အတိုင်း ၁ စက္ကန် တစ်ခါ Pin22 ရဲ့ အဝင်ကို ဖတ်ပေး ပြီး၊ Pin7 ရဲ့ အထွက် LED မီးလုံးကို မိတ်တူတဲ့ မိတ်တူတဲ့ ပြုလုပ် ပေးပါမယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <string>
4 #include "ce_io.h"
5
6 using namespace std;
7
8 int main()
9 {
10
11     CE_IO Pin7(4, OUTPUT);
12     CE_IO Pin22(25, INPUT);
13
14     for(int i=0; i<5; i++) {
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
978
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1088
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1245
1246
1247
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1256
1257
1258
1258
1259
1260
1261
1262
1263
1264
1264
1265
1266
1266
1267
1268
1268
1269
1269
1270
1271
1271
1272
1273
1273
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1281
1281
1282
1283
1283
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1291
1291
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1590
1591
1591
1592
1592
1593
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1600
1601
1601
1602
1602
1603
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1610
1611
1611
1612
1612
1613
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1620
1621
1621
1622
1622
1623
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1630
1631
1631
1632
1632
1633
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1640
1641
1641
1642
1642
1643
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1650
1651
1651
1652
1652
1653
1653
1654
1654
1655
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1660
1661
1661
1662
1662
1663
1663
1664
1664
1665
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1670
1671
1671
1672
1672
1673
1673
1674
1674
1675
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1680
1681
1681
1682
1682
1683
1683
1684
1684
1685
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1690
1691
1691
1692
1692
1693
1693
1694
1694
1695
1695
1696
1696
1697
1697
1698
1698
1699
1699
1700
1700
1701
1701
1702
1702
1703
1703
1704
1704
1705
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1710
1711
1711
1712
1712
1713
1713
1714
1714
1715
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1720
1721
1721
1722
1722
1723
1723
1724
1724
1725
1725
1726
1726
1727
1727
1728
1728
1729
1729
1730
1730
1731
1731
1732
1732
1733
1733
1734
1734
1735
1735
1736
1736
1737
1737
1738
```

```

12     printf("i=%d\n", i);

13

14 //Read input
15 printf("Input= %d \n", Pin22.Read());

16

17 //Write output
18 Pin7.Write(HIGH);
19 printf("Output = 1");
20 usleep(500000);
21 Pin7.Write(LOW);
22 printf(" > 0\n");
23 usleep(500000);

24 }

25 return 0;
26 }
```

စာရင်း ၃.၃: gpiomain.cpp

ပရိုဂရမ် ကို စာရင်း ၃.၄ အတိုင်း Build လုပ်ပြီး Run နိုင် ပါတယ်။ ပရိုဂရမ် ရဲ့ output ကို ပုံ ၃.၄ မှာ ထွေးနိုင် ပါတယ်။

```

pi@raspberrypi:~/gpio $ g++ gpiomain.cpp ce_io.cpp -o gpiomain
pi@raspberrypi:~/gpio $ sudo ./gpiomain
i=0
Input= 0
Output = 1 > 0
i=1
Input= 1
Output = 1 > 0
i=2
Input= 1
Output = 1 > 0
i=3
Input= 1
Output = 1 > 0
i=4
Input= 0
Output = 1 > 0
pi@raspberrypi:~/gpio $
```

ပုံ ၃.၄: GPIO များကို ထိန်းချုပ်သည့် C++ ပရိုဂရမ်။

```

1 $ g++ gpiomain.cpp -o gpiomain
2 $ sudo ./gpiomain
```

စာရင်း ၃.၅: GPIO များကို C++ ဖြင့် ထိန်းချုပ် သည့် ပရိုဂရမ် ကို run ခြင်း။

### ၃.၁.၂ Light sensor ဖြင့်အလင်းရောင်ကိုအာရုံခံခြင်း

Digital Input ကနေ အလင်းရောင် ကို အာရုံခံပြီး၊ သတ်မှတ် ထားတဲ့ အတိုင်း အတာ ထက် မောင်လာ ရင် မီး ခလုတ် ကို အလို အလျောက် အဖွင့် အပိတ် လုပ်ပေး တဲ့ စနစ်လေး တစ်ခု တည်ဆောက် ကြည့် ပါမယ်။ အဲဒီ အတွက် [AliExpress](#) ကနေ ဈေး ပေါပေါနဲ့ အလွယ် အကူ ဝယ်လို့ ရတဲ့ LM393 4 pin light sensor module လေးကို သုံးလိုက် ပါမယ်။

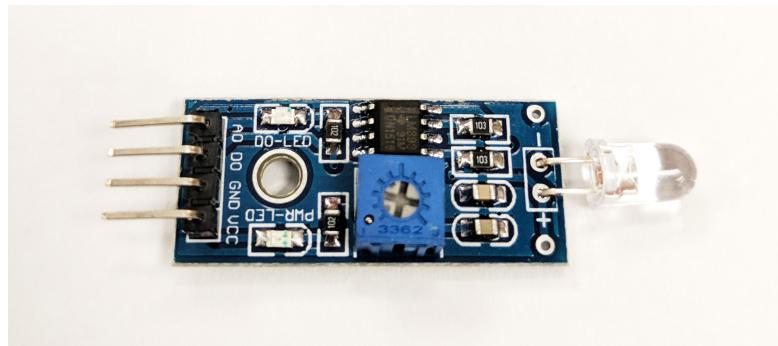
အဲဒီ module က 3.3 V နဲ့ ရော 5 V နဲ့ပါ သုံးလို့ ရပါတယ်။ သူရဲ့ pin လေးခုက အောက်ပါ အတိုင်း ဖြစ်ပါတယ်။

**Vcc:** Power supply pin 3.3 V to 5 V.

**Gnd:** Ground.

**Do:** Digital output.

**Ao:** Analog output.



ပုံ ၃.၅: LM393 4 pin light sensor module

Analog output pin က အလင်းရောင် အနည်း အများ ပေါ်မှ တည်ပြီး ဖို့ အနည်း အများ အချိုးကျ ထုတ်ပေး ပါတယ်။ မောင် လာရင် ဖို့များ လာပြီး၊ လင်းလာ ရင် ဖို့ နည်း လာပါတယ်။ ပုံ ၃.၅ မှာ ပြထား သလို trimmer လေး ပါတဲ့ အတွက်၊ သူကို လှည့်ပြီး digital output ထုတ် ပေးမယ့် အလင်း ရောင် ပမာဏ ကို သတ်မှတ် ချိန်ညို ပေးနိုင် ပါတယ်။ ပါဝါ အတွက် အနီး ရောင် LED မီးလုံး ပါပြီး၊ digital output အတွက် အစိမ်းရောင် LED မီးလုံး ပါတဲ့ အတွက် ချိန်တဲ့ အခါ အဆင်ပြု လွယ်ကူ ပါတယ်။ Trimmer နဲ့ ချိန်ထား တဲ့ အလင်း ရောင်ထက် နည်းသွား တာနဲ့ Do မှာ high ဖြစ် သွားပြီး၊ LED အစိမ်း

ရောင်က မိတ်သွားပါမယ်။ အလင်းရောင် များလာ ရင် Do က low ဖြစ်သွားပြီး LED အစိမ်းလင်းလာပါလိမ့်မယ်။

ဒီ နှမူနာ မှာတော့ sensor ရဲ့ digital output ကိုပဲ သုံးမှာ ဖြစ်တဲ့ အတွက် သူကို RPi ရဲ့ အဝင် Pin22 (GPIO25) နဲ့ ဆက်သွယ် လိုက် ပါမယ်။

## ၃.၁.၃ Relay ဖြင့်ပါဝါအဖွင့်အပိတ်ထိန်းချုပ်ခြင်း

Relay တွေက ဗိုအား ပေးပြီး ထိန်းချုပ်လို့ ရတဲ့ ခလုတ် တွေလို့ ပြောလို့ ရပါတယ်။ Relay မှာ ဗိုအား မြင့် AC ပါဝါ ခလုတ် အနေနဲ့ သုံးနိုင် တဲ့ ငုတ်လေး တွေ ပါပိုတယ်။ ပုံမှန် ဆိုရင် ငုတ် သုံးခု ပါပြီး၊ အလယ်က ဘုံးငုတ် ကို COM (common) လို့ ခေါ်ပြီး၊ အဲဒီ COM နဲ့ ပုံမှန် အားဖြင့် ထိနေတဲ့ ငုတ် ကို NC (normally close) လို့ ခေါ်ပါတယ်။ COM ငုတ် နဲ့ ပုံမှန် အားဖြင့် လွတ်နေပြီး၊ activate လုပ်လိုက် မှ ထိသွားမယ့် ငုတ် ကိုတော့ NO (normally open) ငုတ် လို့ ခေါ်ပါတယ်။

နှမူနာ Relay module တစ်ခု ကို ပုံ ၃.၆ မှာ ပြထား ပါတယ်။ အဲဒီ Relay module ကို 5 V DC ဗိုအား ပေးထား နိုင်ပြီး၊ သူရဲ့ digital input ကို HIGH ဗိုအား အဝင် ပေးလိုက်ရင် အထဲမှာ ရှိတဲ့ လျှပ်စစ် သံလိုက် က သူရဲ့ COM ငုတ် နဲ့ ဆက်ထားတဲ့ သံပြား လေးကို ရွှေ့လိုက် တဲ့ အတွက် NC နဲ့ လွတ်သွားပြီး၊ NO နဲ့ ပြောင်းပြီး ထိသွား ပါတယ်။ အဲဒီနည်း ကိုသုံးပြီး သူတို့ နဲ့ ဆက်သွယ် ထားတဲ့ မီးလုံး၊ မီးခေါ်ပွား စတဲ့ လျှပ်စစ် ပစ္စည်း တွေ အတွက် ခလုတ် အနေနဲ့ ထိန်းချုပ် နိုင် ပါတယ်။



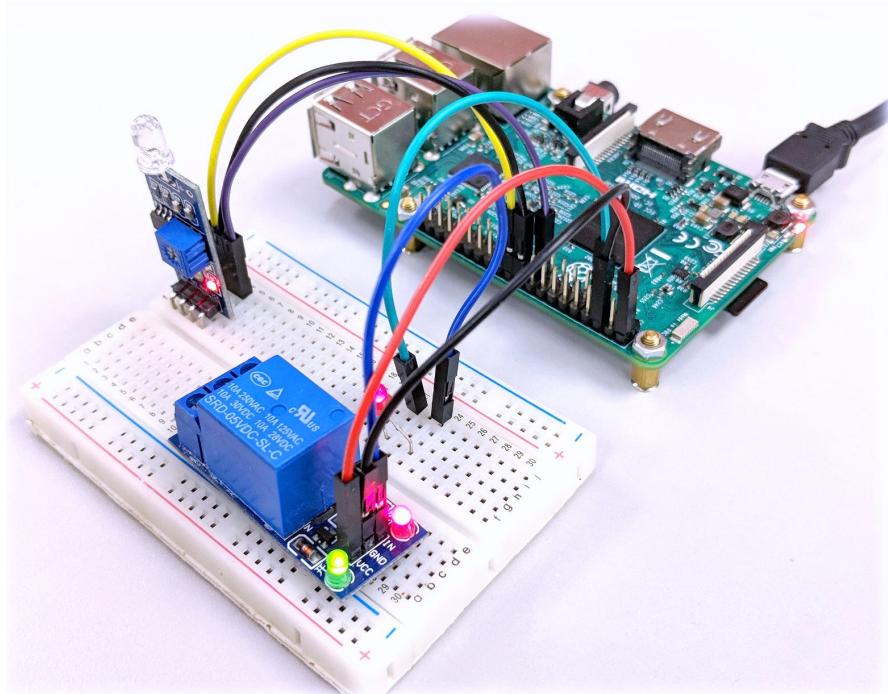
ပုံ ၃.၆: Relay

Relay ကို GND နဲ့ 5 V ပါဝါ ပေးပြီး တဲ့အခါ RPi ရဲ့ Pin7 (GPIO4) နဲ့ ဆက်သွယ် လိုက် ပါမယ်။ သုံးထား တဲ့ relay က 5 V ဖြစ်နေတဲ့ အတွက် ဆက်သွယ် တဲ့ အခါ RPi ရဲ့ IO မို့ 3.3 V နဲ့ အဆင် ပြေအောင်

ကြေားမှာ LED မီးလုံး လေး ကို Relay ဘက်မှာ anode ထားပြီး ခံ ဆက်ဖို့ လိုပါတယ်။

### ၃.၁.၅ Light Activated Switch

ခုနေက light sensor နဲ့ relay ကို သုံးပြီး ညမောင် လာရင် အလိုအလျောက် မီးဖွင့် ပေးမယ့် light activated switch လေး လုပ်ဖို့ အတွက် RPi နဲ့ ဆက်သွယ်ပုံ ကို ပုံ ၃.၇ မှာ ပြထား ပါတယ်။



ပုံ ၃.၇: Light activated switch အတွက် light sensor နှင့် relay ကို RPi ဖြင့် ဆက်သွယ်ပုံ။

အဲဒီ အတွက် နမူနာ ပရိုဂရမ် အနေနဲ့ [lightswitch.cpp](#) ကို စာရင်း ၃.၅ မှာ ပြထား ပါတယ်။ ရှုက နမူနာ အတိုင်း CE\_IO class ကို သုံးပြီး GPIO25 ကို light sensor input အနေ နဲ့ GPIO4 ကို relay အတွက် output အနေ နဲ့ သုံးလိုက် ပါတယ်။ ပြီးတဲ့ အခါ while loop ကို အဆုံး မရှိ ပတ်နေပြီး light sensor က ဖတ်လို့ ရတဲ့ တန်ဖိုး ကို relay အတွက် ထုတ်ပေး မှာ ဖြစ် ပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <string>
4 #include "ce_io.h"
5 using namespace std;
```

```

6 int main()
7 {
8     CE_IO light(25, INPUT);
9     CE_IO relay(4, OUTPUT);
10
11    while(1){
12        //print status
13        printf("Light sensor output = %d \n", light.Read());
14
15        //produce relay output
16        relay.Write(light.Read());
17
18        //sleep
19        usleep(1000000);
20    }
21
22    return 0;
}

```

စာရင်း ၃.၅: lightswitch.cpp

ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run နိုင် ပါတယ်။ ပရိုဂရမ် ကို အဆုံးသတ် ဖို့ အတွက် **ctrl+c** ကို နှိပ်နိုင် ပါတယ်။

```

$ g++ lightswitch.cpp -o lightswitch
$ sudo ./lightswitch

```

## ၃.J Analog input များကိုဖတ်ခြင်း

Raspberry Pi မှာ analog input pin တွေ မပါဘူး အတွက် analog voltage တွေကို ဖတ်ချင်ရင် MCP3008 စာတွဲ analog to digital converter (ADC) chip တွေကို ဆက်သွယ် အသုံးပြု ဖို့ လို ပါတယ်။ အဲဒီ အကြောင်းကို အပိုင်း ၄.၃ မှာ ဆွေးနွေး ထားပါတယ်။

## ၃.၃ Analog output ထုတ်ခြင်း

Raspberry Pi မှာ PWM1 နဲ့ PWM2 ဆိုတဲ့ PWM output နှစ်ခုပါ ပါတယ်။ PWM1 က 40 pin header ရဲ့ Pin12 (GPIO18) မှာ ရှိပြီး၊ PWM2 ကတေသ့ Pin33 (GPIO13) မှာ ဖြစ်ပါ တယ်။ Raspbian မှာ အဲဒီ PWM နှစ်ခုလုံးက အသံ ထုတ်ဖို့ အသံးပြု ထားတာ မို့ သူတို့ ကို သုံးချင် ရင် hardware register တွေကို /dev/mem သုံးပြီး တိုက်ရှိက် ကိုင်တွယ် တာမျိုး ပဲ လုပ်လို့ ရပြီး [Her14]၊ သင့်တော် ကောင်းမွန် တဲ့ အသံးပြု နည်း မရှိ ပါဘူး။

WiringPi ([wiringpi.com](http://wiringpi.com))၊ pigpio ([abyz.me.uk/rpi/pigpio](http://abyz.me.uk/rpi/pigpio)) စတဲ့ လိုင်ဘရီ တွေ သုံးလို့ ရပေ မယ့် PWM စ Digital to Analog Converter (DAC) စတဲ့ hardware တွေ ပါတဲ့ module တွေ ဆက်သွယ် အသံးပြု တဲ့ နည်းကို ပဲ အကြံပြု လို ပါတယ်။ အဲဒီ အကြောင်း တွေကို အပိုင်း [၄.၄](#) မှာ ဆွေးနွေး ပါမယ်။

## ၃.၄ Wiring Pi ကို အသံးပြုခြင်း

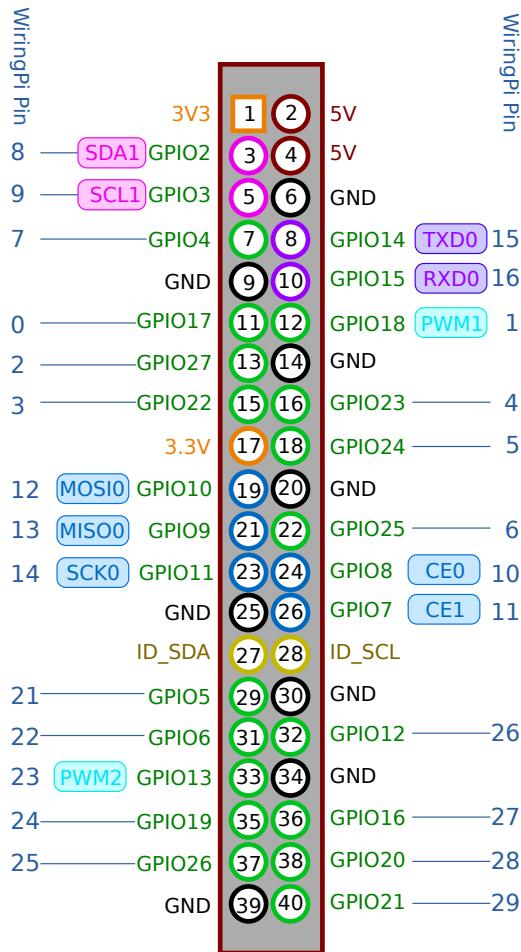
WiringPi - [wiringpi.com](http://wiringpi.com) က Raspberry Pi တွေ ပေါ်မှာ GPIO တွေကို အသံး ပြုဖို့ အတွက် C နဲ့ ရေးသား ထားတဲ့ လိုင်ဘရီ ဖြစ် ပါတယ်။ သူကို အသံးပြု ရတာ Arduino - [arduino.cc](http://arduino.cc) သုံးရ တာနဲ့ ဆင်တူပြီး၊ လွယ်ကူ အဆင် ပြေ တာကြောင့် လူကြိုက် များ၊ အသံးများ တဲ့ လိုင်ဘရီ တစ်ခု ဖြစ် ပါတယ်။ မူရင်း လိုင်ဘရီ က Raspberry Pi အတွက် ပဲ သုံးလို့ ရပြီးတရာ့၍ platform တွေပေါ်မှာ လည်း WiringPi ကို သုံးလို့ ရအောင် ported လုပ်ထား တာတွေ ရှိပါတယ်။

### ၃.၄.၁ Pins များ

WiringPi မှာ သုံးတဲ့ pin နံပါတ် တွေက GPIO pin နံပါတ် တွေနဲ့ မတူ ပါဘူး။ သူတို့ ကို ပုံ ၃.၈ မှာ တွေ့နှင့် ပါတယ်။

Raspbian မှာ WiringPi လိုင်ဘရီ ပါလာ ပြီးသား ဖြစ် ပါတယ်။ Terminal မှာ အောက်ပါ command ကို ရိုက်ကြည့် ပြီး gpio version ကို ကြည့်လိုက် ရင် ပုံ ၃.၉ မှာလို တွေ့နှင့် ပါတယ်။

```
$ gpio -v
```



ঃ ২.৮: WiringPi এর pin স্থিতিগতিঃ।

```
pi@raspberrypi:~ $ gpio -v
gpio version: 2.44
Copyright (c) 2012-2017 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
  Type: Pi 3, Revision: 02, Memory: 1024MB, Maker: Embest
  * Device tree is enabled.
  *--> Raspberry Pi 3 Model B Rev 1.2
  * This Raspberry Pi supports user-level GPIO access.
```

ঃ ২.৯: WiringPi এর version অনুরোধের জন্য।

လက်ရှိ အသုံးပြု နေတဲ့ Raspberry Pi ရဲ့ GPIO တွက် ကြည့်ဖို့ အတွက် အကောင်းဆုံး နည်းကတော့ အောက်ပါ command ကို ရိုက်ကြည့် တဲ့ နည်း ဖြစ်ပါ တယ်။ အဲဒီ အခါ လက်ရှိ Raspberry Pi အတွက် pin တွက် ပုံ ၃.၁၀ မှာ ပြထား သလို တွေ့ရ မှာ ဖြစ်ပါတယ်။

```
$ gpio readall
```

Pi@raspberrypi:~ \$ gpio readall											
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
		3.3v			1    2			5v			
2	8	SDA.1	ALT0	1	3    4			5v			
3	9	SCL.1	ALT0	1	5    6			0v			
4	7	GPIO. 7	IN	1	7    8	0	IN	TxD	15	14	
		0v			9    10	1	IN	RxD	16	15	
17	0	GPIO. 0	IN	0	11    12	1	OUT	GPIO. 1	1	18	
27	2	GPIO. 2	IN	0	13    14			0v			
22	3	GPIO. 3	IN	0	15    16	0	IN	GPIO. 4	4	23	
		3.3v			17    18	0	IN	GPIO. 5	5	24	
10	12	MOSI	ALT0	0	19    20			0v			
9	13	MISO	ALT0	0	21    22	0	IN	GPIO. 6	6	25	
11	14	SCLK	ALT0	0	23    24	1	OUT	CE0	10	8	
		0v			25    26	1	OUT	CE1	11	7	
0	30	SDA.0	ALT0	1	27    28	1	ALT0	SCL.0	31	1	
5	21	GPIO.21	IN	1	29    30			0v			
6	22	GPIO.22	IN	1	31    32	0	IN	GPIO.26	26	12	
13	23	GPIO.23	IN	0	33    34			0v			
19	24	GPIO.24	IN	0	35    36	0	IN	GPIO.27	27	16	
26	25	GPIO.25	IN	0	37    38	0	IN	GPIO.28	28	20	
		0v			39    40	0	IN	GPIO.29	29	21	
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	

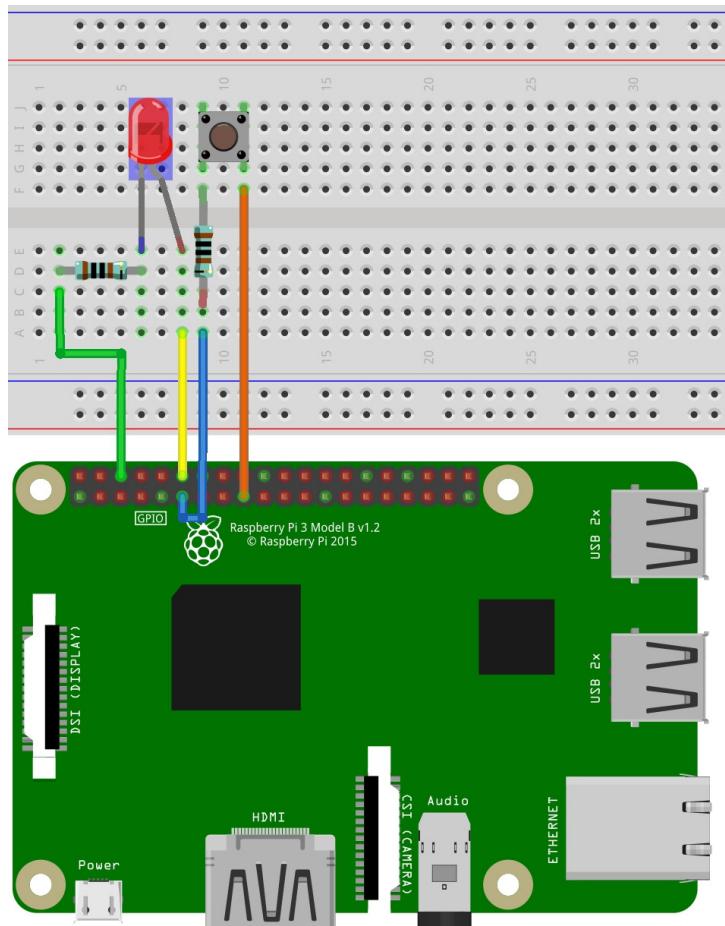
ပုံ ၃.၁၀: WiringPi ၏ Pin များကို readall ဖြင့် ဖော်ခြင်း။

### ၃.၄.၂ Digital IO များသုံးခြင်း

WiringPi ကို သုံးပြီး digital input နဲ့ output တွေ ရေးတာ၊ ဖတ်တာ ကို စမ်းကြည့် ဖို့ အတွက် WiringPi Pin နံပါတ် 0 (header pin 11) မှာ ခလုတ် တစ်ခု ဆက်ပြီး၊ WiringPi Pin 1 (header pin 12) မှာ LED မီးလုံး တစ်လုံး ကို ပုံ ၃.၁၁ မှာ ပြထား သလို ဆက်သွယ် လိုက် ပါမယ်။

## ၃.၄. WIRING PI ကို အသုံးပြုခြင်း

၇၇



ပုံ ၃.၁၁: WiringPi ၏ digital input နှင့် output များကို အသုံးပြုခြင်း။

LED မီးလုံး ကို အဖွင့် အပိတ် လုပ်ကြည့် ဖို့ အတွက် terminal မှာ အောက်က command တွေ သုံးပြီး pin 1 ကို အထွက် အနေနဲ့ သတ်မှတ် 1 နဲ့ 0 ကို ရေးလိုက် တဲ့ အခါ မီးလုံး အဖွင့် အပိတ် ဖြစ်သွား တာကို တွေ့နိုင် ပါတယ်။

```
$ gpio mode 1 out  
$ gpio write 1 1  
$ gpio write 1 0
```

အဲဒီ နည်းအတိုင်း ခလုပ် ရဲ့ pin 0 ကို အဝင် အနေနဲ့ သတ်မှတ် ဝင်လာ တဲ့ တန်ဖိုးကို ဖတ်ဖို့ အတွက် အောက်က command တွေကို သုံးနိုင် ပါတယ်။

```
$ gpio mode 0 in
$ gpio read 0
```

WiringPi ကို C++ နဲ့ သုံးဖို့ အတွက် တော့ ပရိုဂရမ် ရဲ့ အစ မှာ wiringPi.h ကို include လုပ်ပေး ရပါမယ်။ ပြီးတဲ့ အခါ Arduino မှာ အတိုင်း mode ကို pinMode သတ်မှတ်ပြီး၊ digitalWrite နဲ့ ရေးနိုင် ပါတယ်။ LED ကို blink လုပ်ပေးတဲ့ နှမူနာ wio.cpp ကို စာရင်း ၃.၆ မှာ ပြထား ပါတယ်။ ခလုတ်ကို ဖတ်ပြီး နှိပ် ထားတဲ့ အချိန်မှာ blink လုပ်တာ မြန်ပေး ပါမယ်။

```
1 //File: wio.c
2 //Description: simple digital IO using wiringPi
3 //Reference: http://wiringpi.com/examples/quick2wire-and-wiringpi/install-and
4 //             -testing/
5
6 #include <stdio.h>
7 #include <wiringPi.h>
8
9 #define BTN 0 //wiringPi pin 0 (pin 11 on the header)
10#define LED 1 //wiringPi pin 1 (pin 12 on the header)
11int main(void)
12{
13    int p;
14    printf("Wiring Pi digital IO.\n");
15    if (wiringPiSetup() == -1) {
16        printf("Fail to setup WiringPi!\n");
17        return 1;
18    }
19    pinMode(BTN, INPUT);
20    pinMode(LED, OUTPUT);
21
22    while(1)
23    {
24        if (digitalRead(BTN) == HIGH) p = 250;
25        else p = 1000;
26
27        digitalWrite(LED, HIGH); // On
```

## ၃.၄. WIRING PI ကို အသံးပြုခြင်း

၇၉

```
27     delay(p);           // ms
28     digitalWrite(LED, LOW); // Off
29     delay(p);
30 }
31 return 0;
32 }
```

စာရင်း ၃.၆: wio.cpp

ပရိုဂရမ် ကို build လုပ် တဲ့ အခါ wiringPi လိုင်ဘရီ နဲ့ ချိတ်ပြီး အောက်ပါ အတိုင်း build လုပ်၊ run နိုင် ပါတယ်။ ပရိုဂရမ် ကို Ctrl+C ကို နှိပ် ပြီး အဆုံးသတ် နိုင် ပါတယ်။

```
$ g++ wio.cpp -o wio -lwiringPi
$ sudo ./wio
```

### ၃.၄.၃ Wiring Pi ကို source မှ build လုပ်၍ တပ်ဆင်ခြင်း

WiringPi နောက်ဆုံး version ကို source ကနေ build လုပ်ပြီး တပ်ဆင် ချင်တယ် ဆိုရင် လည်း သူရဲ့ source တွေကို [git.drogon.net/wiringPi](https://git.drogon.net/wiringPi) ကနေ ယူပြီး တပ်ဆင် နိုင် ပါတယ် [Hen18]။

လက်ရှိ ရှိနေတဲ့ လိုင်ဘရီ ကို အောက်က command သုံးပြီး ဖြုတ်လိုက် ပါမယ်။

```
$ sudo apt purge wiringpi
$ hash -r
```

ပြီးတဲ့ အခါ အောက်ပါ အတိုင်း download လုပ်ပြီး build လုပ်၊ install လုပ်နိုင် ပါတယ်။

```
$ git clone git://git.drogon.net/wiringPi
$ cd ~/wiringPi
$ ./build
```

အဲဒီ နောက်

```
$ gpio -v
```

နဲ့ ပြန်စစ် ကြည့်တဲ့ အခါ version မြင့် သွားတာကို ပုံ ၃.၁၂ မှာ ပြထား သလို တွေ့နိုင် ပါတယ်။

```
[Compile] gpio.c
[Compile] readall.c
[Link]
[Install]

All Done.

NOTE: To compile programs with wiringPi, you need to add:
      -lwiringPi
      to your compile line(s) To use the Gertboard, MaxDetect, etc.
      code (the devLib), you need to also add:
      -lwiringPiDev
      to your compile line(s).

pi@raspberrypi:~/wiringPi $ gpio -v
gpio version 2.46
Copyright (c) 2012-2018 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
  Type: Pi 3, Revision: 02, Memory: 1024MB, Maker: Embest
  * Device tree is enabled.
  *--> Raspberry Pi 3 Model B Rev 1.2
  * This Raspberry Pi supports user-level GPIO access.

pi@raspberrypi:~/wiringPi $ _
```

ပုံ ၃.၁၂: WiringPi ၏ version အသစ်ကို ပြန်စစ်ခြင်း။

### ၃.၄.၄ Analog Output သုံးခြင်း

Raspberry Pi မှာ PWM အတွက် နှစ်ခု ပါပေါ်မယ့် audio အတွက် သုံးထား တာမူး သူတို့ ကို မောင်တာ တွေ၊ မီးလုံး တွေ ထိန်းဖို့ သုံးချင် တယ် ဆိုရင် audio အတွက် မရှိ အောင် သတိထား ဖို့ လိုပါတယ်။ အဲဒါ က system ရဲ့ stability ကို ထိခိုက် နိုင် သလို၊ အကာယ်၍ GPIO တွေကို ထိန်းပြီး PWM waveform တွေကို emulate လုပ်တဲ့ software-pwm တွေ ဆိုရင် လည်း system ရဲ့ resources တွေ ကုန်ပြီး high frequency, high precision လုပ်ငန်း တွေ အတွက် မသင့်တော် ပါဘူး၊ ဒါကြောင့် ရွှေမှာ ပြောခဲ့ သလိုပဲ အပိုင်း ၄.၅ မှာ ဆွေးနွေး ထားသလို ဖျေးပေါ့ တဲ့ external chip လေးတွေ တပ်ပြီး သုံးတာ က ပိုပြီး ကောင်းမွန် သင့်တော် ပါတယ်။

အကြောင်း အမျိုးမျိုး ကြောင့် သူတို့ကို သုံးဖို့ လိုလာ ရင်တော့ အောက်က နမူနာ wpwm.cpp (စာရင်း ၃.၇) မှာ ပြထား သလို WiringPi နဲ့ သုံးလို့ ရပါတယ်။ Pin mode သတ်မှတ် တဲ့ အချိန် မှာ

## ၃.၄. WIRING PI ကို အသုံးပြုခြင်း

၈၁

PWM\_OUTPUT လို့ သတ်မှတ်ပြီး တန်ဖိုး ရေးတဲ့ အခါ pwmWrite ဆိုတဲ့ function ကို သုံးလိုက် ရုံးပါ လွယ်ကူ ရှိရန် ပါတယ်။ ဒါ ပရိုဂရမ် မှာ WiringPi pin 1 နဲ့ ဆက်ထား တဲ့ LED ကို PWM duty cycle အမျိုးမျိုး ပြောင်းပြီး အလင်း အမြိန် ထိန်းချုပ် ပေး ပါတယ်။

```
1 //File: wpwm.cpp
2 //Description: PWM using WiringPi
3 //Reference: https://github.com/WiringPi/WiringPi/blob/master/examples/pwm.c
4
5 #include<stdio.h>
6 #include<wiringPi.h>
7
8 int main(void)
9 {
10     int i;
11     printf("WiringPi PWM.\n");
12     if (wiringPiSetup() == -1) {
13         printf("Fail to setup WiringPi!\n");
14         return 1;
15     }
16
17     pinMode(1,PWM_OUTPUT); // Set PWM LED as PWM output
18
19     while(1)
20     {
21         for (i = 0; i < 1024; i+=125) {
22             pwmWrite(1, i);
23             delay(500); //ms
24         }
25     }
26     return 0;
27 }
```

စာရင်း ၃.၇: wpwm.cpp

ပရိုဂရမ် ကို wiringPi လိုင်ဘရီ နဲ့ ချိတ်ပြီး build လုပ်၊ run နိုင် ပါတယ်။ ပရိုဂရမ် ကို အဆုံးသတ်

ချင်ရင် တော့ Ctrl+C ကို သုံးနိုင် ပါတယ်။

```
$ g++ wpwm.cpp -o wpwm -lwiringPi  
$ sudo ./wpwm
```

## အကိုးအကားများ

- [Hen18] Gordon Henderson. Wiring Pi - GPIO Interface library for the Raspberry Pi : Download and Install. 2018. url: <http://wiringpi.com/download-and-install/>.
- [Her14] Hertaville. Accessing The Hardware PWM Peripheral on the Raspberry Pi in C++. 2014. url: <http://hertaville.com/rpipwm.html>.

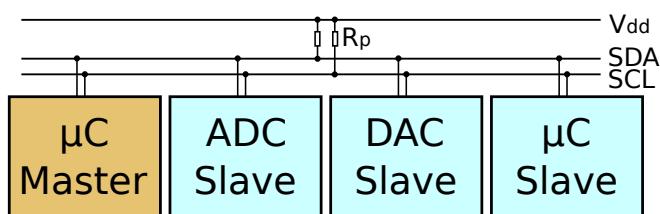
## အခန်း ၄

# Serial Bus များ

IC အချင်းချင်း အကွာအဝေး အနီးအနား ပဲ ဆက်သွယ်တဲ့ အခါ အသုံးများ တဲ့ I2C, SPI စတဲ့ serial bus communication တွေ အကြောင်း ဆွေးနွေး ပါမယ်။

### ၄.၁ I2C

I2C လိုလည်း ခေါ်တဲ့ Inter-Integrated Circuit serial bus က အကွာအဝေး အနီးအနား အတွက် IC chip တွေအချင်းချင်း master အများကြီး နဲ့ slave အများကြီး ဆက်သွယ်နိုင် တဲ့ synchronous ဆက်သွယ်ရေး ဖြစ်ပါတယ်။ Philips Semiconductor (ယခု NXP Semiconductors) က တိုထွင် ခဲ့တာ ပါ။



ပုံ ၄.၁: I2C master အနေနှင့် သုံးထားသည့် microcontroller တစ်ခု ကို slave သုံးခု အား pull-up resistor များသုံး၍ ဆက်သွယ်ပုံ။

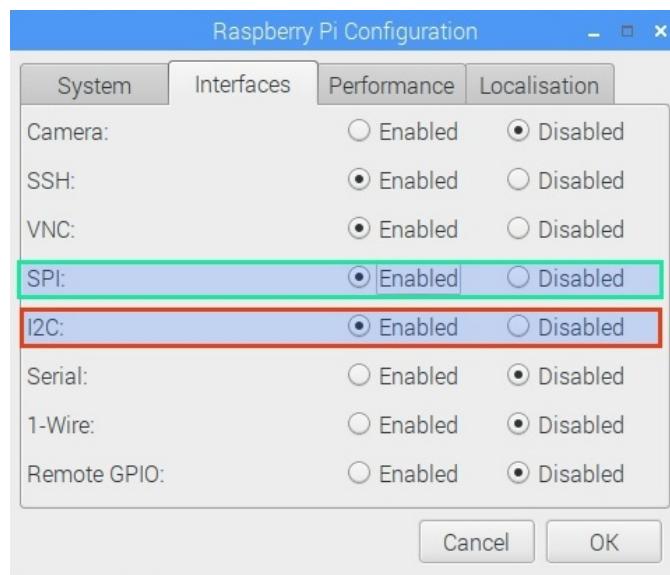
I2C မှာ Serial Data Line (SDA) နဲ့ Serial Clock Line (SCL) လိုလော်တဲ့ bi-directional လိုင်း နှစ်လိုင်း ကို သုံးပါတယ်။ သူတို့က open drain ဖြစ်ပြီး pull-up resistor တွေနဲ့ ဆက်ပြီး သုံးဖို့

လိုပါတယ်။ I2C ဆက်သွယ်မှု နမူနာ တစ်ခု ကို (Wikipedia က ရယူ ထားတဲ့) ပုံ ၄.၁ မှာ ပြထား ပါတယ် [Wik18a]။

Raspberry Pi မှာ I2C bus နှစ်ခု ပါရှိ ပါတယ်။ ပထမ i2c-1 ရဲ့ SDA နဲ့ SCL က header ရဲ့ pin 3 နဲ့ 5 မှာ ဖြစ်ပြီး၊ i2c-0 ရဲ့ SDA နဲ့ SCL က pin 27 နဲ့ 28 မှာ ဖြစ်ပါတယ် (ထေား ၄.၁) [Mol16]။

ထေား ၄.၁: Raspberry Pi ရှိ I2C bus များ။

Device	SDA	SCL	ဖော်ပြချက်
i2c-1	Pin3	Pin5	Pi configuration တွင် enabled လုပ်၍ သုံးနိုင် သည်
i2c-0	Pin27	Pin28	တပ်ဆင် အသုံး ပြုမည့် HAT များကို automatic configuration လုပ်ရန် သီးသန့် ဖယ်ထား ပြီး၊ ပုံမှန်အားဖြင့် အသုံးပြု၍ မရပါ



ပုံ ၄.၂: Raspberry Pi Configuration ကို သုံး၍ I2C ကို enabled လုပ်ခြင်း။

I2C1 ကို enable လုပ်ဖို့ အတွက် Main Menu → Preferences → Raspberry Pi Configuration → Interfaces tab ကို ဖွင့်ပြီး၊ အဲဒီ မှာ I2C ကို ပုံ ၄.၂ မှာ ပြထား သလိုမျိုး enabled လုပ် ပြီးတဲ့ အခါ reboot လုပ်ဖို့ လို ပါတယ်။

ပုံမှန် အားဖြင့် I2C1 ကို ပဲ အသုံးပြုလို ရပြီး၊ I2C0 ကတော့ Raspberry Pi မှာ တပ်ဆင် အသုံးပြုမယ့်

HAT တွေကို အလို အလောက် configure လုပ်ဖို့ အတွက် reserved လုပ်ထား တာမှို စက်ရဲ configuration တွေကို လိုက် မပြင်ပဲ သုံးလို မရ ပါဘူး။ အကယ်၍ မဖြစ်မနေ သုံးဖို့ လိုလာ ရင်တော့ /boot/cmd-line.txt ဆိုတဲ့ ဖိုင် ထဲမှာ ရှိတဲ့ စာကြောင်း မှာ

```
bcm2708.vc_i2c_override=1
```

ကို အဲဒီ လိုင်းမှာပဲ ထပ်ဖြည့် ဖို့လို ပါတယ်။ ပြီးတဲ့ အခါ /boot/config.txt ထဲမှာ လည်း

```
dtparam=i2c_vc=on
```

အဲဒီ နောက် စက်ကို reboot လုပ် လိုက်ရင် i2c bus နှစ်ခု လုံးကို သုံးလို ရပါမယ်။

### ၄.၁.၁ i2c-tools

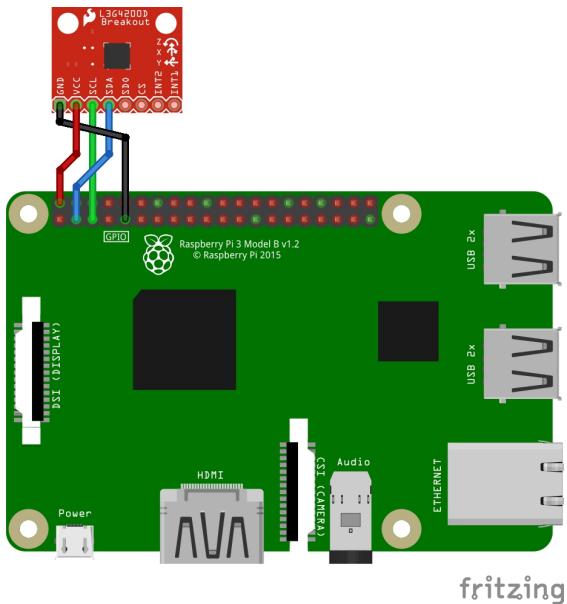
I2C ကို အသုံးပြု ဖို့ အတွက် linux ရဲ့ i2c-tools ကို အောက်ပါ အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt-get install i2c-tools
```

I2C device တွေကို detect လုပ်ဖို့ အတွက် i2cdetect ဆိုတဲ့ command ကို သုံးနိုင် ပါတယ်။ ”-l” က တပ်ဆင်ထားတဲ့ device တွေကို list လုပ်ပေးဖို့ ဆိုလို တာပါ။

```
$ i2cdetect -l
```

i2cdetect command ကို ပဲ ”-y” ထည့်ပေး လိုက်ရင် interactive mode ကို disabled လုပ်တာပါ။ အသုံးပြုသူ ရဲ့ အတည်ပြုချက် တွေကို ပြန် မမေးပဲ တိုက်ရှိက် လုပ်ပေး မှာ ဖြစ်ပါတယ်။ ”-r” ဆိုရင်တော့ receive byte ဖြစ်ပြီး probing လုပ်ကြည့်ဖို့ သုံးနိုင် ပါတယ်။ ပုံ ၄.၃ မှာ ပြထား သလို i2c-1 မှာ L3G4200D Gyroscope module ကို ဆက်သွယ်ပြီး probe လုပ်ကြည့်တဲ့ အခါ ပုံ ၄.၄ အတိုင်း တွေ့ရ ပါတယ်။



ပုံ ၄.၃: Gyroscope ကို I2C သုံး၏ ဆက်သွယ်ခြင်း။

```
$ i2cdetect -y -r 1
```

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
10:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
20:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
30:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
40:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
50:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
60:	-	-	-	-	-	-	-	-	-	-	-	69	-	-	-	-
70:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

ပုံ ၄.၄: I2C bus ကို probe လုပ်ခြင်း။

Probe လုပ်ကြည့်လို ဘာ အဖြစ်မှ ပြန်မရတဲ့ address တွေမှာ “-” လို ပြ နေမှာပါ။ “UU” ကတော့ driver တွေ လက်ရှိ သုံးနေတဲ့ address မို့ probe မလုပ်ဘူး လို ဆိုလို တာပါ။ အဲဒီ address တွေမှာ chip တွေရှိနေတယ် လို ပြောနိုင် ပါတယ်။ Hexadecimal နံပါတ် တစ်ခု တွေရင်တော့ chip ကို တွေ့တဲ့ address လို ဆိုလိုတာပါ။ ဒါ နမူနာ မှာ gyroscope ရဲ့ address ကို 0x69 လို တွေ့ရ ပါတယ်။ L3G4200

ရဲ့ datasheet [STM10] ထဲမှာဖော်ပြထားတဲ့ အတိုင်း SDO pin ကို voltage supply ပေးထားရင် ရမည့် slave address ရဲ့ binary နံပါတ် 1101001 နဲ့ ကိုက်ညီ တာကို တွေ့ရ ပါတယ်။

I2C bus မှာ ချိတ်ထားတဲ့ device ရဲ့ register တွေကို ဖတ်ပြီး ဖော်ပြန့် အတွက် i2cdump ကို သုံးလို့ရ ပါတယ်။ နူးနာ command နဲ့ ရလဒ် ကို ပုံးပြထား ပါတယ်။ Address 0x0F ကဲ Who am I register ရဲ့ တန်ဖိုး 0xD3 ကို အလွယ် တကူ စစ်ကြည့် နိုင် ပါတယ်။

```
$ i2cdump -y 1 0x69
```

```
pi@raspberrypi:~ $ i2cdump -y 1 0x69
No size specified (using byte-data access)
    0   1   2   3   4   5   6   7   8   9   a   b   c   d   e   f      0123456789abcdef
00: c4 66 a4 cc 4c d0 9a c1 41 05 e4 d5 38 87 00 d3  ?f??L???A???8?.?
10: 36 26 66 44 0c a0 40 40 8c 99 71 80 83 80 81 81  6&fD??@?/?q??????
20: 07 00 00 00 00 00 ff a2 04 f9 03 69 f6 00 20  ?.....????i?.
30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
80: c4 66 a4 cc 4c d0 9a c1 41 05 e4 d5 38 87 00 d3  ?f??L???A???8?.?
90: 36 26 66 44 0c a0 40 40 8c 99 71 80 83 80 81 81  6&fD??@?/?q??????
a0: 07 00 00 00 00 00 00 a2 04 f9 03 69 f6 00 20  ?.....????i?.
b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..... .
f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
```

ဦး ၄.၅: I2C bus ရဲ့ device တစ်ခု၏ register များကို ဖတ်ခြင်း။

I2C device ရဲ့ register တန်ဖိုး တွေကို ဖတ်ဖို့ ရေးဖို့ အတွက် i2cget နဲ့ i2cset တို့ကို သုံးနိုင် ပါတယ်။ ဥပမာ i2c-2 မှာ ချိတ်ထားတဲ့ slave address 0x69 ရဲ့ register address 0x21 မှာ ရဲ့ CTRL\_REG2 ကို ဖတ်ပြီး တန်ဖိုး ကို 0x09 ပြောင်း ရေး တဲ့ command တွေကို အောက်က အတိုင်း တွေ့နိုင် ပါတယ်။

```
$ i2cget -y 2 0x69 0x21
$ i2cset -y 2 0x69 0x21 0x09
$ i2cget -y 2 0x69 0x21
```

### ၄.၁.၂ C++ ဖြင့်သုံးခြင်း

I2C bus ကို အသုံးပြု ပြီး device တွေနဲ့ဆက်သွယ် မယ့် C++ နမူနာ ပရိုဂရမ် ကို ဖော်ပြ ဆွေးနွေး ပါမယ် [Eli12; Ada14]။ I2C bus ကို စတင် အသုံးပြု ဖို့ အတွက် သက်ဆိုင်ရာ ”/dev/i2c-1” အစရှိတဲ့ bus ကို ဖွင့် ရပါမယ်။ ရေးတဲ့ အခါ buffer မသုံးအောင် fopen အစား open ကိုပဲ သုံးရ ပါမယ်။

```

1 int file;
2 char *filename = "/dev/i2c-1";
3 if ((file = open(filename, O_RDWR)) < 0) {
4     printf("Failed to open the i2c bus");
5 }
```

I2C bus ကို ဖွင့်ပြီးတဲ့ အခါ သူရဲ့ slave address ကို အောက်ပါ အတိုင်း ioctl ဖန်ရှင် သုံးပြီး သတ်မှတ် စတင် နိုင်ပါတယ်။

```

1 int addr = 0x69;
2 if (ioctl(file, I2C_SLAVE, addr) < 0) {
3     printf("Failed to acquire bus access and/or talk to slave.\n");
4 }
```

ဖတ်ဖို့ အတွက် read ကို သုံးနိုင်ပြီး file handle ရယ်။ ဖတ်လို့ ရတဲ့ data တွေ သိမ်းဖို့ buffer ရယ်။ ဖတ်မယ့် အရေအတွက် ရယ် ပေးဖို့ လိုပါတယ်။

```

1 char buf[2];
2 if (read(file, buf,2) !=2) {
3     printf("Failed to read from the i2c bus.\n");
4 }
```

ရေးမယ် ဆိုရင်တော့ write ကို သုံးနိုင် ပါတယ်။ ခုနါက လိုပဲ file handle ရယ်။ ရေးမယ့် data တွေသိမ်းထားတဲ့ buffer ရယ်။ ရေးမယ့် အရေအတွက် ရယ် ပေးဖို့ လိုပါတယ်။

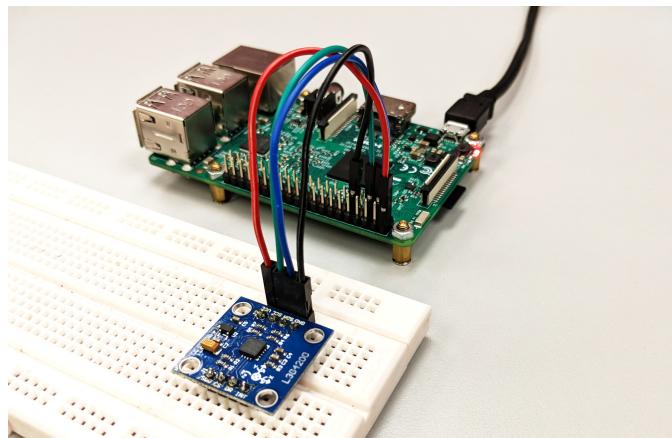
```

1 char buf[2]={0x21,0x09};
2 if (write(file, buf,2) !=2) {
3 printf("Failed to write to the i2c bus.\n");
4 }

```

## ၄.၁.၃ Gyroscope ကိုအသုံးပြုခြင်း

I2C သုံးဖြီး L3G4200D gyroscope ကို အသုံးပြု တဲ့ နမူနာ C++ ပရိုဂရမ် ကို ce\_i2c.h (စာရင်း ၄.၁) နဲ့ i2cgyro.cpp (စာရင်း ၄.၂) မှာ ဖော်ပြ ထား ပါတယ်။ အဲဒီ အတွက် L3G4200D gyroscope module လေးကို ပုံ ၄.၃ နဲ့ ပုံ ၄.၆ မှာ ပြထား သလို ဆက်သွယ် ထား ပါမယ်။



ပုံ ၄.၆: L3G4200D gyroscope module ကို တပ်ဆင်အသုံးပြုခြင်း။

```

1 //File: ce_i2c.h
2 //Description: CE_I2C class to use i2c communication
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye
6
7 //Reference : http://elinux.org/Interfacing_with_I2C_Devices
8

```

```
9 #ifndef CE_I2C_H
10 #define CE_I2C_H
11
12 #include <string.h>
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <unistd.h>
16 #include <linux/i2c-dev.h>
17 #include <sys/ioctl.h>
18 #include <fcntl.h>
19 #include <sstream>
20
21 using namespace std;
22
23 class CE_I2C
24 {
25     public:
26         CE_I2C();
27         CE_I2C(int bus_id,int slave_address);
28         ~CE_I2C();
29         bool Begin(int bus_id,int slave_address);
30         bool Write(char* buf,int n);
31         bool Read(char* buf,int n);
32         template <typename T>
33             string ToString(T Number);
34     private:
35         int fd;
36 };
37
38 template <typename T>
39 string CE_I2C::ToString(T a)
40 {
41     ostringstream ss;
42     ss << a;
43     return ss.str();
44 }
```

```
45
46 CE_I2C::CE_I2C()
47 {
48     //ctor
49 }
50
51 CE_I2C::CE_I2C(int bus_id, int slave_address)
52 {
53     Begin(bus_id, slave_address);
54 }
55
56 CE_I2C::~CE_I2C()
57 {
58     //dtor
59     close(fd);
60 }
61
62 bool CE_I2C::Begin(int bus_id, int slave_address)
63 {
64     string filename = "/dev/i2c-";
65     filename += ToString(bus_id);
66     if ((fd = open(filename.c_str(), O_RDWR)) < 0) {
67         perror("Failed to open the i2c bus\n");
68         return false;
69     }
70     if (ioctl(fd, I2C_SLAVE, slave_address) < 0) {
71         perror("Failed to acquire bus access and/or talk to slave.\n");
72         return false;
73     }
74     return true;
75 }
76
77 bool CE_I2C::Write(char* buf, int n)
78 {
79     if (write(fd, buf, n) != n) {
80         perror("Failed to write to the i2c bus.\n");
```

```

81     return false;
82 }
83 return true;
84 }
85
86 bool CE_I2C::Read(char* buf, int n)
87 {
88     if (read(fd, buf, n) != n) {
89         perror("Failed to read from the i2c bus.\n");
90         return false;
91     }
92     return true;
93 }
94
95 #endif // CE_I2C_H

```

စွာရင်း ၄.၁: ce\_i2c.h

```

1 //File: i2cgyro.cpp
2 //Description: Using L3G4200D gyroscope
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye
6
7 #include<stdio.h>
8 #include"ce_i2c.h"
9 using namespace std;
10 int main()
11 {
12     char d[6]={0,0,0,0,0,0};
13     CE_I2C gyro(1,0x69); //use i2c-1, address 0x69
14
15     //read the "who am i" register at address 0x0F
16     //its value should be 0xD3
17     char raddr=0x0F;

```

```

18     gyro.Write(&raaddr,1);
19     gyro.Read(d,1);
20     printf("I am 0x%02x\n",d[0]);
21
22 //Configure Gyro
23 //CTRL_REG2 = /0 /0 /HPM1 /HPM0 /HPCF3 /HPCF2 /HPCF1 /HPCF0 /
24 //Default = /0 /0 /0 /0 /0 /0 /0 /0 /
25 //HPM = 00 => Normal (High Pass filter Mode selection)
26 //HPC = 1001 => 0.1 High Pass Filter Cut-off freq configuration
27 //write @address 0x21, value = 0x09
28 d[0]=0x21;
29 d[1]=0x09;
30 gyro.Write(d,2);
31
32 //CTRL_REG3 = /I1_Int1 /I1_Boot /H_Lactive /PP_OD /I2DRDY /I2_WTM /I2_ORun /
33 //I2_Empty /
34 //Default = /0 /0 /0 /0 /0 /0 /0 /0 /
35 //Use Default
36
37 //CTRL_REG4 = /BDU /BLE /FS1 /FS0 / - /ST1 /ST0 /SIM /
38 //Default = /0 /0 /0 /0 /0 /0 /0 /0 /
39 //BDU = 1 => Block Data Update
40 //BLE = 0 => Little endian
41 //FS = 00 => 250 dps (Full scale selection)
42 //ST = 000 => Disable Self test
43 //write @ address 0x23, value =0x80
44 d[0]=0x23;
45 d[1]=0x80;
46 gyro.Write(d,2);
47
48 //CTRL_REG5 = /BOOT /FIFO_EN / - /HPen /INT1_Sel1 /INT1_Sel0 /Out_Sel1 /
49 //Out_Sel0 /
50 //Default = /0 /0 /0 /0 /0 /0 /0 /0 /
51 //BOOT = 0 => Normal mode (Reboot Memory Content)

```

```

50 //FIFO_EN = 0 => disable FIFO
51 //HPen = 0 => disable (High Pass Filter)
52 //INT1_Sel = 00 => Non high pass filtered data are used for interrupt
53 //generation
54 //Out_Sel = 00 => no filtering
55 //Use Default
56
56 //CTRL_REG1 = /DR1/DR0/BW1/BW0/PD/Zen/Yen/Xen/
57 //Default = 10 10 10 10 10 1 1 1 1 1
58 //DR = 11 => ODR 800 Hz (output data rate)
59 //BW = 10 => Cut-off 50 (Bandwidth 50 Hz)
60 //PD = 1 => Normal
61 //Zen = Yen = Xen = 1 => Enable
62 //write @ address 0x20, value =0xEF
63 d[0]=0x20;
64 d[1]=0xEF;
65 gyro.Write(d,2);
66 int x,y,z;
67
68 for(int i=0;i<20;i++)
69 {
70     //read address 0x28, OUT_X_L register
71     //set MSB bit for auto address increment
72     raddr=0xA8;
73     gyro.Write(&raddr,1);
74     gyro.Read(d,6);
75     x=((int(d[1])<<8)+d[0])|(d[1]&0x80?0xFFFF0000:0x00000000);
76     y=((int(d[3])<<8)+d[2])|(d[3]&0x80?0xFFFF0000:0x00000000);
77     z=((int(d[5])<<8)+d[4])|(d[5]&0x80?0xFFFF0000:0x00000000);
78     //Print results
79     //printf("x y z = %02x%02x %02x%02x %02x%02x\n",d[1],d[0],d[3],d[2],d
80     [5],d[4]);
81     printf("x y z = %d %d %d\n",x,y,z);
82     usleep(500000);
83 }
84 return 0;

```

84 }

## စာရင်း ၄.J: i2cgyro.cpp

ပရိုဂရမ် ကိုအောက်ပါ အတိုင်း build လုပ်ပြီး run လိုက်တဲ့ အခါ Gyroscope ရဲ့ လှပ်ရှား မှုပေါ်မှတည်ပြီး သက်ဆိုင်ရာ ဝင်ရှိး တွေရဲ့ တန်ဖိုး တွေ လှပ်ရှား ပြောင်းလဲ နေတာ ကို ပုံ ၄.၇ မှာ ပြထားသလို တွေ့ရ မှာ ဖြစ်ပါတယ်။

```
$ g++ i2cgyro.cpp -o i2cgyro
$ ./i2cgyro
```

```
pi@raspberrypi:~/i2c-gyro $ ./i2cgyro
I am 0xd3
x y z = -104 127 103
x y z = -30 -10 -10
x y z = 12302 -507 1934
x y z = 13715 -371 3503
x y z = 839 596 -5310
x y z = 4133 -371 -8219
x y z = 10798 -25 3444
x y z = 28 49 -122
x y z = 305 -99 -285
x y z = 37 11622 20
x y z = 971 -1261 541
x y z = 35 -3831 119
x y z = -2439 6092 -909
x y z = 90 -3325 585
x y z = -904 166 -6082
x y z = 39 16 353
x y z = -441 38 -14878
x y z = 892 -407 7900
x y z = -769 328 -14197
x y z = 259 223 5870
pi@raspberrypi:~/i2c-gyro $
```

ပုံ ၄.၇: Gyroscope ဝင်ရှိးများ၏ ပြောင်းလဲမှတန်ဖိုးများ ကိုဖတ်ခြင်း။

## ၄.J SPI

SPI လို့ ခေါ်တဲ့ serial peripheral interface bus က IC chip တွေ အကွာအဝေး နီးနီးနားနားဆက်သွယ်ဖို့ သုံးနိုင် တဲ့ interface သတ်မှတ်ချက် တစ်ခု ပါ။ သူမှာ transmit လုပ်ဖို့ ဝါယာ တစ်လိုင်း receive လုပ်ဖို့ ဝါယာ တစ်လိုင်း သပ်သပ်ဖို့ ဖြစ်တဲ့ အတွက် full duplex communication

ဖြစ်ပါတယ်။ Master က slave တွေနဲ့ ဆက်သွယ်နိုင် တဲ့ master slave ပုံစံ ဖြစ်ပါတယ်။ ပုံမှန် အားဖြင့် ဝါယာ လေးခု သုံးပြီး၊ အောက်ပါ အတိုင်း ဖြစ်ပါတယ်။

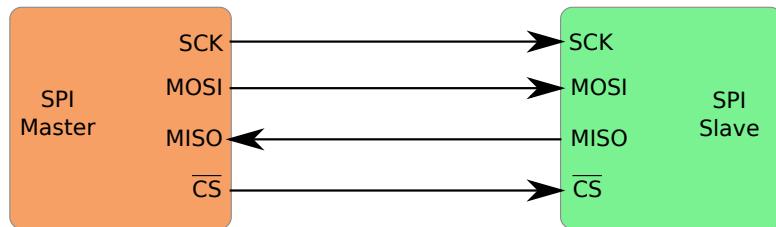
**SCK:** Serial clock (master က ထုတ်ပေးပါတယ်။)

**MOSI:** Master output slave input (master ရဲ့ data အထွက် ဖြစ်ပါတယ်။)

**MISO:** Master intput slave output (master ရဲ့ data အဝင် ဖြစ်ပါတယ်။)

**CS:** Chip select (Slave select လို့လည်း ခေါ်ပြီး master က သူ့ဆက်သွယ်ချင်တဲ့ slave ကို enable လုပ်ဖို့ ထုတ်ပေးတဲ့ active low signal ဖြစ်ပါတယ်။)

SPI bus တစ်ခု ရဲ့ နမူနာ ဆက်သွယ်မှု တစ်ခု ကို ပုံ ၄.၈ မှာ ပြထား ပါတယ် [Wik18b]။



ပုံ ၄.၈: SPI bus အတွက် master နှင့် slave ဆက်သွယ်ခြင်း။

ဒေတာ တွေ ပေးပို့ ဆက်သွယ် တဲ့ အခါ clock ရဲ့ polarity (CPOL) နဲ့ clock ရဲ့ phase (CPHA) ပေါ်မှုတည်ပြီး mode လေးမျိုး ရှိပါတယ် (ပေါ်မှု: ၄.၂, ပေါ်မှု: ၄.၃, ပေါ်မှု: ၄.၄, ပေါ်မှု: ၄.၅)။

ပေါ်မှု: ၄.၂: SPI mode 0

Mode	CPOL	CPHA	ဖော်ပြချက်
0	0	0	CPOL=0 ဖြစ်သဖြင့် clock မှာ ပုံမှန်အား ဖြင့် 0 ဖြစ်ပြီး positive pulse ကို သုံးသည်။ CPHA=0 ဖြစ်သဖြင့် data ကို leading (rising) edge တွင်ဖတ်၍ trailing (falling) edge တွင် ပြောင်းသည်။

အယား ၄.၃: SPI mode 1

Mode	CPOL	CPHA	ဖော်ပြချက်
1	0	1	CPOL=0 ဖြစ်သဖြင့် clock မှာ ပုံမှန်အား ဖြင့် 0 ဖြစ်ပြီး၊ positive pulse ကို သုံးသည်။ CPHA=1 ဖြစ်သဖြင့် data ကို leading (rising) edge တွင် ပြောင်းချုံ၊ trailing (falling) edge တွင် ဖတ်သည်။

အယား ၄.၄: SPI mode 2

Mode	CPOL	CPHA	ဖော်ပြချက်
1	1	0	CPOL=1 ဖြစ်သဖြင့် clock မှာ ပုံမှန်အား ဖြင့် 1 ဖြစ်ပြီး၊ negative pulse ကို သုံးသည်။ CPHA=0 ဖြစ်သဖြင့် data ကို leading (falling) edge တွင်ဖတ်သည့်၊ trailing (rising) edge တွင် ပြောင်းသည်။

အယား ၄.၅: SPI mode 3

Mode	CPOL	CPHA	ဖော်ပြချက်
1	1	1	CPOL=1 ဖြစ်သဖြင့် clock မှာ ပုံမှန်အား ဖြင့် 1 ဖြစ်ပြီး၊ negative pulse ကို သုံးသည်။ CPHA=1 ဖြစ်သဖြင့် data ကို leading (falling) edge တွင် ပြောင်းချုံ၊ trailing (rising) edge တွင် ဖတ်သည်။

Raspberry Pi မှာ SPI0 လို့ ခေါ်တဲ့ SPI bus တစ်ခု ပါပြီး၊ သူမှာ Chip Select နှစ်ခု ပါ ပါတယ်။ Raspberry Pi ရဲ့ SPI ပင်တွေ ကို ယေား ၄.၆ မှာ ပြထား ပါတယ်။ SPI1 ကို enable လုပ်ဖို့ အတွက် Main Menu → Preferences → Raspberry Pi Configuration → Interfaces tab ကို ဖွင့်ပြီး၊ အဲဒီ မှာ SPI ကို ပုံ ၄.၂ မှာ ပြထား သလိုမျိုး enabled လုပ် ပြီးတဲ့ အခါ reboot လုပ်ဖို့ လို ပါတယ်။

ပေါ်ပေါ်။ BeagleBone Black ၏ SPI bus များ။

Bus	Wire	Header	Device	ဖော်ပြုချက်
SPI0	MOSI	P19	MOSI0	ပုံမှန်အားဖြင့် disabled ဖြစ်နေပြီး Pi Configuration တွင် enabled လုပ်၍ သုံးနိုင် သည်
	MISO	P21	MISO0	
	SCLK	P23	SCK0	
	CS0	P24	CE0	
	CS1	P26	CE1	

## ၄.J.၁ Shell

SPI bus ကို ဒေတာ တွေ ပို့ကြည့် ဖို့ အတွက် shell မှာ အောက်က command ကို သုံးပြီး ရှိတဲ့ device တွေကို list လုပ်ကြည့်နိုင် ပါတယ်။ အဲဒီ အခါ ပေါ်လာတဲ့ /dev/spidev0.0 နဲ့ /dev/spidev0.1 ၏ SPI0 ရဲ့ CS0 နဲ့ CS1 အတွက် ဖြစ် ပါတယ်။

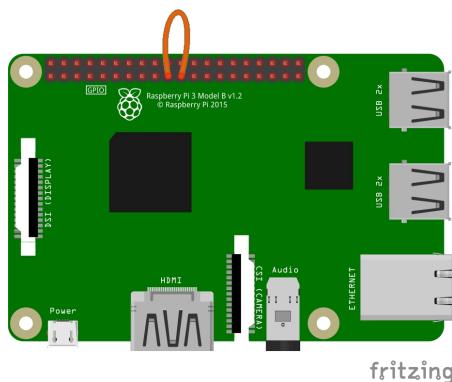
```
$ ls /dev/spi*
```

ပြီးတဲ့ အခါ echo ကို သုံးပြီး /dev/spidev0.0 ဆိုတဲ့ device file မှာ ပို့ချင်တဲ့ တန်ဖိုး ဥပမာ 1,2, နဲ့ 3 စတဲ့ hexadecimal byte တွေကို ရေးပြီး ပို့နိုင် ပါတယ် [ras17d; J15]။

```
$ echo -ne "\x01\x02\x03" > /dev/spidev0.0
```

## ၄.J.၂ C++ ဖြန့်သုံးခြင်း

ပထမ အဆင့် အနေနဲ့ SPI0 ရဲ့ MOSI (Pin19) နဲ့ MISO (Pin21) ကို အချင်းချင်း loop back ပြန်ဆက်ပြီး (ပုံ ၄.၉) C နဲ့ ဒေတာ ပို့တာ၊ လက်ခံတာ အဆင် ပြေ မဖြေ စမ်းသပ် ကြည့် ပါမယ်။



ံ ၄.၉: SPI ၅၀ ဒေတာ အထွက် နှင့် အဝင် ကို loop back ပြန် ဆက်သွယ်ခြင်း။

SPI bus ကို သုံးပြီး ဒေတာ ပိုပေး၊ လက်ခံ နိုင်တဲ့ ရိုးရှင်း တဲ့ နမူနာ C ပရိုဂရမ် spitest.c ကို စာရင်း ၄.၃ မှာ ဖော်ပြထား ပါတယ်။

```

1 // File: spitest.c
2 // Based on spidev_test.c at
3 //   https://raw.githubusercontent.com/raspberrypi/linux/rpi-3.10.y/
4 //     Documentation/spi/spidev_test.c
5
6 #include <stdint.h>
7 #include <unistd.h>
8 #include <stdio.h>
9 #include <stdlib.h>
10 #include <getopt.h>
11 #include <fcntl.h>
12 #include <sys/ioctl.h>
13 #include <linux/types.h>
14 #include <linux/spi/spidev.h>
15
16
17 static void pabort(const char *s)
18 {
19     perror(s);
20     abort();
21 }
```

```
22
23 static int fd;
24 static const char *device = "/dev/spidev0.0";
25 static uint8_t mode = SPI_MODE_0;
26 static uint8_t bits = 8;
27 static uint32_t speed = 500000;
28 static uint16_t delay = 0;
29 static uint8_t tx[] = {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x00};
30 static uint8_t rx[ARRAY_SIZE(tx)] = {0, };
31
32 static void transfer(int fd, int n)
33 {
34     int ret;
35
36     struct spi_ioc_transfer tr = {
37         .tx_buf = (unsigned long)tx,
38         .rx_buf = (unsigned long)rx,
39         .len = n,
40         .speed_hz = speed,
41         .delay_usecs = delay,
42         .bits_per_word = bits,
43     };
44
45     ret = ioctl(fd, SPI_IOC_MESSAGE(1), &tr);
46     if (ret < 1) pabort("can't send spi message");
47 }
48
49
50 int main(int argc, char *argv[])
51 {
52     int ret = 0;
53
54
55     fd = open(device, O_RDWR);
56     if (fd < 0)
57         pabort("can't open device");
```

```
58
59  /*
60   * spi mode
61   */
62  ret = ioctl(fd, SPI_IOC_WR_MODE, &mode);
63  if (ret == -1)
64      pabort("can't set spi mode");
65
66  ret = ioctl(fd, SPI_IOC_RD_MODE, &mode);
67  if (ret == -1)
68      pabort("can't get spi mode");
69
70  /*
71   * bits per word
72   */
73  ret = ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits);
74  if (ret == -1)
75      pabort("can't set bits per word");
76
77  ret = ioctl(fd, SPI_IOC_RD_BITS_PER_WORD, &bits);
78  if (ret == -1)
79      pabort("can't get bits per word");
80
81  /*
82   * max speed hz
83   */
84  ret = ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed);
85  if (ret == -1)
86      pabort("can't set max speed hz");
87
88  ret = ioctl(fd, SPI_IOC_RD_MAX_SPEED_HZ, &speed);
89  if (ret == -1)
90      pabort("can't get max speed hz");
91
92  printf("spi mode: %d\n", mode);
93  printf("bits per word: %d\n", bits);
```

```

94     printf("max speed: %d Hz (%d KHz)\n", speed, speed/1000);

95

96     transfer(fd,6);
97
98     for (ret = 0; ret < 6; ret++) {
99         printf("%.2X ", rx[ret]);
100    }
101
102    puts("");
103
104    printf("Transfer 2nd time.\n");
105    transfer(fd,6);
106
107    for (ret = 0; ret < 6; ret++) {
108        printf("%.2X ", rx[ret]);
109    }
110
111    puts("");
112
113    close(fd);
114
115
116    return ret;
117
118 }
```

စာရင်း ၄.၃: spitest.c

SPI bus ကို စတင် အသုံးပြုဖို့ အတွက် သက်ဆိုင်ရာ ”/dev/spidev0.0” အစရှိတဲ့ bus ကို ဖွင့်ရပါမယ်။ Buffer မသုံးအောင် fopen အစား open ကိုပဲ သုံးရ ပါမယ်။

```

1 char spidev []="/dev/spidev0.0";
2 fd = open((const char*)spidev, O_RDWR);
```

SPI bus ကို ဖွင့်ပြီးတဲ့ အခါ configure လုပ်ဖို့ အတွက် သူရဲ့ file descriptor ကို အောက်ပါ အတိုင်း ioctl ဖန်ရှင် သုံးပြီး သတ်မှတ်နိုင် သလို လက်ရှိ configuration ကို ကြည့်နိုင် ပါတယ်။

```

1 if(ioctl(fd, SPI_IOC_WR_MODE, &mode)==-1){
2     printf("Can't set SPI mode.\n");
3     return -1;
```

```

4 }
5
6 if(ioctl(fd, SPI_IOC_RD_MODE, &mode)==-1){
7 printf("Can't get SPI mode.\n");
8 return -1;
9 }
10
11 if(ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits)==-1){
12 printf("Can't set bits per word.\n");
13 return -1;
14 }
15
16 if(ioctl(fd, SPI_IOC_RD_BITS_PER_WORD, &bits)==-1){
17 printf("Can't get bits per word.\n");
18 return -1;
19 }
20
21 if(ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed)==-1){
22 printf("Can't set max speed hz.\n");
23 return -1;
24 }
25
26 if(ioctl(fd, SPI_IOC_RD_MAX_SPEED_HZ, &speed)==-1){
27 printf("Can't get max speed hz.\n");
28 return -1;
29 }

```

SPI bus က MOSI က ဒေတာ ထွက်တဲ့ အချိန်မှာ MISO က ဒေတာ ကို တပြုင်ထဲ ဖတ်ပါတယ်။ ဖတ်ရုံ ပဲ ဖတ်ချင်ပြီး၊ မရေးချင် ရင် transmit buffer မှာ zero တွေ ကို သုံးလေ့ ရှုပါတယ်။ Transmit buffer က ဒေတာ တွေကို ထုတ်ပြီး၊ receive buffer မှာ ဒေတာ တွေ လက်ခံ ဖို့ အတွက် spi\_ioc\_transfer ဆိုတဲ့ structure ကို သုံးပြီး အောက်ပါ အတိုင်း transfer လုပ်နိုင် ပါတယ်။

```

1 void Transfer(char* tx, char* rx, unsigned int n)

```

```

2 {
3     struct spi_ioc_transfer tr;
4     tr.tx_buf = (unsigned long)tx;
5     tr.rx_buf = (unsigned long)rx;
6     tr.len = n;
7     tr.delay_usecs = delay;
8     tr.speed_hz = speed;
9     tr.bits_per_word = bits;
10    if(ioctl(fd, SPI_IOC_MESSAGE(1), &tr)<1){
11        printf("Can't send SPI message.\n");
12    }
13 }

```

spitest.c ကို build လုပ်ပြီး run လုပ်ကြည့်ဖို့ အတွက် အောက်က command တွေကို သုံးနိုင် ပါတယ်။

```

$ gcc spitest.c -o spitest
$ ./spitest

```

အဲဒီ အခါ transmit buffer ထဲက hexadecimal နံပါတ် တွေ ပြန်ပြီး လက်ခံ ရရှိ တာကို အောက်ပါ ပုံ reffig:bus-loopback-output အတိုင်း တွေ့နိုင် ပါတယ်။ Loop back တပ်ထား တဲ့ ဝါယာ ကို ဖြတ်ပြီး ပြန် run ကြည့်ရင် ဒေတာ တွေ မရောက်တော့ တာကို တွေ့ရ မှာပါ။

```

pi@raspberrypi:~/spi-gyro $ gcc spitest.c -o spitest
pi@raspberrypi:~/spi-gyro $ ./spitest
SPI mode: 0
Bits per word: 8
Max speed: 500000 Hz
0x01 0x02 0x03 0x04 0x05 0x06
pi@raspberrypi:~/spi-gyro $ █

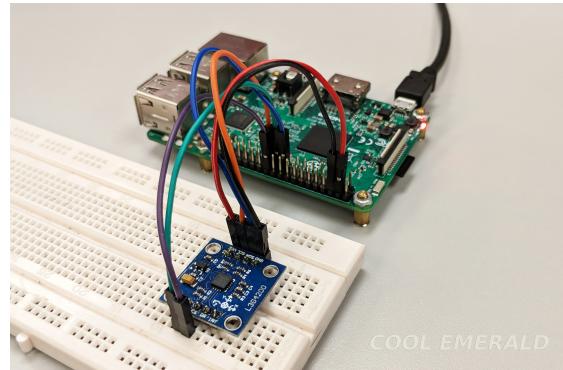
```

ပုံ ၄.၁၀: SPI ကို C ပရိုဂရမ် ဖြင့်စမ်းသပ်ခြင်း။

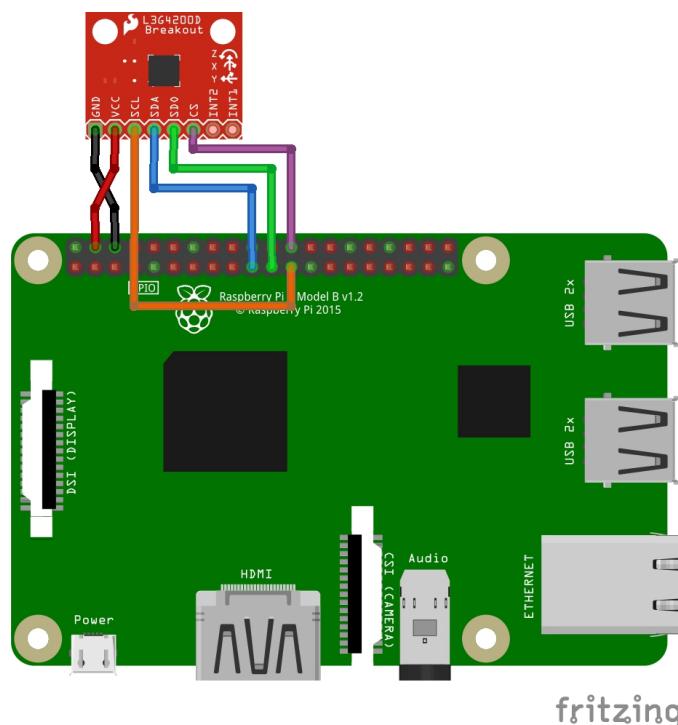
## ၄.၂ Gyroscope ကိုအသုံးပြုခြင်း

SPI0 ကို သုံးပြီး L3G4200D gyroscope ကို အသုံးပြုတဲ့ C++ နမူနာကို [spipyro.cpp](#) (စာရင်း ၄.၄) မှာ တွေ့နိုင် ပါတယ် [Mol14; Eli17b; Vor07]။ အဲဒီ အတွက် ပစ္စည်းများ ဆက်သွယ်ပုံ ကို ပုံ ၄.၁၁ နဲ့ ပုံ ၄.၁၂

မှာ တွေ့နှင့် ပါတယ်။ Module ရဲ့ SDA က SPI mode မှာ SDI ဖြစ်ပါတယ်။ အဲဒီ လိုပဲ SCL က SCLK နဲ့ pin အတူတူ ဖြစ်ပါတယ်။



ပုံ ၄.၁၁: L3G4200 Gyroscope module ကို SPI နှင့် ဆက်သွယ်ခြင်း။



ပုံ ၄.၁၂: SPI နှင့် Gyroscope ဆက်သွယ်မှု သငောက်တပ်း။

```

1 // File: spigyro.c
2 // Description: SPI communication with gyroscope

```

```
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 // References
8 // spidev_test.c @ https://raw.githubusercontent.com/raspberrypi/linux/rpi
-3.10.y/Documentation/spi/spidev_test.c
9
10 #include <stdint.h>
11 #include <unistd.h>
12 #include <stdio.h>
13 #include <stdlib.h>
14 #include <getopt.h>
15 #include <fcntl.h>
16 #include <sys/ioctl.h>
17 #include <linux/types.h>
18 #include <linux/spi/spidev.h>
19
20 static void pabort(const char *s)
21 {
22     perror(s);
23     abort();
24 }
25
26 static int fd;
27 static const char *device = "/dev/spidev0.0";
28 static uint8_t mode = SPI_MODE_0;
29 static uint8_t bits = 8;
30 static uint32_t speed = 500000;
31 static uint16_t delay = 0;
32 static uint8_t tx[] = {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x00};
33 static uint8_t rx[] = {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x00};
34
35 static void Transfer(unsigned int n)
36 {
37     int ret;
```

```
38
39     struct spi_ioc_transfer tr = {
40         .tx_buf = (unsigned long)tx,
41         .rx_buf = (unsigned long)rx,
42         .len = n,
43         .speed_hz = speed,
44         .delay_usecs = delay,
45         .bits_per_word = bits,
46     };
47
48     ret = ioctl(fd, SPI_IOC_MESSAGE(1), &tr);
49     if (ret < 1) pabort("can't send spi message");
50 }
51
52
53 int main(int argc, char *argv[])
54 {
55     int ret = 0;
56
57     //Init SPI
58     fd = open(device, O_RDWR);
59     if (fd < 0) pabort("can't open device");
60
61     ret = ioctl(fd, SPI_IOC_WR_MODE, &mode);
62     if (ret == -1) pabort("can't set spi mode");
63
64
65     ret = ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits);
66     if (ret == -1) pabort("can't set bits per word");
67
68     ret = ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed);
69     if (ret == -1) pabort("can't set max speed hz");
70
71
72     printf("spi mode: %d\n", mode);
73     printf("bits per word: %d\n", bits);
```

```

74 printf("max speed: %d Hz (%d KHz)\n", speed, speed/1000);

75

76 //Transfer

77

78 //b7 = set (1) for reading
79 //b6 = cleared (0) not to auto increase address
80 //b5-b0 = register address

81

82 //read the "who am i" register at address 0x0F
83 //its value should be 0xD3
84 tx[0]=0x8F;//read, not auto 0x0F / 0x80
85 Transfer(2);
86 for(int i=1;i<2;i++){
87     printf("I am 0x%02x",rx[i]);
88 }
89 printf("\n");

90

91 //Configure Gyro
92 //CTRL_REG2 = 10 10 | HPM1| HPM0| HPCF3| HPCF2| HPCF1| HPCF0 |
93 //Default = 10 10 10 10 10 10 10 10 10 1
94 //HPM = 00 => Normal (High Pass filter Mode selection)
95 //HPC = 1001 => 0.1 High Pass Filter Cut-off freq configuration
96 //write @address 0x21, value = 0x09
97 tx[0]=0x21;
98 tx[1]=0x09;
99 Transfer(2);

100

101 //CTRL_REG3 = I1_Int1| I1_Boot| H_Lactive| PP_OD| I2DRDY| I2_WTM| I2_ORun |
102 //I2_Empty |
103 //Default = 10 10 10 10 10 10 10 10 10 10
104 //
105 //Use Default

106

107 //CTRL_REG4 = BDU| BLE| FS1| FS0 | - | ST1| ST0| SIM|
108 //Default = 10 10 10 10 10 10 10 10 10 1
109 //BDU = 1 => Block Data Update

```

```

108 //BLE = 0 => Little endian
109 //FS = 00 => 250 dps (Full scale selection)
110 //ST = 000 => Disable Self test
111 //write @ address 0x23, value =0x80
112 tx[0]=0x23;
113 tx[1]=0x80;
114 Transfer(2);
115
116 //CTRL_REG5 = /BOOT/FIFO_EN/ - /HPen/INT1_Sel1/INT1_Sel0/Out_Sel1/
117 //Out_Sel0/
118 //Default = /0 /0 /0 /0 /0 /0 /0 /0 /0
119 /
120 //BOOT = 0 => Normal mode (Reboot Memory Content)
121 //FIFO_EN = 0 => disable FIFO
122 //HPen = 0 => disable (High Pass Filter)
123 //INT1_Sel = 00 => Non high pass filtered data are used for interrupt
124 generation
125 //Out_Sel = 00 => no filtering
126 //Use Default
127
128 //CTRL_REG1 = /DR1/DR0/BW1/BW0/PD/Zen/Yen/Xen/
129 //Default = /0 /0 /0 /0 /0 /1 /1 /1 /1 /
130 //DR = 11 => ODR 800 Hz (output data rate)
131 //BW = 10 => Cut-off 50 (Bandwidth 50 Hz)
132 //PD = 1 => Normal
133 //Zen = Yen = Xe = 1 => Enable
134 //write @ address 0x20, value =0xEF
135 tx[0]=0x20;
136 tx[1]=0xEF;
137 Transfer(2);
138
139 tx[0]=0xE0;//read ctrl registers
140 Transfer(5);//send addr 1 byte + read 4 byte
for(int i=1;i<5;i++){
    printf("Reg = 0x%02x ",rx[i]);
}

```

```

141     printf("\n");
142
143     int x,y,z;
144     for(int j=0;j<10;j++)
145     {
146         //read address 0x28, OUT_X_L register
147         //set MSB bit for auto address increment
148         tx[0]=0xE8;
149         Transfer(7);
150         x=((int(rx[2])<<8)+rx[1])|(rx[2]&0x80?0xFFFF0000:0x00000000);
151         y=((int(rx[4])<<8)+rx[3])|(rx[4]&0x80?0xFFFF0000:0x00000000);
152         z=((int(rx[6])<<8)+rx[5])|(rx[6]&0x80?0xFFFF0000:0x00000000);
153         printf("x y z = %d %d %d",x,y,z);
154         printf("\n");
155         usleep(500000);
156     }
157
158     //Close
159     close(fd);
160
161     return ret;
162 }
```

စာရင်း ၄.၄: spigo.cpp

ပရီဂရမ် ရဲ့ output ကို ပုံ ၄.၁၃ မှာ ပြထားပါတယ်။

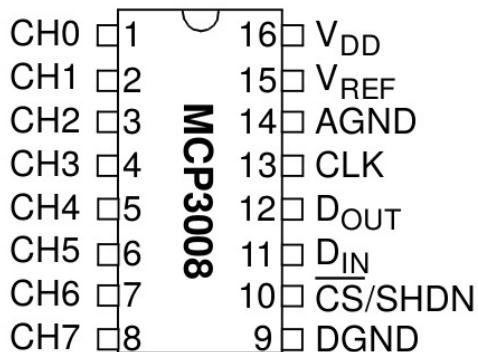
## ၄.၂ Analog အဝင်များကို ဖတ်ခြင်း

MCP3008 က Microchip က ထုတ်ထဲ 8-channel, 10-bit analog to digital converter ဖြစ်ပါတယ်။ သူကို single ended analog input အနေနဲ့ ရှုစ်ခု သံးလို့ ရပြီး၊ differential input pair အနေနဲ့ ဆိုလေးပဲ သံးနိုင် ပါတယ် [Mic07]။ Power supply မိုက 2.7 V ကနေ 5 V အထိ ရပြီး၊ အပူချိန် -40 °C ကနေ +85 °C အထိ ခံနိုင် ပါတယ်။ သူနဲ့ ဆက်သွယ် ဖို့ SPI interface ပါပြီး mode 0 ဒါမူ မဟုတ်

```
pi@raspberrypi:~/spi-gyro $ g++ spipyro.cpp -o spipyro
pi@raspberrypi:~/spi-gyro $ ./spipyro
spi mode: 0
bits per word: 8
max speed: 500000 Hz (500 KHz)
I am 0xd3
Reg = 0xef Reg = 0x09 Reg = 0x00 Reg = 0x80
x y z = 108 69 78
x y z = 74 47 41
x y z = 100 44 25
x y z = 51 35 42
x y z = 2064 -2332 2225
x y z = 352 -665 279
x y z = 300 -2806 -174
x y z = 188 -1967 684
x y z = 1622 520 334
x y z = 5837 305 4877
pi@raspberrypi:~/spi-gyro $
```

ပုံ ၄.၃၃: SPI ဖြင့် Gyroscope ၏ angular velocities များကို ဖတ်ခြင်း။

mode 3 နဲ့ သုံးလို့ ရ ပါတယ်။ MCP3008 ရဲ့ pin တွေကို ပုံ ၄.၃၄ မှာ ပြထား ပါတယ်<sup>၁</sup>။

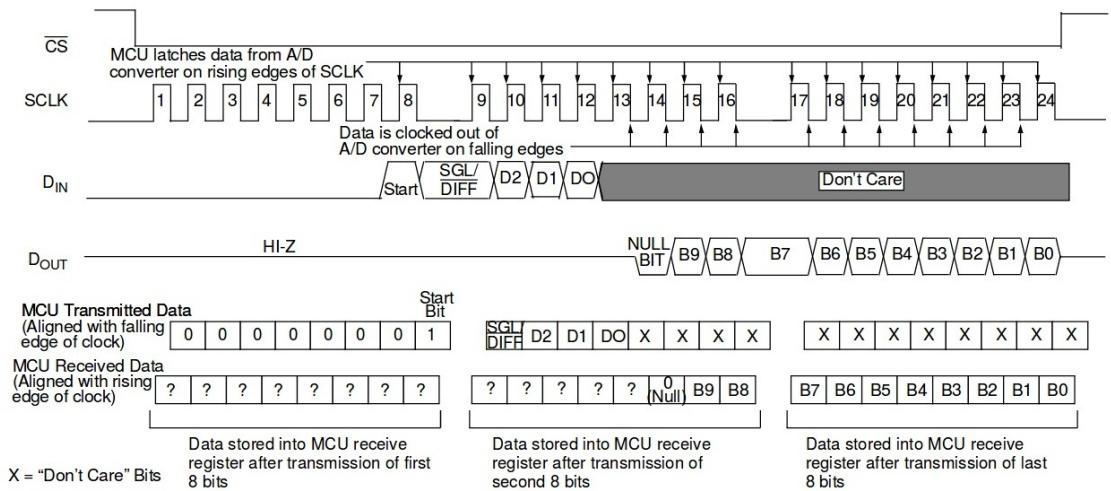


ပုံ ၄.၃၄: MCP3008 ၏ pin များ။

SPI သုံးပြီး ဆက်သွယ်တဲ့ အခါ start bit, control bits နဲ့ analog input data တွေ ဖတ်ဖို့ အတွက် 3 bytes ပိုမို လက်ခံ ဖို့ လိုပါတယ်။ SPI mode 0 သုံးတဲ့ timing diagram ကို ပုံ ၄.၃၅ မှာ ပြထား ပါတယ်။ SPI mode 3 အတွက်လည်း ပို တဲ့ လက်ခံ ရရှိ တဲ့ ဒေတာ တွေက အတူတူ ပါပဲ။

<sup>၁</sup>ပုံ ၄.၃၄, ပုံ ၄.၃၅, နှင့် ပုံ ၄.၃၆ တို့မှာ MCP3008 ၏ datasheet [Mic07] မှ ကူးယူ ထားခြင်း ဖြစ်သည်။

## ፩. SERIAL BUS ማር:



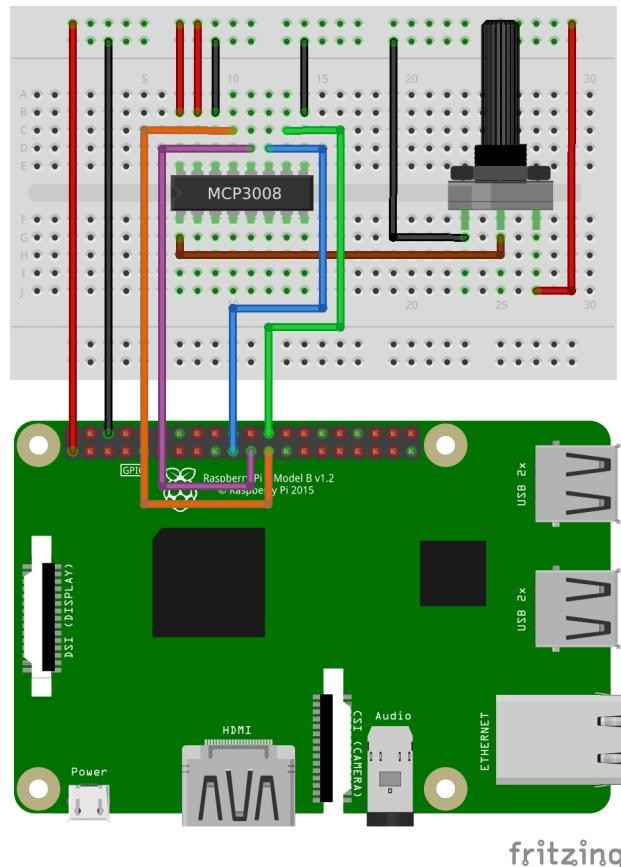
፪.፭: MCP3008 ዘዴርና SPI mode 0 ጥሩን ምርመራዎች ተከተሉዋል

Control Bit Selections				Input Configuration	Channel Selection
Single /Diff	D2	D1	D0		
1	0	0	0	single-ended	CH0
1	0	0	1	single-ended	CH1
1	0	1	0	single-ended	CH2
1	0	1	1	single-ended	CH3
1	1	0	0	single-ended	CH4
1	1	0	1	single-ended	CH5
1	1	1	0	single-ended	CH6
1	1	1	1	single-ended	CH7
0	0	0	0	differential	CH0 = IN+ CH1 = IN-
0	0	0	1	differential	CH0 = IN- CH1 = IN+
0	0	1	0	differential	CH2 = IN+ CH3 = IN-
0	0	1	1	differential	CH2 = IN- CH3 = IN+
0	1	0	0	differential	CH4 = IN+ CH5 = IN-
0	1	0	1	differential	CH4 = IN- CH5 = IN+
0	1	1	0	differential	CH6 = IN+ CH7 = IN-
0	1	1	1	differential	CH6 = IN- CH7 = IN+

፪.፮: MCP3008 ነው control bits ማር፡፡

Analog input အတွက် single ended ပါမှ မဟုတ် differential pair စား mode နဲ့ အဝင် channel ရွေးဖို့ အတွက် control bits တွေကို ပုံ ၄.၁၆ မှာ ပြထား ပါတယ်။

MCP3008 နဲ့ Raspberry Pi တို့ကို ဆက်သွယ်တဲ့ schematic နမူနာ တစ်ခု ကို အောက်က ပုံ ၄.၁၇ မှာ တွေ့နိုင် ပါတယ်။



ပုံ ၄.၁၇: Raspberry Pi နဲ့ MCP3008 ဆက်သွယ်ပုံ နမူနာ။

MCP3008 ကို အသုံးပြုပြီး analog ပိုအား အဝင် ကို ဖတ်တဲ့ နမူနာ C++ ပရိုဂရမ် တစ်ခု ကို spiai.cpp (စာရင်း ၄.၅) နဲ့ ce\_spi.h (စာရင်း ၄.၆) မှာ ဖော်ပြ ထား ပါတယ်။

```

1 // File: spiai.cpp
2 // Description: SPI communication with MCP3008 ADC
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye

```

```

6
7
8 #include <stdio.h>
9 #include <stdlib.h>
10 #include "ce_spi.h"
11 using namespace std;
12
13 int main()
14 {
15     CE_SPI m;
16     m.Begin();
17
18     int x;
19     float K=0.003225806;
20     // (3.3/1023) V per digit for +3.3V full scale
21     // using 10 bit digital output
22
23     for(int j=0;j<10;j++)
24     {
25         m.tx[0]=0x01;//Send start bit - 0000 0001
26         m.tx[1]=0x80;//read channel 0 - | s/~d | d2 d1 d0 | x x x x |
27         // b7= single/~differential = 1 for single ended
28         // d2 d1 d0 = 0 for channel 0
29         m.tx[2]=0x00; // don't care = x x x x x x x x
30         m.Transfer(3);
31         x = m.rx[1]<<8 | m.rx[2]; //read channel 0
32         x&=0x03FF;//mask out invalid bits
33         printf("x = %d \t Voltage = %f ",x,(K*x));
34         printf("\n");
35         usleep(500000);
36     }
37     return 0;
38 }
```

```
1 // File: ce_spi.h
2 // Description: SPI communication class
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 // References
8 // spidev_test.c @ https://raw.githubusercontent.com/raspberrypi/linux/rpi
9 // -3.10.y/Documentation/spi/spidev_test.c
10
11 #ifndef CESPI_H
12 #define CESPI_H
13
14 #include <stdint.h>
15 #include <unistd.h>
16 #include <stdio.h>
17 #include <stdlib.h>
18 #include <getopt.h>
19 #include <fcntl.h>
20 #include <sys/ioctl.h>
21 #include <linux/types.h>
22 #include <linux/spi/spidev.h>
23 #include <string>
24 using namespace std;
25
26 class CE_SPI
27 {
28     public:
29         CE_SPI();
30         virtual ~CE_SPI();
31         bool Begin();
32         void Transfer(unsigned int n);
33         uint8_t tx [BUF_SIZE];
34         uint8_t rx [BUF_SIZE];
```

```

35     private:
36
37         int fd;
38
39         uint8_t mode;
40
41         uint8_t bits;
42
43         uint32_t speed;
44
45         uint16_t delay;
46
47         string spidev;
48
49     };
50
51
52     CE_SPI::CE_SPI()
53 {
54
55         mode=SPI_MODE_0;
56
57         bits=8;
58
59         speed=500000; //500 kHz
60
61         delay=0;
62
63         spidev="/dev/spidev0.0";
64
65         for(int i=0;i<BUF_SIZE;i++){
66
67             tx[i]=0;
68
69             rx[i]=0;
70
71         }
72
73     }
74
75
76     CE_SPI::~CE_SPI()
77 {
78
79         close(fd);
80     }
81
82
83     bool CE_SPI::Begin()
84 {
85
86         fd = open(spidev.c_str(), O_RDWR);
87
88         if (fd < 0){
89
90             perror("Can't open spi device.\n");
91
92             abort();
93
94         }
95
96
97         if(ioctl(fd, SPI_IOC_WR_MODE, &mode)==-1){
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
918
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2137
2138
2139
2139
2140

```

```

71     perror("Can't set SPI mode.\n");
72     abort();
73 }
74
75 if(ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits)==-1){
76     perror("Can't set bits per word.\n");
77     abort();
78 }
79
80 if(ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed)==-1){
81     perror("Can't set max speed hz.\n");
82     abort();
83 }
84 return true;
85 }
86
87 void CE_SPI::Transfer(unsigned int n)
88 {
89     struct spi_ioc_transfer tr = {
90         .tx_buf = (unsigned long)tx,
91         .rx_buf = (unsigned long)rx,
92         .len = n,
93         .speed_hz = speed,
94         .delay_usecs = delay,
95         .bits_per_word = bits,
96     };
97
98     if(ioctl(fd, SPI_IOC_MESSAGE(1), &tr)<1){
99         perror("Can't send SPI message.\n");
100    }
101 }
102
103 #endif // CESPI_H

```

ပရီဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ဖြီး run လိုက်တဲ့ အခါ channel 0 နဲ့ ဆက်သွယ် ထားတဲ့ ဗို့ ပြောင်းလဲ မှုပေါ် မူတည်ဖြီး ရလဒ် တန်ဖိုး တွေ လူပ်ရှား ပြောင်းလဲ နေတာ ကို ပုံ ၄.၁၈ မှာ ပြထား သလို တွေ့ရ မှာ ဖြစ်ပါတယ်။

```
$ g++ spiai.cpp -o spiai
$ ./spiai
```

```
pi@raspberrypi:~/spi-ai $ ./spiai
x = 617          Voltage = 1.990322
x = 599          Voltage = 1.932258
x = 571          Voltage = 1.841935
x = 544          Voltage = 1.754838
x = 519          Voltage = 1.674193
x = 496          Voltage = 1.600000
x = 482          Voltage = 1.554838
x = 460          Voltage = 1.483871
x = 439          Voltage = 1.416129
x = 427          Voltage = 1.377419
pi@raspberrypi:~/spi-ai $
```

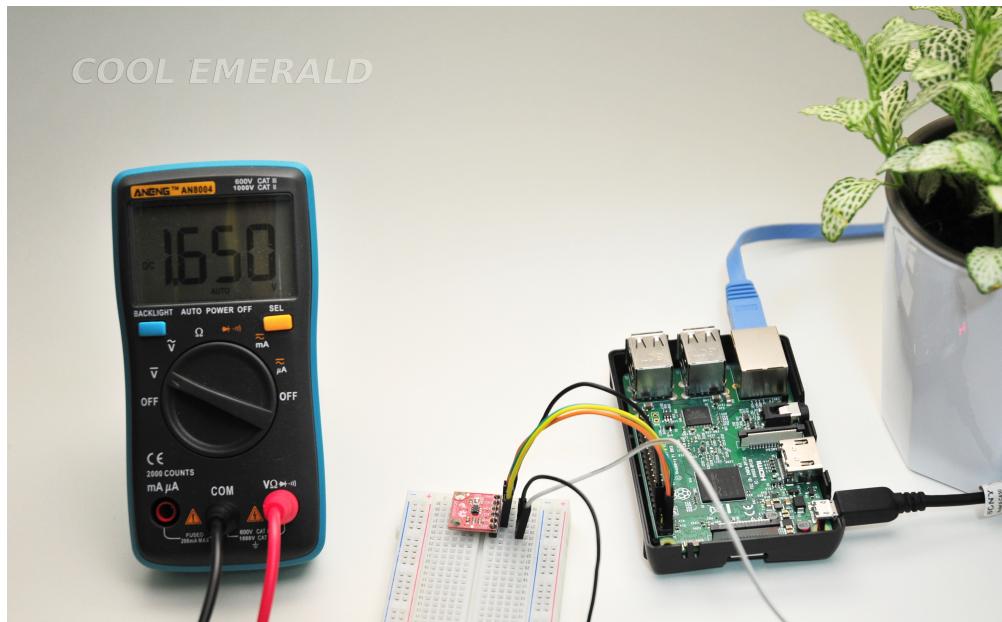
ပုံ ၄.၁၈: spiai.cpp ၏ ရလဒ် နမူနာ။

## ၄.၄ Analog အတွက် နှစ်မျိုး

Analog အတွက် တွေ ထုတ်သုံးဖို့ နည်းလမ်း နှစ်မျိုး သုံးလို့ ရပါတယ်။ ပထမ နည်းက digital to analog converter (DAC) တွေကို သုံးပြီး သူညီ ကနေ full scale voltage ကြားက ဗို့ တန်ဖိုး အမျိုးမျိုး ကို ထုတ်ပေး တာပါ။ နောက် တစ်နည်း ကတေသ့ pulse width modulation (PWM) လို့ ခေါ်တဲ့ high နဲ့ low ဗို့ နှစ်မျိုး ကို ပဲ ထွက်စေခဲ့တဲ့ ပျမ်းမျှ အား အလိုက် pulse ရဲ့ အကျယ်ကို ပြောင်း ပေးတဲ့ နည်းပါ။ ပုံမှန် linear amplifier တွေအတွက် analog ဗို့ အစစ်ကို ရနိုင် တဲ့ DAC သုံးတဲ့ နည်းက ပိုကောင်းပြီး၊ LED မီးလုံး၊ မော်တာ စတာတွေ မောင်းနှင် ဗို့ အတွက် ဆိုရင် တော့ PWM က ပို့ အဆင်ပြေ ပါတယ်။ အသုံးပြု မယ့် စနစ် အပေါ်မူ တည်ပြီး DAC တပ်ဆင် ဖို့ လိုနိုင် သလို၊ PWM တွေ ထပ်ဖြည့် ဖို့ လိုတဲ့ အခါ တွေလည်း ရှိနိုင် တာမို့ အဲဒီ အကြောင်း တွေကို ဆက် ဆွေးနွေး ပါမယ်။

### ၄.၄.၁ Digital to Analog Converter

SBC က သတ်မှတ် လိုက်တဲ့ digital တန်ဖိုး အတိုင်း analog ပို့ ထုတ်ပေး ဖို့ အတွက် Microchip က ထုတ်တဲ့ 8-channel, 10-bit analog to digital converter [MCP4725](#) စတဲ့ chip တွေကို သုံးလို့ ရပါတယ် [[Mic09](#)]။ နှမူနာ အနေနဲ့ SparkFun I2C DAC Breakout - [MCP4725](#) လေးကို သုံးပြီး Raspberry Pi နဲ့ ဆက်သွယ် လိုက်ပါမယ်။ MCP4725 ကို Raspberry Pi နဲ့ အသုံးပြု ဖို့ အတွက် 3.3 V ပါဝါ ပေးပြီး i2c နဲ့ ပုံ ၄.၁၉ မှာ ပြထား သလို ချိတ်ဆက် လိုက် ပါမယ်။ DAC ရဲ့ အထွက်  $V_{out}$  pin ကို multimeter ဒါမှ မဟုတ် oscilloscope နဲ့ ချိတ်ပြီး ကြည့်လို့ ရပါတယ်။



ပုံ ၄.၁၉: MCP4725 ကို ချိတ်ဆက်ခြင်း။

Terminal မှာ အောက်ပါ စာရင်း အတိုင်း i2c bus ကို ဖတ်ကြည့်လိုက် တဲ့ အခါ သူရဲ့ address 0x60 ကို ပုံ ၄.၂၀ အတိုင်း တွေ့နိုင် ပါတယ်။ အဲဒါ က သူရဲ့ device address pin A0 ကို ဘာမှ ဆက်မထား တဲ့ အတွက် default တန်ဖိုး 0 ဖြစ်နေ တဲ့ အချိန် ပါ။ DAC နှစ်ခု ပြိုင်တူ ဆက်မယ် ဆိုရင် နောက်တစ်ခု ရဲ့ A0 ကို  $V_{dd}$  နဲ့ ဆက်ပြီး device address မတူ အောင် လုပ်လို့ ရပါတယ်။

```
$ i2cdetect -l
$ i2cdetect -y -r 1
```

```
pi@raspberrypi:~ $ i2cdetect -l
i2c-1    i2c          bcm2835 I2C adapter
pi@raspberrypi:~ $ i2cdetect -y -r 1
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- - - - - - - - - - - - - - - - - - - - - -
10: -- - - - - - - - - - - - - - - - - - - - -
20: -- - - - - - - - - - - - - - - - - - - - -
30: -- - - - - - - - - - - - - - - - - - - - -
40: -- - - - - - - - - - - - - - - - - - - - -
50: -- - - - - - - - - - - - - - - - - - - - -
60: 60 - - - - - - - - - - - - - - - - - - - - -
70: -- - - - - - - - - - - - - - - - - - - - -
pi@raspberrypi:~ $
```

ဦး ၄.၂၀: I2C bus ကို probe လုပ်ခြင်း။

DAC ကို အသုံးပြု တဲ့ နမူနာ C++ ပရိဂရမ် `i2cao.cpp` (စာရင်း ၄.၇) မှာ `i2c` class အတွက် အရင် ရှုပြုသား `ce_i2c.h` (စာရင်း ၄.၁) ကို ပြန်သုံး ထားပါတယ်။

```
1 // File: i2cao.cpp
2 // Description: analog output with MCP4725
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 #include<stdio.h>
8 #include "ce_i2c.h"
9 using namespace std;
10 int main()
11 {
12     char d[2]={0,0};
13     CE_I2C mcp4725(1,0x60); //use i2c-1, address 0x60
14
15
16     //Write DAC reg using fast mode
17     //1st byte = /c2/c1/PD1/PD0/D11/D10/D9/D8/
18     //Default = /0 /0 /0 /0 /x /x /x /x /
19     //c2 c1 = 00 => Fast mode
20     //PD1 PD0 = 00 => Power down select (00 : normal mode)
21     //D11 D10 D9 D8 = Four MSb of 12 bit DAC value
22
23     //2nd byte = /D7/D6/D5/D4/D3/D2/D1/D0/
```

## ၄.၄. ANALOG အထွက် နှစ်မျိုး

CJ

```
24 // D7 - D0 = LSB of 12 bit DAC value
25
26 int v = 0;
27 float x=0,dx=0,steps=10.0;
28 dx = 3.3 / steps;
29 for(int i=0;i<steps;i++)
30 {
31     // d = v / 3.3 * 4095 = v * 1241 (for 12 bits, 3.3 V full scale)
32     x += dx;
33     v = int(x * 4095.0 / 3.3);
34     printf("x = %f v = %d \n", x,v);
35     d[0] = (v >> 8) & 0x0F;// 4 MSb
36     d[1] = v & 0xFF; // LSB
37     mcp4725.Write(d,2);
38     usleep(3000000);
39 }
40 return 0;
41 }
```

စာရင်း ၄.၇: i2cao.cpp

```
pi@raspberrypi:~/i2c-ao $ ./i2cao
x = 0.330000 v = 409
x = 0.660000 v = 819
x = 0.990000 v = 1228
x = 1.320000 v = 1638
x = 1.650000 v = 2047
x = 1.980000 v = 2457
x = 2.310000 v = 2866
x = 2.640000 v = 3276
x = 2.970000 v = 3685
x = 3.300000 v = 4094
pi@raspberrypi:~/i2c-ao $
```

၄.၇: i2cao.cpp ၏ output

ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး run လိုက်တဲ့ အခါ ပုံ ၄.၇ မှာ ပြထား တဲ့ အထွက် တွေအတိုင်း multimeter မှာ တိုင်းတာ လို ရတဲ့ တန်ဖိုး တွေ က လိုက်ပါ ပြောင်းလဲ နေတာ ကို တွေ့ရ မှာ ဖြစ်ပါတယ်။

```
$ g++ i2cao.cpp -o i2cao
$ ./i2cao
```

### ၄.၄.၂ PWM အထွက်များ

PWM အထွက် ထုတ်သုံးတဲ့ နမူနာ အနေနဲ့ ပါယာ နှစ်ကြိုး ပဲ လိုတဲ့ i2c interface ကို သုံးတဲ့ NXP ရဲ့ PCA9685 16-channel, 12-bit PWM ကို သုံးလိုက် ပါမယ် [NS15]။ အဲဒီ အထွက် Adafruit က ထုတ်တဲ့ [PWM and Servo driver PCA9685 breakout](#) လေးကို အလွယ် တကူး သုံးနိုင် ပါတယ်။

PCA9685 မှာ သူရဲ့ i2c device address ကိုသတ်မှတ် ဖို့ အထွက် address pin A0 - A5 ခြောက်ခဲ့ပါတဲ့ အထွက် reserved လုပ်ထား တဲ့ address တွေ ကလွှဲလို့ စုစုပေါင်း PCA9685 ဒေါ် ခု တဗြိုင်ထဲ ဆက်သွယ် အသုံးပြု လို့ ပါတယ်။ Adafruit PCA9685 breakout မှာ အဲဒီ address pin တွေက ground ကို pull-down လုပ်ထားတဲ့ အထွက် jumper တွေကို ဘာမှ မဖြူဖြဲ့ ရင် default device address က 0x40 ဖြစ် ပါတယ်။

အဲဒီ အပြင် လိုသလို enable လုပ်လို့ ရတဲ့ programmable address လေးခု ပါ ပါသေး တယ်။ ALLCALLADR လို့ ခေါ်တဲ့ ပို့လိုက် တဲ့ i2c command ကို ရှိသမျှ PCA9685 တွေက လက်ခံ မယ့် address (default 0x70) က ပုံမှန် အားဖြင့် enabled ဖြစ်နေ မှာပါ။ နောက်ထပ် RGB group တွေခဲ့ သုံးလို့ ရအောင် enabled လုပ်ထား သမျှ အုပ်စု လိုက် လုပ်ခိုင်း နိုင်တဲ့ SUBADR1, SUBADR2 , and SUBADR3 လို့ ခေါ်တဲ့ address သုံးခု (default values 0x71, 0x72, 0x74) ကတော့ ပုံမှန် အားဖြင့် disabled ဖြစ်နေ ပါတယ်။ သူတို့ ရဲ့ address တွေက programmable ဖြစ်တဲ့ အထွက် default တန်ဖိုး တွေ အစား စိတ်ကြိုက် ပြင်ဆင် သတ်မှတ် လိုလည်း ရပါတယ်။

ဒါကြောင့် အောက်က command တွေသုံးပြီး i2c bus ကို probe လုပ်ကြည့် လိုက်ရင် device address 0x40 ရယ်။ i2c bus ပေါ်မှာ ရှိသမျှ PCA9685 device အားလုံး လက်ခံမယ့် default address 0x70 ရယ် နှစ်ခု ကို ပုံ ၄.၂ အတိုင်း တွေ့ရပါမယ်။ သူရဲ့ register တန်ဖိုး တွေကို လည်း တွေ့နိုင် ပါတယ်။

```
$ i2cdetect -l
$ i2cdetect -y -r 1
$ i2cdump -y 1 0x40
```

## ፭.፭. ANALOG အတွက် နှစ်မျိုး

၁၂၃

```
pi@raspberrypi:~ $ i2cdetect -l
i2c-1  i2c          bcm2835 I2C adapter
pi@raspberrypi:~ $ i2cdetect -y -r 1
  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: --
10: --
20: --
30: --
40: 40
50: --
60: --
70: 70
pi@raspberrypi:~ $ i2cdump -y 1 0x40
No size specified (using byte-data access)
  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f      0123456789abcdef
00: 11 04 e2 e4 e8 e0 00 00 00 10 00 00 00 10 00 00  ??????...?...?..
10: 00 10 00 00 00 10 00 00 00 10 00 00 00 10 00 00  .?...?...?...?..
20: 00 10 00 00 00 10 00 00 00 10 00 00 00 10 00 00  .?...?...?...?..
30: 00 10 00 00 00 10 00 00 00 10 00 00 00 10 00 00  .?...?...?...?..
40: 00 10 00 00 10 XX  .?...?XXXXXXXXXX
50: XX XXXXXXXXXXXXXXXXXX
60: XX XXXXXXXXXXXXXXXXXX
70: XX XXXXXXXXXXXXXXXXXX
80: XX XXXXXXXXXXXXXXXXXX
90: XX XXXXXXXXXXXXXXXXXX
a0: XX XXXXXXXXXXXXXXXXXX
b0: XX XXXXXXXXXXXXXXXXXX
c0: XX XXXXXXXXXXXXXXXXXX
d0: XX XXXXXXXXXXXXXXXXXX
e0: XX XXXXXXXXXXXXXXXXXX
f0: XX 00 00 00 00 1e 00 XXXXXXXXXXXX...?.
pi@raspberrypi:~ $
```

ဦး ፭.၂၂: PCM9685 အတွက် I2C bus ကို probe လုပ်ခြင်း။

Adafruit PCA9685 breakout မှာ #OE pin က ground ကို pull-down လုပ်ထားတဲ့ အတွက် သူ့ကို ဘာမှ မပေး ရင် output တွေက enabled ဖြစ်နေ ပါယ်။

## PWM Frequency

ထွက်လာ မယ့် PWM waveform ရဲ့ frequency ကို address 0xFE က PRE\_SCALE register မှာ သတ်မှတ် လို ရပါ တယ်။ Oscillator frequency,  $f_o$ , က internal oscillator အတွက် ဆိုရင် 25 MHz ဖြစ် ပါတယ်။ PWM frequency ကို  $f_p$  လို ခေါ်မယ် ဆိုရင် သတ်မှတ် ရမယ့် prescaler တန်ဖိုး  $N_{ps}$  ကို အောက်က ညီမျှခြင်း ፭.၁ နဲ့ ရှာဖို့ ပါတယ်။

$$N_{ps} = \lfloor \frac{f_o}{4096 \times f_p} \rfloor - 1 \quad (၇.၁)$$

အဲဒီ မှာ  $\lfloor N \rfloor$  က rounding လုပ်တဲ့ operation ပါ။ ဉာပမာ PWM frequency 50 Hz ရဲ့ အတွက် ဆိုရင်

$$N_{ps} = \lfloor \frac{25000000}{4096 \times 50} \rfloor - 1 \quad (၇.၂)$$

## C++ ဖြင့် ထိန်းချုပ်ခြင်း

PCA9685 ကို အသုံးပြု ဖြစ်၍ frequency 50 Hz, 25% duty cycle ရှိတဲ့ PWM waveform ထုတ်ပေး တဲ့ နှမူနာ C++ ပရိုဂရမ် ကို [i2c-pwm.cpp](#) (စာရင်း ၄.၈) မှာ ဖော်ပြ ထားပါတယ်။

```
1 // File: i2c-pwm.cpp
2 // Description: PWM output with PCA9685
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 #include<stdio.h>
8 #include"ce_i2c.h"
9 using namespace std;
10 int main()
11 {
12     char d[8] = { 0,0,0,0,0,0,0,0 };
13     CE_I2C pca9685(1,0x40); //use i2c-1, address 0x40
14
15     //MODE1 register, address 0x00
16     //MODE1      = /Restart/ExtClk/AI /SLEEP/SUB1/SUB2/SUB3/ALLCALL /
17     //Default   = 10      10      10  11      10      10      10      11      /
18     //Restart   = 0 => Restart disabled
19     //ExtClk    = 0 => External clock disabled
20     //AI        = 1 => Auto increment enabled
21     //SLEEP     = 0 => Normal mode (don't sleep)
22     //SUB1     = 0 => Do not respond to sub address group 1
23     //SUB2     = 0 => Do not respond to sub address group 2
24     //SUB3     = 0 => Do not respond to sub address group 3
```

```

25 //ALLCALL = 1 => Respond to all call address
26
27
28 //To sleep (default value for MODE1 Register)
29 d[0] = 0;// address
30 d[1] =0x11;
31 pca9685.Write(d,2);
32 usleep(1000); //wait a while
33
34 //Set frequency 50 Hz
35 //PRE_SCALE register, address 0xFE
36 //N= [25000000/(4096*f)]-1 =121 = 0x79
37 d[0]=0xFE;
38 d[1]=0x79;
39 pca9685.Write(d,2);
40 usleep(1000); //wait a while
41
42 //Go back to Normal mode
43 //MODE1 register, address 0x00
44 d[0] = 0;// address
45 d[1] = 0x21;
46 pca9685.Write(d,2);
47 //Need at least 500us to wake up from sleep
48 usleep(1000);
49
50 //Output settings
51 //MODE2 register, address 0x01
52 //MODE2 = /Reserved /INVRT/OCH/OUTDRV/OUTNE/
53 //Default = 10 10 10 10 10 11 100 /
54 //Reserved=000=> Reserved
55 //INVRT = 0 => Output logic state not inverted
56 //OCH = 0 => Output change on stop command instead of ACK
57 //OUTDRV = 1 => push-pull output instead of open-drain
58 //OUTNE = 00 => output 0 when output is disabled (i.e. LEDn = 0 when #OE
59 = 1)
d[0] = 1;// address

```

```

60 //Value to set = 0000 0100
61 d[1] = 0x04;
62 pca9685.Write(d,2);

63
64 //Control PWM
65 //LED on = on at 0 of 0-4095 clock cycles
66 //LED off = off at 1024 of 0-4095 clock cycles
67 d[0] = 0x06; //address of LED0 ON_L is at 0x06
68 d[1] = 0x00; //ON_L
69 d[2] = 0x00; //ON_H
70 d[3] = 0x00; //OFF_L
71 d[4] = 0x04; //OFF_H - (Note: overwriting full off bit 4 to 0)
72 //OFF_H = 0x10 means always off,
73 //always off has higher priority than always on
74 pca9685.Write(d,5);
75 return 0;
76 }
```

### စာရင်း ၄.၈: i2c-pwm.cpp

အဲဒီ ပရိုကရမဲ မှာ duty cycle အမျိုးမျိုး ပြောင်းထုတ် ကြည့်ပြီး၊ အထွက် ကို LED မီးလုံး တစ်လုံး ပုံ ၄.၂၃ မှာ ပြထား သလို ဆက်ပြီး အကြမ်း ဖျင်း ကြည့်နိုင် ပါတယ်။ PCA9685 က source current 10 mA ထုတ်ပေး နိုင်ပြီး၊ 25 mA sink လုပ်ပေး နိုင်ပါ တယ်။ Adafruit PCA9685 breakout ရဲ့ အထွက် တွေမှာ 220 Ω limiting resistor ပါတဲ့ အထွက် LED ကို တိုက်ရှိက် ဆက်လို ရပါတယ်။

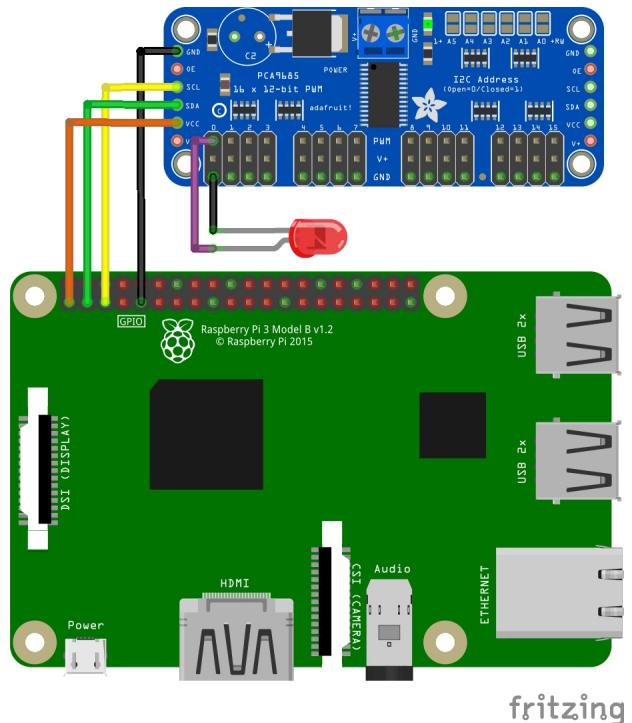
ပရိုကရမဲ ကို အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run လိုက်တဲ့ အခါ LED မီးလုံး ရဲ့ အလင်း၊ အမြိုန်က duty cycle သတ်မှတ်မှု အတိုင်း ရလာမှာ ဖြစ်ပြီး၊ oscilloscope နဲ့ ချိတ်ဆက် တိုင်းတာ ကြည့်တဲ့ အခါ ပုံ ၄.၂၄ မှာ ပြထား တဲ့ အထွက် တွေအတိုင်း တွေ့ရ ပါတယ်။

```

$ g++ i2c-pwm.cpp -o i2c-pwm
$ ./i2c-pwm
```

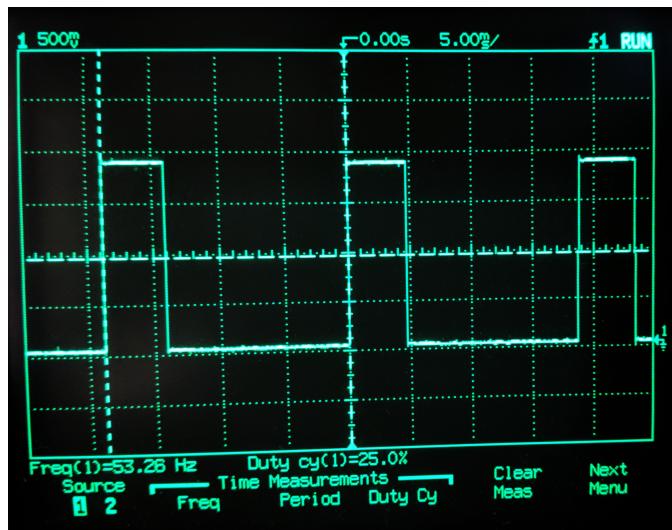
## ၄.၄. ANALOG အထွက် နှစ်မိုး

၂၂



fritzing

ံ ၄.၂၃: PCA9685 ကို Raspberry Pi, LED မီးလုံး တို့ဖြင့် ဆက်သွယ်ပဲ။



ံ ၄.၂၄: i2c-pwm.cpp ၏ အထွက် ကို oscilloscope ဖြင့် တိုင်းတာ တွေ့ရှိ ရပဲ။

## အကိုးအကားများ

- [Ada14] Adam. Using the I2C interface. 2014. url: <http://www.raspberry-projects.com/pi/programming-in-c/i2c/using-the-i2c-interface>.
- [Eli12] Elinux. Interfacing with I2C Devices. 2012. url: [http://elinux.org/Interfacing\\_with\\_I2C\\_Devices](http://elinux.org/Interfacing_with_I2C_Devices).
- [Eli17b] Elinux. RPi SPI. 2017. url: [https://elinux.org/RPi\\_SPI](https://elinux.org/RPi_SPI).
- [J15] Byron J. Raspberry Pi SPI and I2C Tutorial. 2015. url: <https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial>.
- [Mic07] Microchip. MCP3004/MCP3008 2.7V 4-Channel/8-Channel 10-Bit A/D Converters with SPI Serial Interface. 2007. url: <http://ww1.microchip.com/downloads/en/DeviceDoc/21295C.pdf>.
- [Mic09] Microchip. MCP4725 - 12-Bit Digital-to-Analog Converter with EEPROM Memory. 2009. url: <http://ww1.microchip.com/downloads/en/DeviceDoc/22039d.pdf>.
- [Mol14] Derek Molloy. Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. Wiley, 2014. isbn: 1118935128. url: <http://www.exploringbeaglebone.com/>.
- [Mol16] Derek Molloy. Exploring Raspberry Pi: Interfacing to the Real World with Embedded Linux. Wiley, 2016. isbn: 9781119188687. url: <http://www.exploringrpi.com/>.
- [NS15] NXP-Semiconductors. PCA9685 - 16 channel, 12-bit PWM I2C-bus LED controller. 2015. url: <https://www.nxp.com/docs/en/data-sheet/PCA9685.pdf>.
- [ras17d] raspberrypi.org. SPI. 2017. url: <https://www.raspberrypi.org/documentation/hardware/raspberrypi/spi/>.

- [Vor07] Anton Vorontsov. SPI testing utility (using spidev driver). 2007. url: [https://raw.githubusercontent.com/torvalds/linux/master/tools/spi/spidev\\_test.c](https://raw.githubusercontent.com/torvalds/linux/master/tools/spi/spidev_test.c).
- [Wik18a] Wikipedia. I<sup>2</sup>C. 2018. url: <https://en.wikipedia.org/wiki/I%C2%B2C>.
- [Wik18b] Wikipedia. Serial Peripheral Interface Bus. 2018. url: [https://en.wikipedia.org/wiki/Serial\\_Peripheral\\_Interface\\_Bus](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus).
- [STM10] STMicroelectronics. MEMS motion sensor: ultra-stable three-axis digital output gyroscope. 2010. url: <http://www.mouser.com/ds/2/389/13g4200d-954834.pdf>.

## အခန်း ၅

# စက်အမြင်

သင့်၏ စက်တွင် ကင်မရာ တပ်ဆင်ပြီး ဓာတ်ပုံ၊ ဗိုဒ္ဓိယို ရှိက်ကူး ခြင်းများ၊ ပုံရှိပြုစပ်ခြင်း (image processing) များ၊ စက်အမြင် (machine vision) နှင့်ဆိုင်သော အသုံးပြုမှု များ အတွက် OpenCV ကို သုံးနိုင် သည်။

OpenCV သည် ပညာရေး အတွက် သာမက၊ စီးပွားရေး အတွက်ပါ အလကား သုံးစွဲခွင့် ရှိသော free software တစ်ခု ဖြစ်ပြီး၊ BSD license နှင့် ထုတ်သည် [Ope17d]။ C/C++၊ Python၊ Java များနှင့် တွဲဖက် အသုံးပြု နိုင်ပြီး Windows၊ Linux၊ Mac OS များ အပေါ်တွင် သာမက iOS၊ Android အစ ရှိသော mobile platform များ အတွက်ပါ အလုပ် လုပ်သည်။ OpenCV ကို ထိရောက် မြန်ဆန်စွာ တွက်ချက်မှု နှင့် အချိန် နှင့် အမျှ အရေးကြီး သော real-time အသုံးချမှု များ အတွက် အမိက ထားကာ ဒီဇိုင်း ထုဆစ် ပြုလုပ် ထားသည်။ C/C++ သုံး၍ ရေးထား သဖြင့် multi-core processing များ၏ ကောင်းကွက် ကိုလည်း အသုံးချ နိုင်သည်။ OpenCV ကိုအနုပညာ လုပ်ငန်း များမှ အစ၊ မိုင်းရှားခြင်း၊ မြေပုံများ ချိတ်ဆက်ခြင်း၊ အဆင့်မြင့် စက်ရုပ်များ ဖန်တီးခြင်း အထိ နယ်ပယ် အမျိုးမျိုး တွင် ကျပ်ကျယ် ပြန်ပြန် အသုံးပြု နေကြ သည်။

## ၅.၁ နောက်ခံအကြောင်းအရာ

OpenCV ၏ အဓိပ္ပာယ် မှာ Open Source Computer Vision Library ဖြစ်၍ မူရင်းကုဒ် များအား ဖွင့်ပြ ထားသည်။ ကွန်ပျူးတာ နှင့် စက်အမြင် ဆိုင်ရာ လုပ်ငန်း များတွင် ကိုးကား အသုံးပြု စရာ ဆော်ဖို့ library ဖြစ်သည်။ OpenCV ကို စတင် ဖန်တီးစဉ် မှစ၍ ကွန်ပျူးတာ နှင့် စက်အမြင် ဆိုင်ရာ အများ သူငါ အလွယ် တကူ အသုံးပြု နိုင်သည့် ဘုံယန္တရား တစ်ခု ရရန် ရည်ရွယ်သည်။ စီးပွားရေး ထုတ်ကုန်

များတွင်ပါ စက်အမြင် ဆိုင်ရာ အပိုင်း များတွင် တိုးတက်မှ ရှိစေရန် ဖြစ်သည်။ BSD-license နှင့် ပေးထားသည့် အတွက် စီးပွားရေး လုပ်ငန်း များတွင် အလွယ် တကူ ယူသုံး ပြုပြင် နိုင်သည်။

ဤ library တွင် အကောင်းဆုံး ရေစွဲရန် ချိန်ညီ ထားသော နည်းလမ်း (algorithm) ပေါင်း ၂၅၀၀ ကျော် ရှိသည်။ နည်းဟန်းများ မှုစဉ် နောက်ဆုံးပေါ် နည်းလမ်းသစ် များအထိ ပြည့်စုံ နှစ်ပွား ပါဝင် သည်။ ထို နည်းလမ်း များကို မျက်နှာ များကို ရှာဖွေ သိရှိခြင်း၊ မှတ်မိခြင်း (face detection and recognition)၊ အရာဝါး များကို ခွဲခြား သိရှိခြင်း (object identification)၊ ဗြိဒ္ဓယုံး အရာ ၀၎ၢ၊ များနောက် လိုက်ခြင်း (tracking moving objects)၊ သုံးဘက်မြင် မော်တယ် ထုတ်ခြင်း (3D modeling)၊ ရုပ်ပုံ များကို ဆက်ခြင်း (stitching)၊ သိမ်းထား သည့် ပုံများ ထဲမှ တူသော ပုံကို ရွှေ့ထုတ်ခြင်း၊ ရွှေ့ခြင်း များကို မှတ်မိခြင်း၊ ပြည့်စွက် အာရုံရိပ် (augmented reality) များ အတွက် အမှတ် အသား များ လုပ်ခြင်း တို့တွင် အသုံးပြု နိုင်သည်။

## ၅.၂ ဖွဲ့စည်းပုံ

OpenCV ကို အပိုင်း (module) များ ခွဲ၍ တည်ဆောက် ထားသည်။ ထို့ကြောင့် များစွာသော shared သို့မဟုတ် static libraries များ ပါဝင် နေသည်။ အခါက အသုံး များသော အပိုင်း များမှာ အောက်ပါ အတိုင်း ဖြစ်သည် [Ope17c]။

**core** အခြေခံ အချက် အလက် ပုံစံများ၊ multi-dimensional array Mat နှင့် အခြား အပိုင်းများ အတွက်ပါ လိုအပ်သော အခြေခံ လုပ်ဆောင်ချက် များ (core functionalities) ပါဝင်သည့် ကျေစ်လစ် သည့် အခြေခံ ကျသော အပိုင်း ဖြစ်သည်။

**imgproc** ရုပ်ပုံ များကို ပုံစွဲကြ ပြုပြင်သော ဖယ်တာ (filter) များ၊ ပုံ အရွယ် အစား၊ အမြင် ပြောင်းခြင်း များ၊ အရောင် ပြောင်းခြင်း များ၊ histogram အစ ရှိသည်တို့ ပါဝင်သော ပုံရိပ် ပြုစပ်ခြင်း (image processing) အပိုင်း ဖြစ်သည်။

**video** ဗြိဒ္ဓယုံး များကို ခွဲခြမ်း သုံးသပ်၍ ရွှေ့လျားမှု ကို ခန်းမှန်း ခြင်းများ၊ နောက်ခံ မြင်ကွင်း ကို ဖယ်ဖျောက် ခြင်းများ၊ အရာ ၀၎ၢ၊ များ နောက် ခြေရာခံ ခြင်း များ တို့ ပါဝင်သည့် အပိုင်း ဖြစ်သည်။

**calib3d** ကင်မရာ ချိန်ညီခြင်း (camera calibration)၊ object pose ခန်းမှန်း ခြင်း၊ stereo correspondence algorithms၊ သုံးဘက် မြင်ကွင်း ပြန် တည်ဆောက် ခြင်း (3D reconstruction) တို့ ပါဝင်

သည်။

**features2d** ပုံများရှိ အဓိက အသွင် အပြင် (salient feature) များကို ရှာဖြင့် နှင့် descriptor matcher များ ပါဝင်သည်။

**objdetect** သတ်မှတ် ထားသော အရာ ဝို့ များကို ရှာဖြင့် (ဥပမာ မျက်နှာ (face detection)၊ မျက်လုံး၊ မတ်ချက်၊ လူ၊ ကား၊ စသည် များ)။

**highgui** ရိုးရှင်းသည့် UI (user interface) လုပ်ဆောင်မှု များအတွက် အလွယ် တကူ သုံးနိုင်သော interface ဖြစ်သည်။

**Video I/O** ဗိုဒ္ဓယို များ ဖမ်းယူ ခြင်း၊ နှင့် video codecs များ ပါဝင် သည်။

**gpu** OpenCV module များ အတွက် GPU accelerated algorithm များ ပါဝင် သည်။

အခြား အပိုင်း များလည်း ရှိသေး သည်။ လက်ရှိ OpenCV ကို တည်ဆောက် ထားပုံမှာ fully re-enterable ဖြစ် သဖြင့် function တစ်ခု ကို မတူညီ သော thread များမှ ပြောင်တူ သုံးနိုင် သည်။

## ၅.၃ တပ်ဆင်ခြင်း

### ၅.၃.၁ ရှိထားရန်လိုအပ်သည့် packages များ

OpenCV ကို Linux တွင် တပ်ဆင် ရန် ပထမ အဆင့် အနေနှင့် အောက်ပါ packages များ စက်ထဲ တွင် ရှိရန် လိုအပ် သည် [[Ope17b](#); [Eme17](#)]။

1. GCC 4.4.x or later
2. CMake 2.6 or higher
3. Git
4. GTK+2.x or higher, including headers (libgtk2.0-dev)
5. pkg-config
6. Python 2.6 or later and Numpy 1.5 or later with developer packages (python-dev, python-numpy)

7. ffmpeg or libav development packages: libavcodec-dev, libavformat-dev, libswscale-dev
8. [optional] libtbb2 libtbb-dev
9. [optional] libdc1394 2.x
10. [optional] libjpeg-dev, libpng-dev, libtiff-dev, libdc1394-22-dev

ထို packages များအား စက်ထဲသို့ ထည့်သွင်း လိုပါက Synaptic Manager သုံး၍ သော လည်းကောင်း၊ terminal တွင် အောက်ရှိ စာရင်း ၅.၁ နှင့် ၅.၂ ပါ command များ ရိုက်နှုပ်၍ သော လည်းကောင်း ထည့်နိုင် သည်။

```

1 $ sudo apt update
2 $ sudo apt install build-essential
3 $ sudo apt install cmake git libgtk-3-dev pkg-config libavcodec-dev
   libavformat-dev libswscale-dev

```

စာရင်း ၅.၁: OpenCV အတွက် လိုအပ်သော packages များ ရယူခြင်း။

```

1 $ sudo apt install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev
   libpng-dev libtiff-dev libdc1394-22-dev
2 $ sudo apt install libv4l-dev libxvidcore-dev libx264-dev

```

စာရင်း ၅.၂: OpenCV အတွက် optional packages များ ရယူခြင်း။

### ၅.၃.၂ apt ဖြင့်တပ်ဆင်ခြင်း

OpenCV ကို Linux တွင် ရယူ တပ်ဆင် ရန် လွယ်ကူ ရှိုးရှင်း သော နည်းလမ်း တစ်ခု အဖြစ် အောက်ပါ အတိုင်း terminal တွင် ရိုက်ယူ နိုင်သည်။

```
$ sudo apt install libopencv-dev
```

## ၅.၃.၃ Source မှ build လုပ်ခြင်း

ထိုသို့ မဟုတ်ပဲ လက်ရှိ OpenCV အခြေကျ ဗာရွင်း ကို ရယူ တပ်ဆင်လို ပါက [OpenCV for Linux/Mac \(https://opencv.org/\)](https://opencv.org/) တွင် ရယူ ရန်လိုသည်။ ထိုမှ ရလာသော zip ဖိုင်အား Archive Manager သုံး၍ extract လုပ်နိုင်သည်။

ထို အခြေကျ ဗားရွင်း ကို မယူပဲ နောက်ဆုံးထွက် cutting-edge opencv ဗားရွင်း ကို ရယူ မည် ဆိုပါက Git repository ရှိ [OpenCV repository](#) တွင် ရယူ နိုင်သည်။ [OpenCV contrib repository](#) များ ကိုပါ တပ်ဆင် မည် ဆိုပါက လည်း အောက်ပါ အတိုင်း ယူနိုင် သည်။

```
$ cd ~
$ git clone https://github.com/opencv/opencv.git
$ git clone https://github.com/opencv/opencv_contrib.git
```

ဤနေရာတွင် ရလာသည့် folder မှာ opencv ဖြစ်သဖြင့် ထို နေရာသို့ သွား၍ build ဆိုသည့် folder တစ်ခု ဖန်တီးကာ ဖိုင်များ ထုတ်၍ သိမ်းဆည်းရန် အောက်ပါ စာရင်း ၅.၃ ရှိ command များ ရှိက်မည်။ Shared libs ကို unset လုပ်ချင် ပါက စာရင်း ၅.၄ ရှိ option ကို ထည့်နိုင် သည်။

```
1 $ cd ~/opencv
2 $ mkdir build
3 $ cd build
4 $ cmake -D CMAKE_BUILD_TYPE=Release \
5 -D CMAKE_INSTALL_PREFIX=/usr/local \
6 -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules ...
```

စာရင်း ၅.၃: OpenCV ကို build လုပ်ခြင်း။

```
1 -D BUILD_SHARED_LIBS=OFF
```

စာရင်း ၅.၄: Static libs ပြုလုပ်ခြင်း။

ခေါ် စောင့်ဆိုင်း ပြီးသည့် အခါ စာရင်း ၅.၅ အတိုင်း build လုပ်၍ install လုပ်မည်။ တပ်ဆင် ပြီးသည့် OpenCV ဗားရွင်း ကို စာရင်း ၅.၆ တွင် ပြထား သော command ဖြင့် ကြည့် နိုင်သည်။

```
1 $ make
```

```
2 $ sudo make install
```

စာရင်း ၅.၅: OpenCV ကို install လုပ်ခြင်း။

```
1 $ pkg-config --modversion opencv
```

စာရင်း ၅.၆: OpenCV ဘားရှင်းကို စစ်ခြင်း။

## ၅.၃.၄ GCC၊ CMake တိုဖြင့် အသုံးပြုခြင်း

OpenCV ကို သုံးရန် အလွယ်ဆုံး နည်းမှာ CMake ဖြင့် သုံးခြင်း ဖြစ်သည် [Ope17e]။ CMake နှင့် မရင်းနှီးပါက CMake tutorial (<https://cmake.org/cmake-tutorial/>) တွင် သွားရောက်လေ့လာ နိုင်သည်။

ပထမ ခြေလှမ်း အနေ နှင့် ပုံတစ်ပုံ ကို ဖတ်၍ ပြသည့် ရိုးရှင်းသည့် နမူနာ လေးအား စမ်းသပ် ကြည့်မည်။ ထို အတွက် thiri.jpg ဆိုသည့် ဓာတ်ပုံကို home folder တွင် ထားလိုက် မည်။ ထိုနောက် စာရင်း ၅.၇ တွင် ဖော်ပြ ထားသည့် disp.cpp ဆိုသည့် ပရီဂရမ် လေးအား ဖန်တီး လိုက်မည်။ ပြီးသည့် အခါ opencv စသဖြင့် ထဲတွင် ဖွံ့ဖြိုးထိန်းသွေးလိုက်မည်။

```
1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main(int argc, char** argv )
5 {
6     Mat image;
7     image = imread( "./thiri.jpg", 1 );
8     if ( !image.data ) {
9         printf("No image data \n");
10        return -1;
11    }
12    namedWindow("Display Image", WINDOW_AUTOSIZE );
13    imshow("Display Image", image);
14    waitKey(0);
15    return 0;
```

16

}

စာရင်း ၅.၈: ပုံ တစ်ပုံ ကို ဖတ်၍ ပြသည့် disp.cpp ပရီဂရမ်။

ဤနေရာ တွင် thiri.jpg အတွက် path မှာ ”/home/yan/thiri.jpg” ဖြစ်ပြီး သင့်ပုံ ရှိသည့် folder ၏ username တိုနှင့် ကိုက်ညီသည့် path ကို အစားထိုး ရမည်။ imread သည် လမ်းကြောင်း ပေးလိုက်သည့် ပုံဖိုင်ကို ဖတ်သည်။ ဒုတိယ argument ဖြစ်သည့် 1 မှာ ကာလာပုံ ဖတ်မည် ဟု ဆိုလိုခြင်း ဖြစ်သည်။ 0 ဆိုပါက အဖြူ အမည်း ပြောင်း၍ ဖတ်မည်။ ပုံကို ဖတ်၍ မရ ပါက message ရှိကြပြ၍ ထွက်သွားမည် ဖြစ်ပြီး၊ ဖတ်၍ အောင်မြင် ပါက imshow ကိုသုံး၍ ပုံကို ထုတ်ပြမည် ဖြစ်သည်။ တဖန် CMakeLists.txt ဆိုသည့် ဖိုင်ကို အောက်ပါ စာရင်း ၅.၉ အတိုင်း ဖန်တီးမည်။

```
1 cmake_minimum_required(VERSION 2.8)
2 project( disp )
3 find_package( OpenCV REQUIRED )
4 add_executable( disp disp.cpp )
5 target_link_libraries( disp ${OpenCV_LIBS} )
```

စာရင်း ၅.၉: CMakeLists.txt

ဤဘွင် လိုချင်သည့် ပရီဂရမ် ကို အောက်ပါ စာရင်း ၅.၉ ရှိ command များအား terminal တွင် ရိုက်ခြင်းဖြင့် ထုတ်လုပ် ရရှိနိုင်၊ run ကြည့်နိုင် သည်။ GUI application ဖြစ်သည့် အတွက် VNC ကို သုံး၍ run ဖိုလို သည်။

```
1 $ cd opencv
2 $ cmake .
3 $ make
4 $ gksudo ./disp
```

စာရင်း ၅.၉: DisplayImage ကို CMake ဖြင့် build လုပ်၍ run ခြင်း။

## pkg-config

CMake ကို မသုံးပဲ pkg-config ဖြင့် စာရင်း ၅.၁၀ အတိုင်း build လုပ်၍ လည်း run နိုင်သည်။

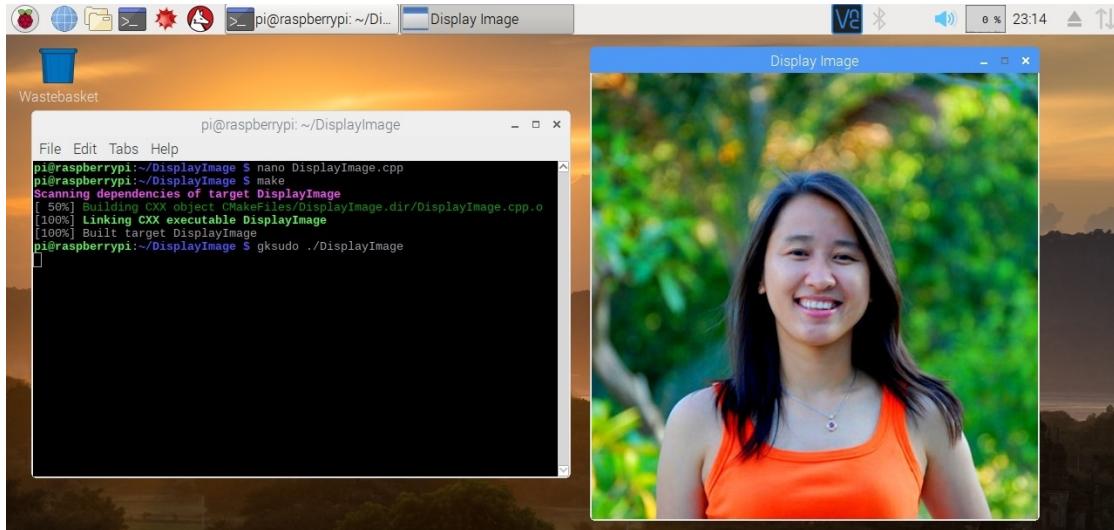
```

1 $ g++ disp.cpp `pkg-config --cflags --libs opencv` -o disp
2 $ gksudo ./disp

```

စာရင်း ၅.၁၀: DisplayImage ကို g++, pkg-config တိုဖြင့် build လုပ်၍ run ခြင်း။

ထိုနောက် တွင် ရလာသော binary ကို run လိုက်သည့် အခါ ပုံ ၅.၁ အတိုင်း ရရှိမည်။



ပုံ ၅.၁: DisplayImage ပရိုဂရမ် ၏ ရလာ၌ ကို terminal နှင့် ယူလိုက်ပြထားသည်။

Shared lib အား ရှာ မရ သည့် error ရခဲ့လျှင် /etc/ld.so.conf.d/opencv.conf ဟူသည့် ဖိုင်အား အောက်ပါ အတိုင်း ဖန်တီး နိုင်သည်။

```
$ sudo nano /etc/ld.so.conf.d/opencv.conf
```

ပြီးသည့် အခါ opencv ကို တပ်ဆင် ထားသည့် နေရာ ပေါ်မူတည်၍

```
/usr/local/opencv/
```

သို့မဟုတ်

```
/usr/local/lib/
```

ကို opencv.conf တွင် ဖြည့်မည်။ ထို့နောက် အောက်ပါ command အား terminal တွင် ရှိက်ထည့် နိုင်သည်။

```
$ sudo ldconfig
```

## ၅.၄ ကင်မရာကိုသုံးခြင်း

OpenCV တွင် ဗိုလ်ချုပ်များကို ဖမ်းယူရန် သို့မဟုတ် ဖတ်ရန် VideoCapture ဟုခေါ်သည့် class ပါရှိသည် [Ope16]။ ဗိုလ်ချုပ် ဖိုင်များ ရေးသားရန်အတွက်ကို မူ VideoWriter Class ကိုသုံးနိုင်သည် [Ope17f]။ ဗိုလ်ချုပ် တစ်ခု ကို ပုံရှိပ် များအား အစီအစဉ် လိုက် ပေါင်းစပ် ပြုလုပ် ထားသည် ဟု ဆိုနိုင်သည်။ ထိုသို့ ဖွဲ့စည်း ထားသည့် ပုံရှိပ် တစ်ခု ခြင်းစီ ကို ပြကွက် (frame) ဟု ခေါ်ပြီး၊ တစ် စက္န် အတွင်း ပြောင်းလဲ သွားသည့် ပြကွက် အရေး အတွက် ကို ပြနှိန်း (frame rate) ဟု ခေါ်မည် [Lag14]။



ပုံ ၅.၂: Logitech C922 Pro Stream Webcam

နမူနာ အနေနှင့် USB Webcam တစ်ခု ကို စက်တွင် တပ်ဆင်၍ ထို ကင်မရာ မှ ပုံရှိပ် များကို ဖမ်းယူ ၍ ပြုသ ကာ ဗိုလ်ချုပ် ဖိုင် အနေနှင့်လည်း သိမ်းဆည်း ပေးသည့် ပရိုဂရမ် တစ်ခု ကို ဖော်ပြုမည်။ ဤ နမူနာ

တွင် Logitech C922 Pro Stream Webcam (ပုံ ၅.၂) ကို အသုံး ပြုထားပြီး အခြား အဆင်ပြေ ရာ webcam များအား လည်း အသုံးပြု နိုင် သည်။ Video capture ပြုလုပ် သည့် ပရိုဂရမ် video-capture.cpp ကို စာရင်း ၅.၁၁ တွင် ဖော်ပြု ထားသည်။

```

1 #include <stdio.h>
2 #include <iostream>
3 #include <opencv2/opencv.hpp>
4
5 using namespace std;
6 using namespace cv;
7
8 int main(int argc, char** argv)
9 {
10     VideoCapture cap(0); //Default camera
11     //VideoCapture cap("./sample.mp4"); //open video file
12
13     if (!cap.isOpened())
14     {
15         printf("Video is not opened. \n");
16         return -1;
17     }
18     else
19     {
20         printf("Video is opened. \n");
21     }
22
23     union { int v; char c[5]; } uEx;
24     uEx.v = static_cast<int>(cap.get(CAP_PROP_FOURCC));
25     uEx.c[4] = '\0';
26     printf("Codec: %s \n", uEx.c);
27
28     Size S = Size((int)cap.get(CAP_PROP_FRAME_WIDTH), (int)cap.get(
29     CAP_PROP_FRAME_HEIGHT));
30     printf("Frame size: %d x %d \n", S.width, S.height);
31
32     double rate = cap.get(CAP_PROP_FPS); //Frame rate
33     printf("Frame rate: %f \n", rate);
34     int dperiod = 1000 / rate;

```

```

32 //dperiod = 1;
33 int ex = VideoWriter::fourcc('M', 'J', 'P', 'G');
34 //int ex = VideoWriter::fourcc('X', 'V', 'I', 'D');//https://www.xvid.com
35 /
36 //int ex = VideoWriter::fourcc('X', '2', '6', '4');
37 //int ex = -1;//pop up window to choose
38
39 const string vpath="./capture.avi";
40 VideoWriter outputVideo(vpath, ex , rate, S, true);
41 if (!outputVideo.isOpened())
42 {
43     cout << "Could not open the output video to write."<< endl;
44     waitKey(5000);
45     return -1;
46 }
47
48 Mat frame;
49 for (int i = 0;; i++) {
50     if (!cap.read(frame)) break;
51     outputVideo << frame;
52     imshow("Frame", frame);
53     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
54 }
55 cap.release();
56 outputVideo.release();
57 //waitKey(5000);
58 return 0;
59 }
```

စာရင်း၂၁၁: Video capturing

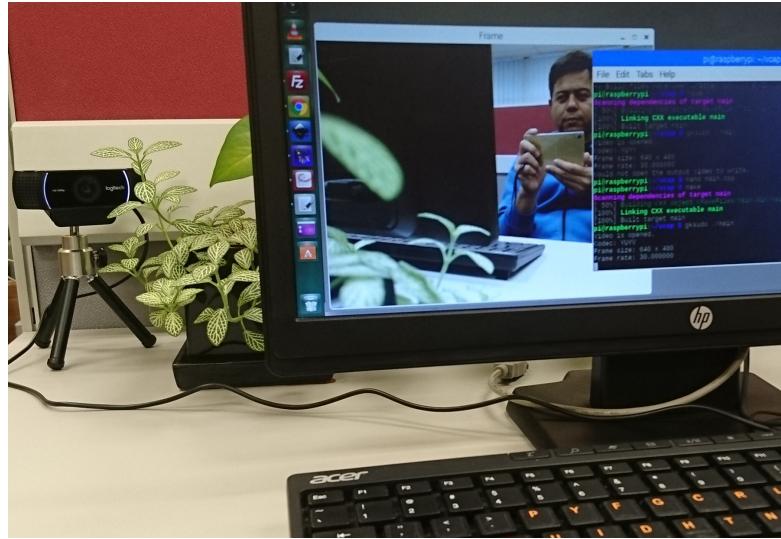
ပရိုဂရမ် အစ ရှိ VideoCapture အတွက် object တစ်ခု ကို initialize ပြလုပ် ရာတွင် ဖြိုဒီယို ဖိုင် တစ်ခု ၏ path ကို ထည့်ပေး ပါက ဖိုင်ကို ဖွင့်ဖတ် ပေးမည် ဖြစ်ပြီး၊ ၀ ကို အသုံးပြု ပါက စက်၏ default ကင်မရာ ကို ဖတ် မည် ဖြစ်သည်။ ထိုနောက် ဖြိုဒီယို ဖွင့်ခြင်း အဆင်ပြီ မပြု မပြု isOpened ဆိုသည့်

method ဖြင့် စစ်ဆေး နိုင်သည်။

လက်ရှိ ဗီဒီယို ၏ setting များကို get ဟူသည့် method သုံးကာ ဖတ်ကြည့် နိုင်ပြီး၊ set ကို သုံး၍ ပြုပြင် နိုင်သည်။ ဗီဒီယို ၏ Codec ကို get(CAP\_PROP\_FOURCC) ဟု ဖတ်နိုင် သည်။ ထို get method ၏ return အမျိုးအစား မှာ double ဖြစ်သည်။ Codec မှာ character လေးလုံး သုံး သဖြင့် union ကို သုံး၍ ဖတ်နိုင် သည်။ ထိုနောက် ပြောက် အရွယ် အစား၊ ပြန်နှုန်း တို့ကို CAP\_PROP\_FRAME\_WIDTH , CAP\_PROP\_FRAME\_HEIGHT , CAP\_PROP\_FPS တို့ ဖြင့် ဖတ်သည်။

ဆက်လက်၍ ကင်မရာ မှ ဖတ်၍ ရရှိသော ပုံရှိပ် များကို ဗီဒီယို ဖိုင် အနေဖြင့် သိမ်းရန် VideoWriter တစ်ခု ကို ဖန်တီး သည်။ Initialize ပြုလုပ် ရာတွင် ဖိုင်ကို သိမ်းဆည်း မည့် path ကို ထည့်ပေး နိုင်သည်။ ထိုနောက် codec အတွက် fourcc ဟု ခေါ်သည့် 4-character code ကို ထည့်ပေး သည်။ ဤ နမူနာ တွင် codec ကို motion-jpeg codec သုံးရန် VideoWriter::fourcc('M', 'J', 'P', 'G') ဟု ရယူ သည်။ ထိုထို မဟုတ်ပဲ <https://www.xvid.com> တွင် free ရယူ နိုင် သော XVID စသည့် codec တို့ကို VideoWriter::fourcc('X', 'V', 'I', 'D') အစ ရှိသဖြင့် သတ်မှတ် နိုင်သည်။ စက်၏ တွက်ချက်နိုင် စွမ်းမြင့်ပါက ဖိုင် အရွယ်အစား ပို သေးငယ် သော VideoWriter::fourcc('X', '2', '6', '4') ကိုလည်း ရွေးနိုင် သည်။ အကယ်၍ ထို argument အတွက် -1 ကို သုံးပါက GUI ဖြင့် ရွှေးချယ်ရန် pop-up window ပေါ်လာ မည်။

နောက်ပိုင်း loop ထဲတွင် VideoCapture ၏ read ဖြင့် frame များကို တစ်ခု ခြင်း ဖတ်၍ VideoWriter ဖြင့် ရေးသည်။ waitKey သည် သတ်မှတ် ထားသည့် milliseconds အချိန် ရပ်စောင့်၍ ထည့်သွင်း ရိုက်ထည့် သည့် key တစ်ခု ကို စောင့်ဖတ် ပေးသည်။ ထိုသို့ ဖတ်နိုင် ရန် imshow ဖြင့် ပြထား သည့် ပြောက် ဝင်းခွဲး မှာ active ဖြစ်နေရန် လို သည်။ Escape key ၏ တန်ဖိုး 27 ကို ဖတ်၍ ရပါက loop မှ ထွက်၍ ပရိုကရမဲ အဆုံးသတ် သည်။ Webcam တစ်ခု ကို Raspberry Pi တွင် opencv ဖြင့် ချိတ်ဆက် အသုံးပြု နေပုံ ကို ပုံ ၅.၃ တွင် ဖော်ပြ ထားသည်။



ပုံ ၅.၃: Webcam ကို Raspberry Pi ဖြင့် ချိတ်ဆက် အသုံးပြုခြင်း။

## ၅.၅ Real-time Face Detection

ဗိုဒ္ဓယို အတွင်း မှ မျက်နှာ များအား အချိန် နှင့် တပြေးညီ ရှာဖွေခြင်း ကို ပြုလုပ် မည်။ ထိုအတွက် Cascade Classifier [Ope17a] ကို သုံးနိုင် သည်။ နမူနာ facedetection.cpp ပရိ ကရမ ကို အောက်ပါ စာရင်း ၅.၅ တွင် တွေ့နိုင် သည်။

```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 #include<string>
4 using namespace std;
5 using namespace cv;
6
7 string face_cascade_name = "./haarcascade_frontalface_alt.xml";
8 string eyes_cascade_name = "./haarcascade_eye_tree_eyeglasses.xml";
9 CascadeClassifier face_cascade;
10 CascadeClassifier eyes_cascade;
11
12 void detectAndDisplay(Mat frame)
13 {
14     std::vector<Rect> faces;
```

```

15 Mat frame_gray;
16
17 cvtColor(frame, frame_gray, COLOR_BGR2GRAY);
18 equalizeHist(frame_gray, frame_gray);
19
20 face_cascade.detectMultiScale(frame_gray, faces, 1.1, 2, 0 |
21     CV_HAAR_SCALE_IMAGE, Size(120, 120)); // Detect faces
22
23 for (size_t i = 0; i < faces.size(); i++)
24 {
25     Point center(faces[i].x + faces[i].width*0.5, faces[i].y + faces[i].
26     height*0.5);
27     ellipse(frame, center, Size(faces[i].width*0.5, faces[i].height*0.5), 0,
28     0, 360, Scalar(255, 0, 255), 4, 8, 0);
29
30     Mat faceROI = frame_gray(faces[i]);
31     std::vector<Rect> eyes;
32
33     eyes_cascade.detectMultiScale(faceROI, eyes, 1.1, 2, 0 |
34         CV_HAAR_SCALE_IMAGE, Size(30, 30)); // In each face, detect eyes
35
36     for (size_t j = 0; j < eyes.size(); j++)
37     {
38         Point center(faces[i].x + eyes[j].x + eyes[j].width*0.5, faces[i].y +
39         eyes[j].y + eyes[j].height*0.5);
40         int radius = cvRound((eyes[j].width + eyes[j].height)*0.25);
41         circle(frame, center, radius, Scalar(255, 0, 0), 4, 8, 0);
42     }
43
44 imshow("Frame", frame);
45 }
46
47 int main(int argc, char** argv)
{
    //Load the cascades
    if (!face_cascade.load(face_cascade_name)) { printf("--(!)Error loading\n") }
```

```

        ; return -1; };

48 if (!eyes_cascade.load(eyes_cascade_name)) { printf("--(!)Error loading\n")
    ; return -1; };

49

50 VideoCapture cap(0); //Default camera
51 if (!cap.isOpened()) { printf("Video is not opened. \n"); return -1; }
52 else { printf("Video is opened. \n"); }

53

54 union { int v; char c[5]; } uEx;
55 uEx.v = static_cast<int>(cap.get(CAP_PROP_FOURCC));
56 uEx.c[4] = '\0';
57 printf("Codec: %s \n", uEx.c);

58

59 //cap.set(CAP_PROP_FRAME_WIDTH, 640);
60 //cap.set(CAP_PROP_FRAME_HEIGHT, 480);
61 //cap.set(CAP_PROP_FPS, 30);
62 Size S = Size((int)cap.get(CAP_PROP_FRAME_WIDTH), (int)cap.get(
    CAP_PROP_FRAME_HEIGHT));
63 printf("Frame size: %d x %d \n", S.width, S.height);

64

65 double rate = cap.get(CAP_PROP_FPS); //Frame rate
66 printf("Frame rate: %f \n", rate);
67 int dperiod = 33;

68

69 Mat frame;
70 for (int i = 0;; i++) {
71     if (!cap.read(frame)) break;
72     detectAndDisplay(frame);
73     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
74 }
75 cap.release();
76 //waitKey(5000);
77 return 0;
78 }

```

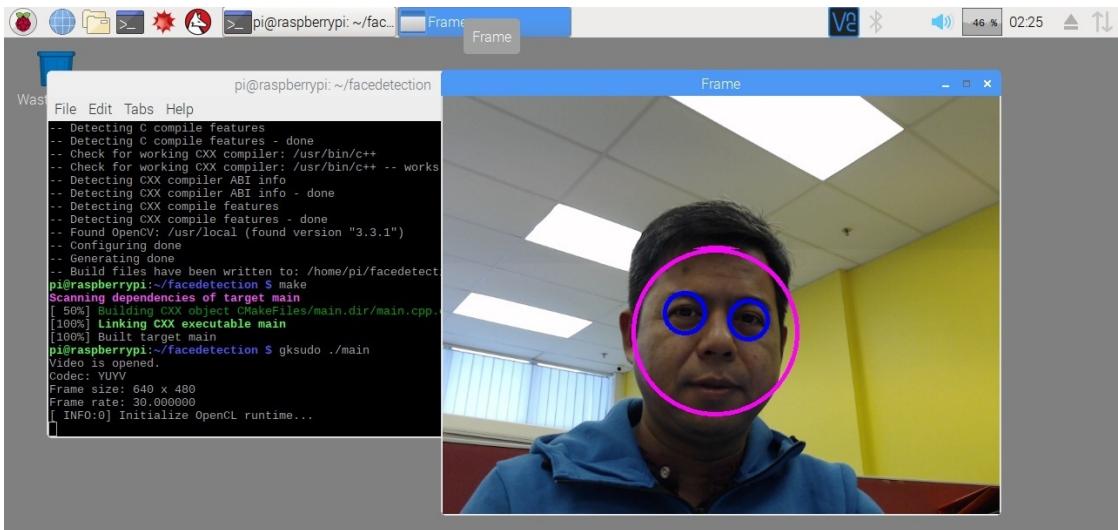
ဤ ပရိုဂရမ် တွင် မျက်နှာကို ရှာဖွေရန် “haarcascade\_frontalface\_alt.xml” နှင့် မျက်လုံး များအား ရှာဖွေရန် “haarcascade\_eye\_tree\_eyeglasses.xml” ဆိုသည့် ဖိုင်များ အား သုံးပါမည်။ ထို ဖိုင်များ သည် “/home/pi/opencv/data/haarcascades” အောက်တွင် ရှိသည်။ မျက်နှာ နှင့် မျက်လုံးများ အတွက် CascadeClassifier object နှစ်ခုကို ကြော် ပြီး load ကိုသုံး၍ အဆိုပါ ဖိုင် များအား ဖတ်သည်။

အချိန် နှင့် တပြေးညီ ဖွံ့ဖြိုးပိုက်ရန် VideoCapture object ကို ကြော် သည့် အခါ ဖိုင် နာမည် အစား 0 ကို သုံးသည့် အတွက် သင့် စက် တွင် တပ်ဆင် ထားသော ကင်မရာ ကို သုံးသည်။ VideoCapture ၏ get ကို သုံး၍ လက်ရှု ဖွံ့ဖြိုးပို ၏ setting များကို ဖတ်နိုင်ပြီး၊ set ကို သုံး၍ လိုသလို ပြင်နိုင် သည်။ ဤ ပရိုဂရမ် တွင် ပြောက် အရွယ် အစား နှင့် ပြနှုန်းကို set နှင့် ပြင်လိုက် သည်။ ထိုနောက် ပြောက် တစ်ခုခြင်း ဖတ်၍ detectAndDisplay ဖန်ရှင်ကို သုံးကာ မျက်နှာ ကို ရှာ၍ ပိုင်းပြသည်။

detectAndDisplay ဖန်ရှင် တွင် ပုံရိပ်ကို အဖြူ အမည်း ပြောင်း၍ detectMultiScale ကို သုံး၍ ရှာသည်။ သူ၏ ပုံစံမှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

```
void CascadeClassifier::detectMultiScale(const Mat& image,
vector<Rect>& objects, double scaleFactor=1.1,
int minNeighbors=3, int flags=0,
Size minSize=Size(), Size maxSize=Size())
```

ပထမ Parameters ဖြစ်သည့် image မှာ ရှာလို သည့် ပုံရိပ် ဖြစ်ပြီး၊ ဒုတိယ objects တွင် ရှာတွေ သည့် နေရာ များ၏ စတုဂံ များအား သိမ်းပေး မည်။ ထိုနောက် ရှာတွေသည့် နေရာ နှင့် အရွယ် များကို သုံး၍ circle နှင့် စက်ဝိုင်း များ ဆွဲ၍ ပုံကို imshow နှင့် ပြပေး သည်။ အောက်ပါ ပုံ ၅.၄ တွင် တချိန်နှင့် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း နှင့် ရလာ သည့် ရလဒ် တို့ကို ပြထား သည်။



ပုံ ၅.၄: တခိုန် နှင့် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း။

## ၅.၆ RaspiCam

Raspberry Pi မှာ camera module တပ်ဆင် ဖို့ အတွက် connector ပါပြီး၊ webcam အစား camera module ကို သုံးပြီး ပုံရှိပို့ တွေကို ရယူ လိုလည်း ရ ပါတယ် [Sal17]။ အဲဒီ အတွက် RaspiCam ဆိုတဲ့ C++ API ကို <https://sourceforge.net/projects/raspicam/files/> ကနေ ရယူနိုင် ပါတယ်။ ဒီ နှမူနာ မှာတော့ **raspicam-0.1.6.zip** ကို သုံးထား ပါတယ်။ ဖိုင်ကို uncompress လုပ်ပြီး compile လုပ်ဖို့ အတွက် အောက်က စာရင်း ၅.၃၃ မှာ ပြထားတဲ့ command တွေကို သုံးနိုင် ပါတယ်။

```
1 $ unzip raspicam-0.1.6.zip
2 $ cd raspicam-0.1.6
3 $ mkdir build
4 $ cd build
5 $ cmake ..
```

စာရင်း ၅.၃၃: RaspiCam C++ API ကို install လုပ်ခြင်း။

OpenCV ရှိနေ တယ် ဆိုရင် ပုံ ၅.၅ မှာ ပြထား သလို

```
-- CREATE OPENCV MODULE=1
```

လို့ပြနေ ပါမယ်။

```

pi@raspberrypi: ~/raspicam-0.1.6/build
File Edit Tabs Help
--enable- -g3 -O0 -DDEBUG -D_DEBUG -W -Wextra -Wno-return-type -lpthread
-- CMAKE_CXX_FLAGS:           -std=c++0x -Wl,--no-as-needed -Wall -ffunction-sections
-- CMAKE_BINARY_DIR:          /home/pi/raspicam-0.1.6/build
--
-- CMAKE_SYSTEM_PROCESSOR = armv7l
-- BUILD_SHARED_LIBS = ON
-- BUILD_UTILS = ON
-- CMAKE_INSTALL_PREFIX = /usr/local
-- CMAKE_BUILD_TYPE = Release
-- CMAKE_MODULE_PATH = /usr/local/lib/cmake;/usr/lib/cmake
--
-- CREATE_OPENCV_MODULE=1
-- CMAKE_INSTALL_PREFIX=/usr/local
-- REQUIRED_LIBRARIES=/opt/vc/lib/libmmal_core.so;/opt/vc/lib/libmmal_util.so;/opt/vc/lib/libmmal.so
--
-- Change a value with: cmake -D<Variable>=<Value>
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/pi/raspicam-0.1.6/build
pi@raspberrypi:~/raspicam-0.1.6/build $
```

ပုံ ၅.၅: Raspicam ကို တပ်ဆင်ခြင်း။

အဲဒီ နောက် အောက်က အတိုင်း build လုပ်၊ install လုပ်ပြီး လိုအပ်သော ldconfig ကို update လုပ်လိုက် ပါမယ်။

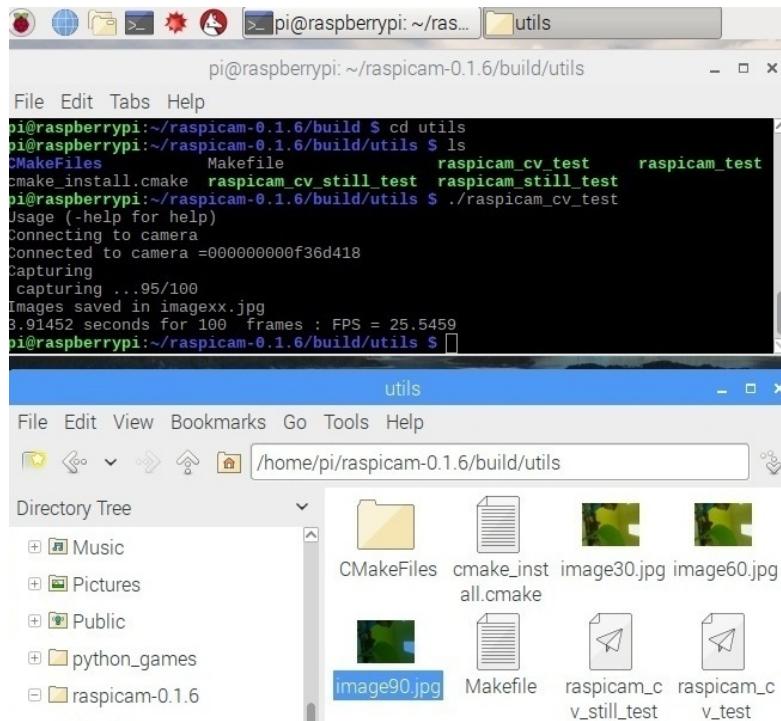
```

$ make
$ sudo make install
$ sudo ldconfig
```

OpenCV ရှိနေတယ် ဆိုရင် utils ဆိုတဲ့ အခန်း ထဲမှာ raspicam\_cv\_test ဆိုတဲ့ နမူနာ ပရိုဂရမ် ရှိနေ မှာ ဖြစ်ပြီး တပ်ဆင်မှု အဆင်ပြု မပြု စမ်းကြည့်ဖို့ အတွက် အောက်က အတိုင်း run ကြည့်နိုင် ပါတယ်။ ပရိုဂရမ် output နဲ့ ပုံရိပ် တွေ အဲဒီ အခန်း ထဲမှာပဲ ရလာ တာကို ပုံ ၅.၆ မှာ ပြထား ပါတယ်။

```

$ cd utils
$ ./raspicam_cv_test
```



ပုံ ၅.၆: RaspiCam ကို စစ်ဆေးခြင်း။

## ၅.၆.၁ OpenCV နှင့် သုံးခြင်း

RaspiCam ကို သုံးဖို့ အတွက်

```
#include <raspicam/raspicam_cv.h>
```

ဆိုပြီး သူအတွက် header ဖိုင် ကို ထည့်ဖို့ လိုပါတယ်။ Webcam အတွက် VideoCapture class ကို သုံးခဲ့ပေမယ့် Raspberry Pi ရဲ့ camera module အတွက် တော့ RaspiCam\_Cv class ကို သုံးပါမယ်။

```
raspicam::RaspiCam_Cv cap;
```

ပုံရိပ် ကို ဖမ်းယူ ဖို့ အတွက် grab နဲ့ retrieve ဆိုတဲ့ method တွေကို သုံးနိုင် ပါတယ်။ RaspiCam ကို OpenCV နဲ့ တွဲ သုံးတဲ့ ရိုးရှင်းတဲ့ နမူနာ `raspicamcv.cpp` ကို စာရင်း ၅.၁၄ မှာ ပြထား ပါတယ်။ Camera ရဲ့ တန်ဖိုး တွေကို set နဲ့ ပြောင်းလဲ နိုင်ပြီး၊ ပြောင်းတဲ့ နမူနာ တွေကို utils အခန်း ထဲက `raspicam_cv_test.cpp` ထဲမှာ တွေ့နိုင် ပါတယ်။ ပြောင်းပြီး တဲ့ အခါ release လုပ်ပြီး၊ ပြန် open လုပ်ဖို့

လုပါတယ်။

```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 #include <raspicam/raspicam_cv.h>
4
5 using namespace std;
6 using namespace cv;
7
8 int main(int argc, char** argv)
9 {
10     raspicam::RaspiCam_Cv cap;
11     if (!cap.open()) { printf("RaspiCam is not opened. \n"); return -1;}
12     else { printf("RaspiCam is opened. \n");}
13     Mat frame;
14     for (int i=0;;i++) {
15         cap.grab();
16         cap.retrieve(frame);
17         imshow("Frame", frame);
18         if (waitKey(40) == 27) break; //if 'Esc' key is pressed
19     }
20     cap.release();
21     return 0;
22 }
```

စာရင်း ၂.၁၄: RaspiCam ကို OpenCV ဖြင့် တွဲသုံးသည့် နမူနာ raspicamcv.cpp

နမူနာ ပရိုဂရမ် ကို အောက်က အတိုင်း build လုပ်ပြီး run နိုင် ပါတယ်။

```

$ g++ raspicamcv.cpp `pkg-config --cflags --libs opencv` -o raspicamcv -I/usr
/local/include/ -lraspicam -lraspicam_cv
$ ./raspicamcv
```

## ၅.၆.၂ CMake ဖွင့် Build လုပ်ခြင်း

CMake နဲ့ build လုပ်မယ် ဆိုရင်တော့ CMakeLists.txt ဆိုတဲ့ ဖိုင်ကို စာရင်း ၅.၁၅ မှာ ပြထား သလို ဖန်တီး ပြီး သိမ်းလိုက် ပါမယ်။

```

1 cmake_minimum_required (VERSION 2.8)
2 project (raspicamcv)
3 set(raspicam_DIR "/usr/local/lib/cmake")
4 find_package(raspicam REQUIRED)
5 find_package(OpenCV)
6 IF ( OpenCV_FOUND AND raspicam_CV_FOUND )
7 MESSAGE(STATUS "COMPILING")
8 add_executable (raspicamcv raspicamcv.cpp)
9 target_link_libraries (raspicamcv ${raspicam_CV_LIBS})
10 ELSE()
11 MESSAGE(FATAL_ERROR "OPENCV NOT FOUND IN YOUR SYSTEM")
12 ENDIF()
```

စာရင်း ၅.၁၅: RaspiCam အတွက် CMakeLists.txt

ပြီးတဲ့ အခါ အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run နိုင် ပါတယ်။

```

$ mkdir build
$ cd build
$ cmake ..
$ make
$ ./raspicamcv
```

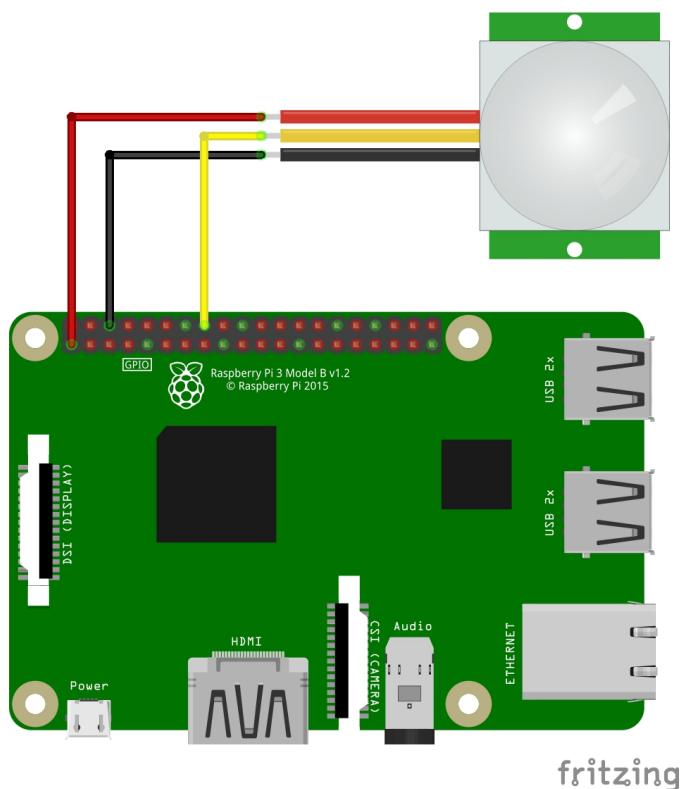
## ၅.၇ Smart Surveillance ကင်မရာပြုလုပ်ခြင်း

RPi ကို သုံးပြီး smart surveillance ကင်မရာ တစ်ခု ကို ကိုယ်တိုင် ပြုလုပ် ဖန်တီး ကြည့်ပါမယ်။ ပုံမှန် ဘာမှ ထူးခြားမှု မရှိတဲ့ အချိန်မှာ 5x speed နဲ့ နေ့စဉ်၊ ဒါမှ မဟုတ် နာရီ၊ အလိုက် ဗိုလ်ယို ဖိုင်တွေ ခဲ့ပြီး record လုပ်ပေး နိုင် အောင် လုပ်ပါမယ်။ ဖမ်းယူ ရရှိတဲ့ ပုံရှိပ် တွေကို analysis လုပ်နေပြီး လူ ကိုယ်ခန္ဓာ တွေကို တွေ့တာ နဲ့ ပုံမှာ အစိမ်းရောင် စတုဂံ နဲ့ ဘောင်ခတ် ပြပြီး၊ ပုံမှန် speed နဲ့ ပြောင်းပြီး record

လုပ်ပါမယ်။ Passive infrared sensor (PIR sensor) ကိုပါ သုံးပြီး sensor က လူရှိပ် ကို အာရုံခံ မိရင်လည်း ပုံရှိပါ မှာ အနီရောင် indicator ပြုပြီး၊ ပုံမှန် နှုန်းနဲ့ ပြောင်း record လုပ်နိုင် ပါမယ်။

ဒီ နမူနာ ပရိုဂရစ် မှာ လုပ်လို ရတဲ့ အကြောင်း ကုဒ် အကြမ်း ရေးပြ ရုံ ဖြစ်ပြီး၊ တစ်ခုခု ထူးခြားရင် ကိုယ့်ရဲ့ အီးမေးလ် ကို လှမ်းပို တာ၊ အီးက လူတွေ ရဲ့ မျက်နှာကို မှတ်မိတာ၊ သတ်မှတ် ထားတဲ့ အချိန် အပိုင်း အခြား အလိုက် လိုသလို record လုပ်တာ စတာ တွေကို စိတ်ကူး ရှုံရင် ရှိသလို ထပ်ဖြည့် နိုင် ပါတယ်။

ဈေးပေါ်တဲ့ [HC-SR501 PIR motion sensor](#) လေးကို [Aliexpress](#) စတဲ့ online shop တွေ ကနေ အလွယ် တကူ မှာ သုံးနိုင် ပါတယ်။ သူက 3.3 V ရော့၊ 5 V နဲ့ ပါ သုံးလို ရပြီး၊ RPi နဲ့ သုံးမှာ ဖြစ်လို 3.3 V ပါဝါပေး၊ sensor ရဲ့ output pin ကို RPi ရဲ့ GPIO23 နဲ့ ပုံ ၅.၇ မှာ ပြထား သလို ဆက်နိုင် ပါတယ်။



ပုံ ၅.၇: PIR sensor ကို ချိတ်ဆက်ခြင်း။

ပုံရှိပါ ထဲမှာ Body detection လုပ်ဖို့ အတွက် တော့ အပိုင်း ၅.၅ မှာ face detection လုပ်သလို ပဲ Haar feature-based cascade classifiers ကို သုံးမှာ ဖြစ်လို အဲဒီ မှာ haarcascade\_fullbody.xml ဆိုတဲ့ ဖိုင် နာမည် ပြောင်း လိုက်ရှုပါပဲ။ PIR sensor ဆက်ထား တဲ့ input ကို ဖတ်ပြီး၊ active ဖြစ်နေရင်

PIR DETECTED ဆိုတဲ့ စာကို puttext ဖန်ရှင် သုံးပြီး အနီရောင် နဲ့ ဖော်ပြ ပါမယ်။ နမူနာ surveillance.cpp ဆိုတဲ့ ပရိဂရမ် လေးကို စာရင်း ၅၁၆ မှာ ပြထား ပါတယ်။

```

1 //File: SmartCam.cpp
2 //Description: Smart surveillance camera using body detection and PIR sensor
3 //WebSite: http://cool-emerald.blogspot.sg
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye

6
7
8 #include <stdio.h>
9 #include <opencv2/opencv.hpp>
10 #include <string>
11 #include <time.h>
12 #include "ce_io.h"

13
14 #define CE_RASPICAM -1
15 #define CE_WEBCAM 0

16
17 //Choose the camera type
18 #define CE_CAMERA_TYPE CE_RASPICAM

19
20 #if CE_CAMERA_TYPE==CE_RASPICAM
21     #include <raspicam/raspicam_cv.h>
22 #endif // CE_CAMERA_TYPE

23
24 using namespace std;
25 using namespace cv;

26
27 #define CE_DAILY 3
28 #define CE_HOURLY 2
29 #define CE_MINUTELY 1
30 #define CE_MAKEVIDEO CE_MINUTELY
31 #define CE_PERIOD_NORMAL 100
32 #define CE_PERIOD_FAST 500
33

```

```

34 const string wpath=". /";
35 string body_cascade_name = wpath+"haarcascade_fullbody.xml";
36 //string body_cascade_name = wpath+"haarcascade_frontalface_alt.xml";
37 CascadeClassifier body_cascade;
38
39 int detectAndDisplay(Mat& frame)
40 {
41     std::vector<Rect> bodies;
42     Mat frame_gray;
43     int r=0;
44
45     cvtColor(frame, frame_gray, COLOR_BGR2GRAY);
46     equalizeHist(frame_gray, frame_gray);
47
48     body_cascade.detectMultiScale(frame_gray, bodies, 1.1, 2, 0 |
49         CV_HAAR_SCALE_IMAGE, Size(50, 100));// Detect bodies
50
51     for (size_t i = 0; i < bodies.size(); i++)
52     {
53         Point p1(bodies[i].x, bodies[i].y);
54         Point p2(bodies[i].x + bodies[i].width, bodies[i].y + bodies[i].
55             height);
56         rectangle(frame, p1, p2, Scalar(0, 255, 0), 4, 8, 0);
57         r=1;
58     }
59     return r;
60 }
61 string GetTimeStr()
62 {
63     time_t t = time(0); //now
64     struct tm * ts=localtime(&t);
65     string str="";
66     string tstr="";
67     tstr=to_string(ts->tm_year+1900);
68     str+=string(4-tstr.length(), '0')+tstr+"-";

```

```

69     tstr=to_string(ts->tm_mon+1);
70     str+=string(2-tstr.length(),'0')+tstr+"-";
71     tstr=to_string(ts->tm_mday);
72     str+=string(2-tstr.length(),'0')+tstr;
73     #if (CE_MAKIDEO < CE_DAILY)
74         str+=" ";
75         tstr=to_string(ts->tm_hour);
76         str+=string(2-tstr.length(),'0')+tstr;
77         #if (CE_MAKIDEO < CE_HOURLY)
78             str+=":";
79             tstr=to_string(ts->tm_min);
80             //str+=string(2-tstr.length(),'0')+tstr+":";;
81             //tstr=to_string(ts->tm_sec);
82             str+=string(2-tstr.length(),'0')+tstr;
83         #endif // MAKEVIDEO
84     #endif // MAKEVIDEO
85     return str;
86 }
87
88 int main(int argc, char** argv)
89 {
90     printf("SmartCam ... \n");
91     printf("Select the image frame and press ESC to exit.\n");
92
93     //init IO
94     CE_IO pir_sensor(23, INPUT);
95
96     //Load the cascades
97     if (!body_cascade.load(body_cascade_name)) { printf("Error in loading
98         haarcascade.\n"); return -1; };
99
100 #if CE_CAMERA_TYPE==CE_RASPICAM
101     raspicam::RaspiCam_Cv cap;
102     if (!cap.open())
103 #else
104         VideoCapture cap(0); //Default camera

```

```

104     if (!cap.isOpened())
105 #endif // CE_CAMERA_TYPE
106     { printf("Video is not opened. \n"); return -1; }
107 else
108 { printf("Video is opened. \n"); }

109

110

111     Size S = Size((int)cap.get(CAP_PROP_FRAME_WIDTH), (int)cap.get(
112         CAP_PROP_FRAME_HEIGHT));
113     printf("Frame size: %d x %d \n", S.width, S.height);

114     //If you want to change the camera resolution
115 //S = Size(640,480);
116     //cap.set(CAP_PROP_FRAME_WIDTH, S.width);
117     //cap.set(CAP_PROP_FRAME_HEIGHT, S.height);
118     //cap.release();
119     //cap.open(); //need to reopen with the new size

120

121     double rate = cap.get(CAP_PROP_FPS); //Frame rate
122     printf("Frame rate: %f \n", rate);

123

124     //int ex = VideoWriter::fourcc('X', '2', '6', '4');//small file size
125     int ex = VideoWriter::fourcc('M', 'J', 'P', 'G');
126     rate = 10; //changed frame rate to 10 fps
127     int dperiod = 100; //period to wait
128     string fname=GetTimeStr()+" .avi";
129     string previousfname=fname;
130     VideoWriter *outputVideo;
131     outputVideo=new VideoWriter(wpath+fname, ex , rate, S, true);
132     if(!outputVideo->isOpened())
133     {
134         cout << "Could not open video writer" << endl;
135         waitKey(5000);
136         return -1;
137     }
138     Mat frame;

```

```
139 int dFlag=0;
140 int pFlag=0;
141
142 for (int i = 0;; i++) {
143 #if CE_CAMERA_TYPE==CE_RASPICAM
144     cap.grab();
145     cap.retrieve(frame);
146 #else
147     if (!cap.read(frame)) break;
148 #endif // CE_CAMERA_TYPE
149     dFlag=detectAndDisplay(frame);
150     pFlag=pir_sensor.Read();
151
152     if(pFlag){
153         putText(frame,"PIR DETECTED", Point(40,40), FONT_HERSHEY_PLAIN
154         ,2.0, CV_RGB(255, 0, 0),2.0);
155     }
156
157     fname=GetTimeStr()+" .avi";
158     if(previousfname!=fname){
159         previousfname=fname;
160         outputVideo->release();
161         outputVideo=new VideoWriter(wpath+fname, ex , rate, S, true);
162         if(!outputVideo->isOpened())
163             {
164                 cout << "Could not open video writer."<< endl;
165                 waitKey(5000);
166                 return -1;
167             }
168     else{
169         *outputVideo << frame;
170     }
171
172     imshow("Frame", frame);
173     if(dFlag || pFlag){
```

```

174         dperiod=CE_PERIOD_NORMAL ;
175     }
176     else
177     {
178         dperiod=CE_PERIOD_FAST ;
179     }
180     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
181 }
182 cap.release();
183 outputVideo->release();
184 //waitKey(5000);
185 return 0;
186 }
```

စာရင်း ၅.၁၆: Smart Surveillance Camera

ဗိုဒ္ဓယို တွေ့ကို သိမ်းတဲ့ အခါ X264 ကို သုံးရင် ဖိုင် အချုပ်အစား တော်တော် သေးငယ်တဲ့ ဗိုဒ္ဓယို ဖိုင်တွေကို ရနိုင် ပါတယ်။ ဒါ ပေမယ့် CPU usage က တက်လာပြီး နေးလေး နေတာ တွေ့နိုင် ပါတယ်။ MJPG နဲ့ ဆိုရင်တော့ ဖိုင် size ကြိုးပြီး တွက်ချက် လုပ်ဆောင်မှု ပိုပါး ပါတယ်။ အဲဒီ ပရိုဂရမ် မှာ USB WebCam ကို သုံးမလား၊ Raspberry Pi camera module ကို သုံးမလား ရွေးလို ရပါတယ်။ Raspberry Pi camera module ကို သုံးထား ရင် တော့ အပိုင်း ၅.၆ မှာ ပြောခဲ့ သလို သူ့အတွက် လိုင်ဘရီ တွေကို ထည့်ပြီး build လုပ်ဖို့ လိုပါလိမ့် မယ်။ အခန်း ၃ မှာ ဆွေးနွေး ခဲ့တဲ့ GPIO class ကို လည်း ပြန်သုံး ထားပါတယ်။ ပရိုဂရမ် မှာ c++11 standard လိုတဲ့ အတွက် build လုပ်တဲ့ အခါ အောက်က အတိုင်း bulid လုပ်ပြီး run နိုင် ပါတယ်။

```

$ g++ surveillance.cpp `pkg-config --cflags --libs opencv` -std=c++11 -o
    surveillance -I/usr/local/include/ -lraspicam -lraspicam_cv
$ gksudo ./surveillance
```

## အကိုးအကားများ

- [Eme17] Cool Emerald. OpenCV on Linux using g++, CMake, Qt, Code::Blocks. 2017. url: <http://coolemerald.blogspot.sg/2017/11/opencv-on-linux-using-g-cmake-qt.html>.
- [Lag14] Robert Laganiere. OpenCV Computer Vision Application Programming Cookbook. 2nd. Packt Publishing, 2014. isbn: 1782161481, 9781782161486.
- [Ope16] OpenCV. VideoCapture Class Reference. 2016. url: [http://docs.opencv.org/3.2.0/d8/dfe/classcv\\_1\\_1VideoCapture.html](http://docs.opencv.org/3.2.0/d8/dfe/classcv_1_1VideoCapture.html).
- [Ope17a] OpenCV. Cascade Classifier. 2017. url: [http://docs.opencv.org/2.4/doc/tutorials/objdetect/cascade\\_classifier/cascade\\_classifier.html](http://docs.opencv.org/2.4/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html).
- [Ope17b] OpenCV. Installation in Linux. 2017. url: [http://docs.opencv.org/2.4/doc/tutorials/introduction/linux\\_install/linux\\_install.html](http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html).
- [Ope17c] OpenCV. Introduction to OpenCV. 2017. url: <http://docs.opencv.org/3.2.0/d1/dfb/intro.html>.
- [Ope17d] OpenCV. Open source computer vision. 2017. url: <http://opencv.org>.
- [Ope17e] OpenCV. Using OpenCV with gcc and CMake. 2017. url: [http://docs.opencv.org/2.4/doc/tutorials/introduction/linux\\_gcc\\_cmake/linux\\_gcc\\_cmake.html](http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_gcc_cmake/linux_gcc_cmake.html).
- [Ope17f] OpenCV. VideoWriter Class Reference. 2017. url: [http://docs.opencv.org/trunk/dd/d9e/classcv\\_1\\_1VideoWriter.html](http://docs.opencv.org/trunk/dd/d9e/classcv_1_1VideoWriter.html).
- [Sal17] Rafael Munoz Salinas. RaspiCam: C++ API for using Raspberry camera with/without OpenCv. 2017. url: <https://www.uco.es/investiga/grupos/ava/node/40>.

## အခန်း ၆

# GUI application များရေးသားခြင်း

wxWidgets က Windows । Linux နဲ့ Mac OSX အစ ရှိတဲ့ ပလက်ဖောင်း အမျိုးမျိုး ပေါ်မှာ GUI applications တွေ ရေးဖို့ အတွက် C++ library ပါ။ သူနဲ့ GUI code တွေ ရေးပြီး ရင် ပလက်ဖောင်း အမျိုးမျိုး ပေါ်မှာ ကုစ် ကို သိပ်ပြင် စရာ မလိုပဲ တန်းပြီး compile လုပ်၊ run လုပ်လို ရပါတယ်။

wxWidgets က free လည်းပေး open source လည်း ဖြစ် တဲ့ software ပါ။ ကိုယ်ပိုင် ဆော့တဲ့ တွေ ထုတ်မယ် ဆိုရင် လည်း ဘာမှ ကန်သတ်ချက် တွေ မရှိ ပါဘူး [wxW98]။ အဲဒါက Qt နဲ့ အမိက ကွာခြားချက် ပါ။ Qt က LGPLv3 လိုင်စင် ကို free ပေးထားပြီး၊ ကိုယ်ပိုင် စီးပွားရေး အတွက် သုံးမယ် ဆိုရင် ကန်သတ်ချက် တရီး၊ ရှိတာကြောင့် လိုင်စင် ဝယ်ဖို့ လိုကောင်း လိုနိုင် ပါတယ် [Qt17]။

Native platform ကို တတ်နိုင် သလောက် သုံးထား တာမို့ wxWidgets သုံးပို့ ရလာတဲ့ GUI တွေဟာ သုံးတဲ့ platform နဲ့ လိုက်ဖက်ပြီး ပင်ကို အမြင် အတိုင်း တသားထဲ ကျ တာကို ခံစား ရမှာပါ [wxW12]။ Standard C++ ကိုပဲ သုံးထားပြီး Qt တို့လို အထူး extension တွေ မသုံးထား တဲ့ အတွက် ရှုပ်ထွေးမှု နည်းတာ ကလည်း ကောင်းတဲ့ အချက် တစ်ခုပါ။

wxWidgets နဲ့ ရလာတဲ့ binary application တွေဟာ သေးယော ပေါ့ပါး တာမို့ embedded system တွေအတွက် အထူး သင့်တော် ပါတယ်။ နောက်တစ်ခါ library အရွယ်အစား တွေ ယုဉ်ရင်လည်း ဥပမာ အနေနဲ့ Qt library ကို တပ်ဆင်ရင်  $\approx$  200 MB လောက် အရွယ် ရှိပေမယ့် wxWidgets library က  $\approx$  30 MB လောက်ပဲ ယူပါတယ်။

wxWidgets က C++ အတွက် သာ မကဲ့ python, perl, php, java, lua, lisp, erlang, eiffel, C# (.NET), BASIC, ruby နဲ့ javascript အတွက် တောင်မှ bindings [wxW15] တွေ ရှိပါတယ်။ wxWidgets က တော်တော် ပြည့်စုံ ရန့်ကျက် တဲ့ GUI toolkits ဖြစ်ပြီး၊ utility classes လည်း အများကြီး ရှိတာမို့ ကောင်းမွန် သင့်တော် တဲ့ GUI toolkits အနေနဲ့ ညွှန်းဆို ချင်ပါတယ်။

wxWidgets ကို အသုံးပြုတဲ့ သူတွေ၊ အဖွဲ့အစည်းတွေ အများကြီး ရှိပြီး အဲဒီ အထဲမှာ လူသိများ တာတွေက NASA, AMD, Xerox, နဲ့ Open Source Applications Foundation (OSAF) တို့ဖြစ်ပါတယ်။ ထင်ရှားတဲ့ wxWidgets applications တွေက AVG AntiVirus, Audacity, Filezilla, Code::Blocks, CodeLite တို့ဖြစ်ပါတယ်။

## ၆.၁ wxWidgets ကိုတပ်ဆင်ခြင်း

wxWidgets ကို Raspberry Pi မှာ ထည့်ဖို့အတွက် အောက်က စာရင်း ၆.၁ အတိုင်း terminal မှာ command တွေရှိကြပြီး တပ်ဆင်ထည့်သွင်းနိုင် ပါတယ် [wxW14]။

```
1 $ sudo apt install build-essential
2 $ sudo apt install libwxgtk3.0-dev
```

စာရင်း ၆.၁: Linux ဘွင် wxWidgets ကို တပ်ဆင်ခြင်း။

နမူနာ အနေနဲ့ ရိုးရှင်း တဲ့ wxsimple.cpp ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၆.၂ အတိုင်း ဖန်တီး လိုက် ပါမယ် [Zet13]။

```
1 #include <wx/wx.h>
2 class Simple : public wxFrame
3 {
4 public:
5     Simple(const wxString& title);
6
7 };
8 Simple::Simple(const wxString& title)
9     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(250, 150))
10 {
11     Centre();
12 }
13
14 class MyApp : public wxApp
15 {
16 public:
17     virtual bool OnInit();
```

```

18 };
19
20 IMPLEMENT_APP(MyApp)
21 bool MyApp::OnInit()
22 {
23     Simple *simple = new Simple(wxT("Simple"));
24     simple->Show(true);
25
26     return true;
27 }

```

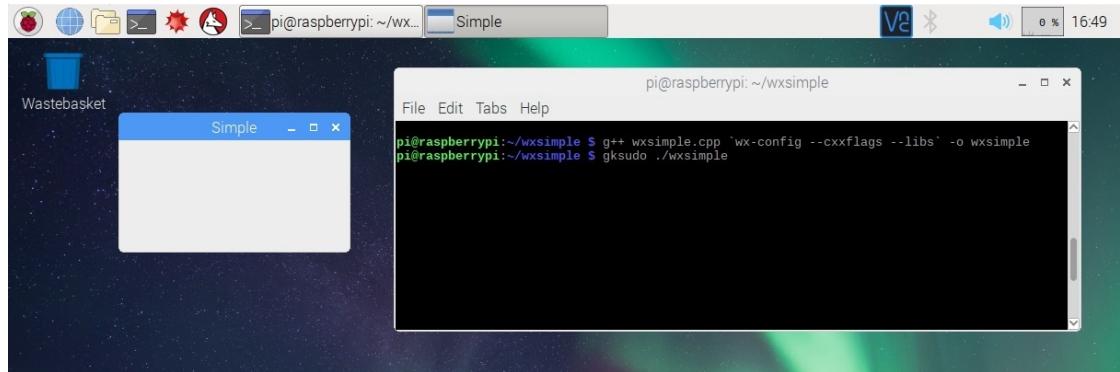
စာရင်း ၆.၂: wxsimple.cpp

ပြီးတဲ့ အခါ အောက်က command ကို သုံးပြီး compile လုပ်နိုင် ပါတယ်။ `wx-config --cxxflags` က compile လုပ်ဖို့ အတွက် လိုအပ်တဲ့ flags တွေကို ထူတ်ပေး ပြီး `wx-config --libs` က link လုပ်ဖို့ အတွက် လိုအပ်တဲ့ flags တွေကို ထူတ်ပေး ပါတယ်။

```
$ g++ wxmlite.cpp `wx-config --cxxflags --libs` -o wxmlite
```

GUI application ဖြစ်တဲ့ အတွက် VNC ကို သုံး ဒါမှ မဟုတ် monitor နဲ့ ဆက် ပြီး run ဖို့လို ပါတယ်။ ပြီးရင် terminal မှာ ခုန်က ရလာတဲ့ binary ကို အောက်က command နဲ့ run လိုက်တဲ့ အခါ ပုံ ၆.၃ အတိုင်း တွေ့နိုင် ပါတယ်။

```
$ gksudo ./wxmlite
```



ပုံ ၆.၁: wxWidgets အတွက် wxsimple.cpp နှမူနာ။

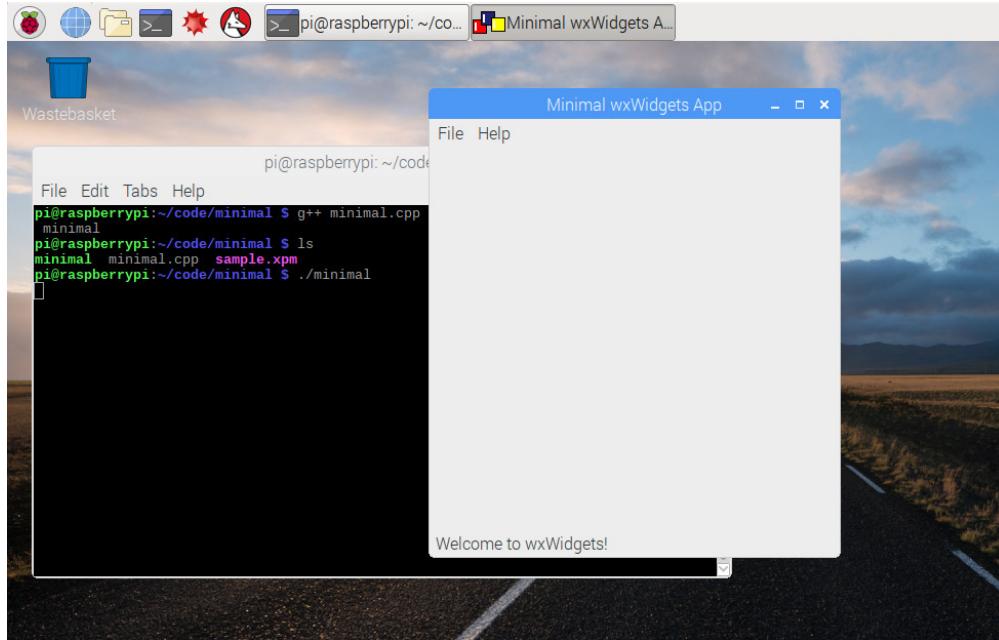
## ၆.J wxWidgets ကို source ဖဲ့ build လုပ်ခြင်း

wxWidgets ကို source ကနေ စိတ်ကြိုက် build လုပ်ချင် ရင်ပဲ ဖြစ်ဖြစ်၊ အရင် wheezy ဗာရ္ဇား မှာပဲ ဖြစ်ဖြစ် အောက်က အတိုင်း build လုပ်ပြီး install လုပ်နိုင် ပါတယ်။

```
$ sudo apt install build-essential
$ sudo apt install libgtk-3-dev
$ wget https://github.com/wxWidgets/wxWidgets/releases/download/v3.0.4/
      wxWidgets-3.0.4.tar.bz2
$ tar xjvf wxWidgets-3.0.4.tar.bz2
$ cd wxWidgets-3.0.4
$ mkdir gtk-build
$ cd gtk-build
$ ../configure --enable-unicode --disable-shared --with-gtk=3
$ make
$ sudo make install
$ wx-config --version
```

Build လုပ်ပြီးတဲ့ wxWidgets ကို သုံးပြီး samples အခန်း ထဲက minimal.cpp (Appendix A စာရင်း ၁.၁) ဆိုတဲ့ နှမူနာကို အောက်က command တွေနဲ့ run ကြည့် လိုက်တဲ့ အခါ ပုံ ၆.J အတိုင်း တွေ့ရ ပါတယ်။ အဲလို မဟုတ်ရင် ခုနဲ့ က wxsimple.cpp ကို build လုပ်တဲ့ အတိုင်း လုပ်ရင် လည်း ရပါတယ်။

```
$ cd gtk-build/samples/minimal
$ make
$ ./minimal
```



ပုံ ၆.၂: wxWidgets ဖြင့် minimal နမူနာကို run ခြင်း။

## ၆.၃ OpenCV နှင့် wxWidgets ကိုတဲ့သုံးခြင်း

OpenCV နဲ့ wxWidgets တဲ့ သုံးတဲ့ အကြောင်း ဆွေးနွေး ပါမယ်။ Linux နဲ့ terminal ပေါ်မှာ command ရိုက်ထည့် ပြီး build လုပ်တာ လွယ်ကူ ရိုးရှင်း ပါတယ်။ နမူနာ အနေနဲ့ `wxcvssimple.cpp` ဆိုတဲ့ ရိုးရှင်း တဲ့ ပရိုဂရမ် လေး တစ်ခု ရေးကြည့် ပါမယ်။ သူကို စာရင်း ၆.၃ မှာ ဖော်ပြထား ပါတယ်။

```

1 //File: wxcvssimple.cpp
2 //Description: A simple example to use OpenCV with wxWidgets
3 //Author: Yan Naing Aye
4 //Date: 2017 November 01
5 //MIT License - Copyright (c) 2017 Yan Naing Aye
6
```

```

7 #include <wx/wx.h>
8 #include <opencv2/opencv.hpp>
9 #include "util.h"
10 using namespace cv;
11
12 class MyFrame : public wxFrame
13 {
14     wxStaticBitmap *lena;
15     wxStaticBitmap *thiri;
16 public:
17     MyFrame(const wxString& title);
18
19 };
20 MyFrame::MyFrame(const wxString& title)
21     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(512, 256))
22 {
23     Centre();
24     lena = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("lena.jpg"),
25         wxBITMAP_TYPE_JPEG), wxPoint(0,0), wxSize(256, 256));
26     thiri = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("lena.jpg"),
27         wxBITMAP_TYPE_JPEG), wxPoint(256,0), wxSize(256, 256));
28
29     //From opencv to wx
30     Mat imcv1=imread("thiri.jpg",CV_LOAD_IMAGE_COLOR);
31     wxBitmap imwx1=wx_from_mat(imcv1);
32     thiri->SetBitmap(imwx1);
33
34     //From wx to opencv
35     wxImage imwx2;
36     imwx2.LoadFile(wxT("lena.jpg"), wxBITMAP_TYPE_JPEG);
37     Mat imcv2=mat_from_wx(imwx2);
38     namedWindow("Img", CV_WINDOW_AUTOSIZE);
39     imshow("Img", imcv2);
40 }
41
42 class MyApp : public wxApp

```

```

41 {
42     public:
43         virtual bool OnInit();
44 };
45
46 IMPLEMENT_APP(MyApp)
47 bool MyApp::OnInit()
48 {
49     if ( !wxApp::OnInit() )
50         return false;
51     wxInitAllImageHandlers();
52     MyFrame *frame = new MyFrame(wxT("Simple wxWidgets + OpenCV"));
53     frame->Show(true);
54
55     return true;
56 }
```

စာရင်း ၆.၃: wxcvssimple.cpp

ပရိုဂရမ် အစမှာ Application ရဲ့ OnInit() ဆိုတဲ့ method ထဲမှာ

```
wxInitAllImageHandlers();
```

ဆိုတာကို ထည့်ပါမယ်။ အဲဒီနောက် MyFrame ဆိုတဲ့ wxFrame ရဲ့ derived class ထဲမှာ ပုံရှိပို့ တွေကို ဖော်ပြန့် wxStaticBitmap variable တွေကို ကြော်လိုက် ပါမယ်။ MyFrame ရဲ့ constructor မှာ wxStaticBitmap တွေကို ဖန်တီးဖို့ အောက်က ကုဒ် ကို သုံးနိုင် ပါတယ်။

```
lena = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("lena.jpg"),
    wxBITMAP_TYPE_JPEG), wxPoint(0,0), wxSize(256, 256));
```

Image တွေ အတွက် OpenCV မှာသုံးတဲ့ Mat နဲ့ wxWidgets ရဲ့ wxImage ကို အပြန် အလှန် ပြောင်းပေးတဲ့ mat\_from\_wx | wx\_from\_mat အစ ရှိတဲ့ ဖန်ရှင် တွေကို စာရင်း ၆.၄ မှာ ဖော်ပြ ထားတဲ့ util.h ထဲမှာ တွေ့နိုင် ပါတယ် [Dad10]။

```
1 //File: util.h
```

```

2 //Description: Functions to convert wxImage and OpenCV Mat
3 //Author: Yan Naing Aye
4 //MIT License - Copyright (c) 2017 Yan Naing Aye
5
6 #include <wx/wx.h>
7 #include <opencv2/opencv.hpp>
8 using namespace cv;
9
10 wxImage wx_from_mat(Mat &img) {
11     Mat im2;
12     if(img.channels() == 1){cvtColor(img, im2, COLOR_GRAY2RGB);}
13     else if (img.channels() == 4) { cvtColor(img, im2, COLOR_BGRA2RGB);}
14     else {cvtColor(img, im2, COLOR_BGR2RGB);}
15     long imsize = im2.rows*im2.cols*im2.channels();
16     wxImage wx(im2.cols, im2.rows, (unsigned char*)malloc(imsize), false);
17     unsigned char* s=im2.data;
18     unsigned char* d=wx.GetData();
19     for (long i = 0; i < imsize; i++) { d[i] = s[i];}
20     return wx;
21 }
22
23 Mat mat_from_wx(wxImage &wx) {
24     Mat im2(Size(wx.GetWidth(),wx.GetHeight()), CV_8UC3,wx.GetData());
25     cvtColor(im2,im2,COLOR_RGB2BGR);
26     return im2;
27 }
```

စာရင်း ၆.၄: util.h

ဆက်ပြီး MyFrame ရဲ့ constructor ထဲမှာ သီရိ ရဲ့ ပုံကို OpenCV သုံးပြီး imread နဲ့ ဖတ်လိုက်ပါတယ်။ ဖတ်လို ရတဲ့ Mat အမျိုး အစား ပုံကို wx\_from\_mat ဖန်ရှင် ကို သုံးပြီး wxImage အမျိုး အစား ပြောင်းလိုက် ပါတယ်။ ပြီးတော့ wxStaticBitmap ရဲ့ SetBitmap method သုံးပြီး Frame ပေါ်မှာ ဖော်ပြ လိုက် ပါတယ်။

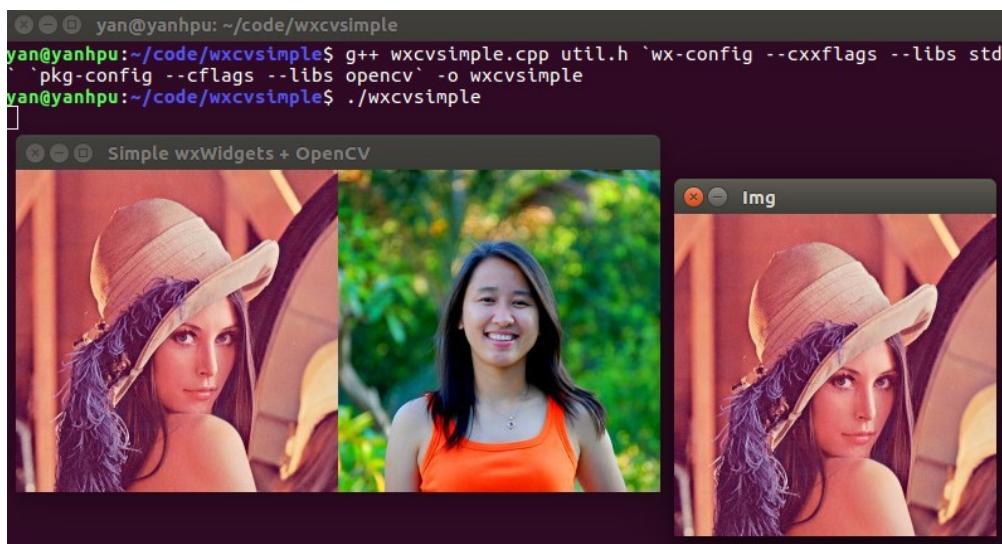
နောက်တစ်ခါ wxImage အမျိုး အစား variable တစ်ခု ကြော်ပြီး LoadFile method နဲ့ လိုနာ

ရဲ့ ပုံကို wxWidget သုံးပြီး ဖတ်လိုက် ပါတယ်။ ရလာ တဲ့ wxImage အမျိုး အစား ပုံကို mat\_from\_wx ဆိုတဲ့ ဖန်ရှင် သုံးပြီး OpenCV အတွက် Mat အမျိုး အစား ပြောင်းလိုက် ပါတယ်။ ရလာတဲ့ ပုံရှုပို ဖြစ်ပါတယ်။

ပရီ ကရမဲ ကို build လုပ်ဖို့ အတွက် terminal မှာ အောက်က command ကို သုံးနိုင် ပါတယ်။

```
$ g++ wxcvssimple.cpp util.h `wx-config --cxxflags --libs std` `pkg-config --cflags --libs opencv` -o wxcvssimple
```

Terminal မှာ ./wxcvssimple ကို ရိုက်ပြီး ပရီကရမဲ ကို run လိုက်တဲ့ အခါ ရလာတဲ့ ရလဒ် ကို ပုံ ၆.၃ မှာ တွေ့နိုင် ပါတယ်။

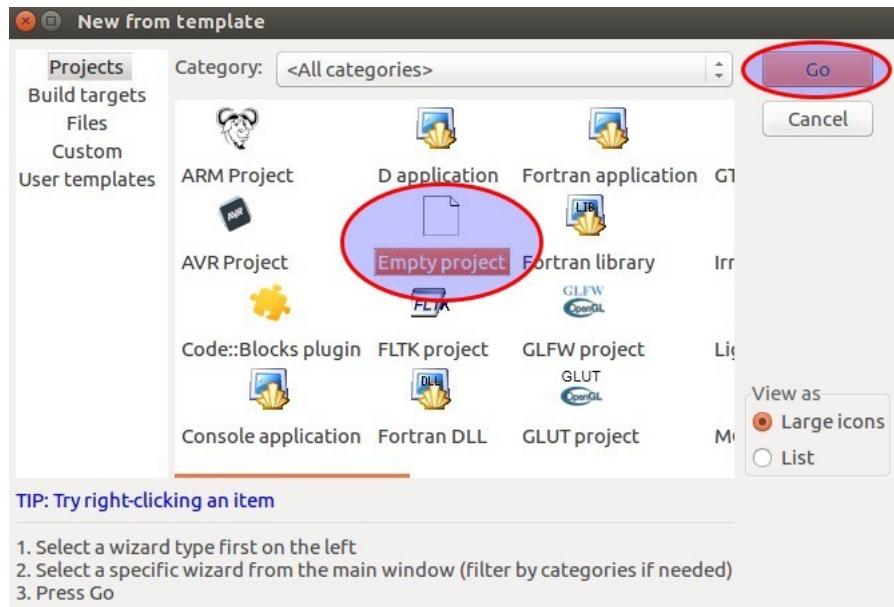


ပုံ ၆.၃: wxWidgets နှင့် OpenCV ကို တွဲစပ် အသုံးပြု ထားသည့် ရှိုးရှင်းသော နမူနာ။

### ၆.၃.၁ Code::Blocks

Application တစ်ခု ကို Raspberry Pi ပေါ်မှာ တိုက်ရှိက် develop လုပ်ရ တာ ထက် စာရင် အပိုင်း J.၉ မှာ ပြောခဲ့သလို desktop ကွန်ပျူးတာ ပေါ်မှာ ပရီကရမဲ ကို အရင် ရေးပြီး မှ Raspberry Pi ပေါ် ပြောင်းရ တာ ပိုပြီး လွယ်ကူ အဆင်ပြေ ပါတယ်။ Code::Blocks မှာ wxWidgets ကို OpenCV နဲ့ တွဲသုံးဖို့အတွက် wxWidgets project အသစ် တစ်ခု ကို ပုံ ၆.၄ မှာ ဖော်ပြု ထားတဲ့ အတိုင်း empty project တစ်ခု ဖန်တီးပြီး wxcv.cpp ဆိုတဲ့ project နာမည် ပေးလိုက် ပါမယ်။ ပြီးတဲ့ အခါ GNU GCC Compiler ကို

ရွှေး ပါမယ်။



ပုံ ၆.၇: Code::Blocks တွင် Empty project စာစ်ခု ဖန်တီးခြင်း။

File Menu → New → Empty File ကို နိပ်ပြီး စာရင်း ၆.၅ မှာ ပြထားတဲ့ wxcv.cpp ဖိုင်ကို ဖန်တီးလိုက် ပါမယ်။

```

1
2 #include <wx/wx.h>
3 #include <opencv2/opencv.hpp>
4 using namespace cv;
5
6 class Simple : public wxFrame
7 {
8 public:
9     Simple(const wxString& title);
10
11 };
12 Simple::Simple(const wxString& title)
13     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(250, 150))
14 {
15     Centre();

```

## ၆.၃. OPENCV နှင့် WXWIDGETS ကိုတဲ့သံဃားခြင်း

၁၆၉

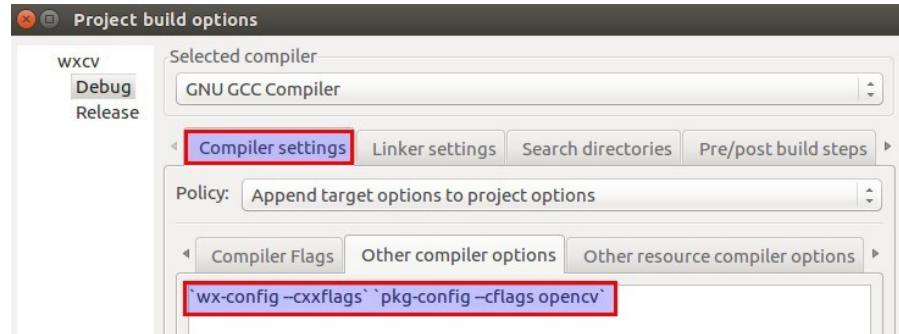
```
16 Mat img=imread("grove.jpg",CV_LOAD_IMAGE_COLOR);
17 namedWindow("Img",CV_WINDOW_AUTOSIZE);
18 imshow("Img",img);
19 }
20
21 class MyApp : public wxApp
22 {
23     public:
24         virtual bool OnInit();
25 };
26
27 IMPLEMENT_APP(MyApp)
28 bool MyApp::OnInit()
29 {
30     Simple *simple = new Simple(wxT("Simple"));
31     simple->Show(true);
32
33     return true;
34 }
```

စာရင်း ၆.၅: wxcv.cpp

ပြီးရင် Project Menu → Build Options... ကို နှုန်းပြီး Compiler settings tab → Other compiler options မှာ ပုံ ၆.၅ အတိုင်း

```
`wx-config --cxxflags` `pkg-config --cflags opencv`
```

ကို Debug အတွက်ရော၊ Release အတွက်ပါ သတ်မှတ် နိုင် ပါတယ်။

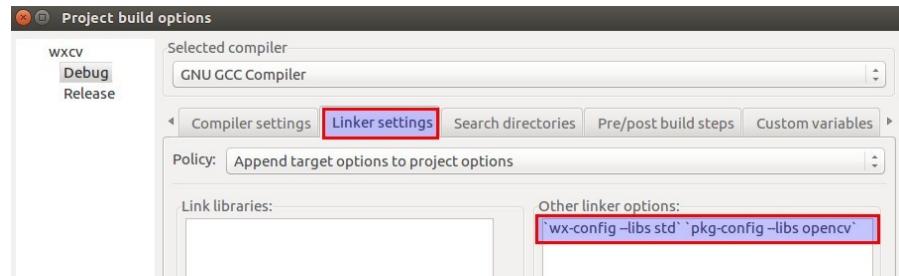


ပုံ ၆.၅: Compiler options များသတ်မှတ်ခြင်း။

နောက်တစ်ခါ Linker settings tab → Other linker options မှာ လည်း ပုံ ၆.၆ အတိုင်း

```
'wx-config --libs std` `pkg-config --libs opencv'
```

ကို ထပ် သတ်မှတ်ပါမယ်။

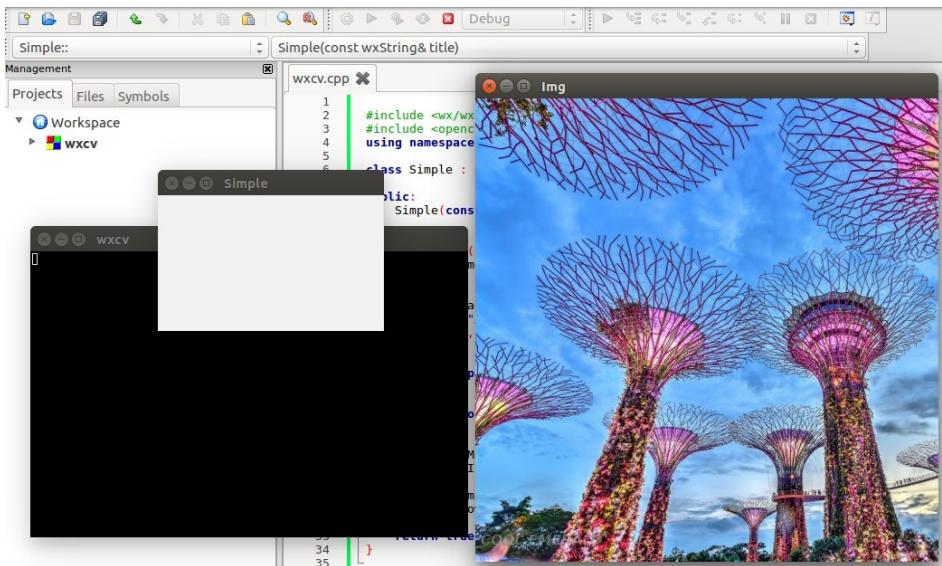


ပုံ ၆.၆: Linker options များသတ်မှတ်ခြင်း။

နောက်ဆုံးမှာ F9 ခလုပ်ကို နိုပ်ပြီး Build and run လုပ်လိုက် တဲ့အခါ ပုံ ၆.၇ မှာ ပြထား သလို ပရိုကရမဲ ရဲ output ကို ထွေးနိုင် ပါတယ်။

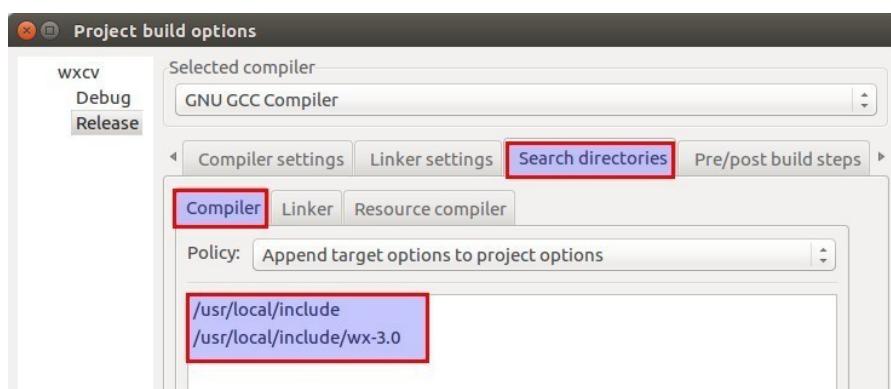
## ၆.၃. OPENCV နှင့် WXWIDGETS ကိုတဲ့သုံးခြင်း

၁၇၁



ပုံ ၆.၇: OpenCV နမူနာ ပရိုဂရမ်ကို Code::Blocks တွင် run ခြင်း။

ပရောဂျက် နဲ့ ပတ်သက် တဲ့ အချက်အလက် တွေကို IDE ကို ပိုမြီး သိစေခဲ့ရင် optional အဆင့်တွေ အနေဖြင့် Project Menu → Build Options... ကို နှိပ်ပြီး Search directories tab → Compiler tab မှာ ပုံ ၆.၈ အတိုင်း /usr/local/include နဲ့ /usr/local/include/wx-3.0 ကို Debug အတွက်ရော၊ Release အတွက်ပါ သတ်မှတ် နိုင် ပါတယ်။

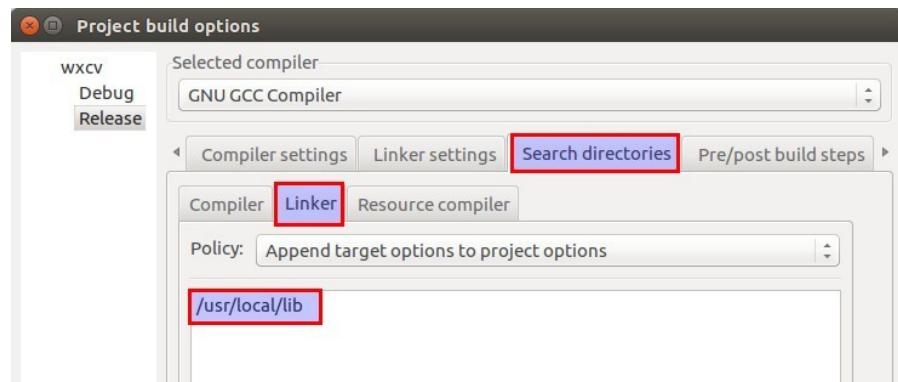


ပုံ ၆.၈: Compiler Search directories များသတ်မှတ်ခြင်း။

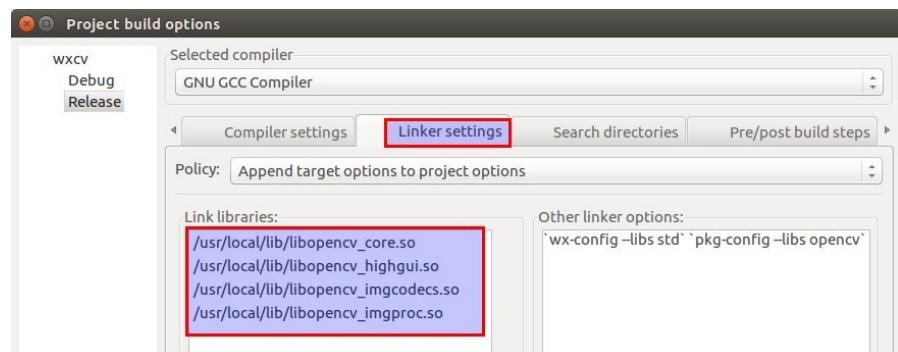
နောက်တစ်ခါ ဘေးဘက် က Linker tab မှာ ပုံ ၆.၉ အတိုင်း lib တွေရဲ့ လမ်းကြောင်း /usr/local/lib ကို သတ်မှတ် ပါမယ်။ Linker settings tab မှာလည်း ပုံ ၆.၁၀ အတိုင်း Link libraries တွေ ဖြစ်တဲ့

```
libopencv_core.so
libopencv_highgui.so
libopencv_imgcodecs.so
libopencv_imgproc.so
```

အစ ရှုတဲ့ library တွေကို လိုသလို သတ်မှတ် ဖိုင် ပါတယ်။



ပုံ ၆.၉: Library ဖိုင်များအတွက် Search directories များသတ်မှတ်ခြင်း။



ပုံ ၆.၁၀: Link libraries များသတ်မှတ်ခြင်း။

Search directories တွေနဲ့ပတ်သက် တဲ့ အချက်အလက် တွေကို အောက်က command တွေကို terminal မှာ ရိုက် ကြည့်ပြီး လည်း စစ်ဆေး သိရှိ ဖိုင် ပါတယ်။

```
$ wx-config --cxxflags
$ pkg-config --cflags opencv
```

```
$ wx-config --libs std
$ pkg-config --libs opencv
```

Transparency ပါတဲ့ channel 4 ခု ရှိတဲ့ ပုံစိုပ် တွေကို imread နဲ့ ဖတ်တဲ့ အခါ IMREAD\_UNCHANGED ကို သုံးနိုင် ပြီး၊ အဲဒီ အတွက် နမူနာ ကို Appendix A စာရင်း ၅.၂ မှာ ပြထားတဲ့ [wxcv-alpha.cpp](#) ဆိုတဲ့ ပရိုဂရမ် မှာ တွေ့နိုင် ပါတယ်။

## ၆.၅ Autostart

GUI application တွေ အတွက် စက် စပွင့် တာနဲ့ အလို အလျောက် စတင် အလုပ် လုပ် ဖို့ သတ်မှတ် ပေးနိုင် ပါတယ် [Ada16b]။ User pi အတွက် autostart ကို ပြုပြင်ဖို့ အောက်က command သုံးပြီး edit လုပ်နိုင် ပါတယ်။

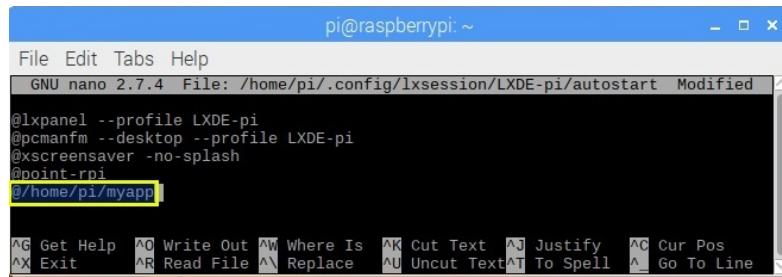
```
$ sudo nano /home/pi/.config/lxsession/LXDE-pi/autostart
```

အဲဒီ ထဲမှာ အလို အလျောက် စတင် ချင်တဲ့ ပရိုဂရမ် ကို ရှေ့မှာ @ သက်တဲ့ ခံပြီး ထည့်ပေး လိုက်ရုံ ပါပဲ။ Autostart အလုပ် လုပ် မလုပ် စမ်းကြည့် ဖို့ စက်တဲ့ မှ ပါပြီးသား lxterminal လို application ကို အလို အလျောက် ပွင့်လာ မလာ အောက်က စာကြောင်းကို ဖြည့်၊ reboot လုပ်ကြည့်ပြီး စမ်းသပ် ကြည့်နိုင် ပါတယ်။

@lxterminal

နောက် ဥပမာ အနေနဲ့ home အခန်း ထဲက myapp ဆိုတဲ့ ပရိုဂရမ် ကို အလို အလျောက် စချင် ရင် အဲဒီ autostart ဖိုင်ထဲ မှာ အောက်က စာကြောင်းကို ပုံ ၆.၁၁ အတိုင်း ဖြည့်နိုင် ပါတယ်။

```
@/home/pi/myapp
```



```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 2.7.4 File: /home/pi/.config/lxsession/LXDE-pi/autostart Modified
@lxpanel --profile LXDE-pi
@pcmanfm --desktop --profile LXDE-pi
@xscreensaver -no-splash
@point-rpi
@/home/pi/myapp

AG Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^L Go To Line
```

ပုံ ၆.၁၁: Autostart ဖိုင်ကို ပြင်ဆင်ခြင်း။

IO pin စတာ တွေသုံး ဖို့လို တဲ့ application တွေဆို root privileges တွေ လိုတဲ့ အတွက် ရှုမှာ sudo ခံဖို့လိုပါတယ်။ GUI application တွေ အတွက် ဆိုရင် sudo အစား gksudo ကို သုံးတာ ပိုကောင်းပါတယ်။

```
@gksudo /home/pi/myapp
```

တစ်ခါ တစ်လေ application ကို စက်ဖွင့် ဖွင့်ချင်း ချက်ချင်း မစ သေးပဲ delay ခေါ် ခံပြီး မှ စချင် တာမျိုး၊ ဒါမှ မဟုတ် working directory ကို ပြောင်းတာ စတဲ့ တွေား command တွေ နဲ့ ပါ တွဲ လုပ်ချင် ရင်တော့၊ အဲဒါ တွေကို script ဖိုင် တစ်ခု ထဲမှာ စု ရေးပြီး၊ autostart ကနေ အဲဒီ script ကို လုမ်း ခေါ်နိုင် ပါတယ်။ ဥပမာ myapp ကို delay ခံပြီး၊ root နဲ့ run ပေးတဲ့ start\_myapp.sh ဆိုတဲ့ script ကို စာရင်း ၆.၆ မှာ ပြထား သလိုမျိုး ရေးနိုင် ပါတယ်။

```
1 #!/bin/bash
2 sleep 5
3 gksudo ./myapp
```

စာရင်း ၆.၆: start\_myapp.sh

အဲဒီလို script ရေးပြီး တဲ့ အခါ သူကို executable ဖြစ်အောင် အောက်ပါ အတိုင်း ပြောင်းပေး နိုင် ပါတယ်။

```
$ sudo chmod +x /home/pi/start_myapp.sh
```

ပုံမှန် ဆိုရင် root privileges အတွက် sudo ကို သုံးတဲ့ အခါ password ကို ထည့်ပေးဖို့ လို ပါတယ်။ Raspberry Pi မှာတော့ user pi အတွက် password မတောင်း အောင် လုပ်ထား လေ့ ရှိ ပါတယ်။

တကယ်လို သင့် စက်မှာ က password ထည့်ဖို့ လိုမယ် ဆိုရင် gksudo သုံးထား တဲ့ application က အလို အလျောက် စနိုင်မှာ မဟုတ် ပါဘူး။ အဲဒါ ဆိုရင် sudoers မှာ ပြင်ဆင် သတ်မှတ် ပေးဖို့ အောက်က command ကို သုံးနိုင် ပါတယ်။

```
$ sudo visudo
```

အဲဒီ sudoers ဖိုင်မှာ password မလိုပဲ ဖွင့် စေခဲ့ပါ တဲ့ ပရိုဂရမ် ကို နောက်ဆုံး လိုင်းမှာ အောက်က အတိုင်း သတ်မှတ် ပေးနိုင် ပါတယ် (ပုံ ၆.၁၂)။

```
pi ALL=(ALL) NOPASSWD: /home/pi/myapp
```

```
pi ALL=(ALL) NOPASSWD: /home/pi/myapp
```

ပုံ ၆.၁၂: Application တစ်ခု ကို password မလိုအောင် sudoers တွင် ပြင်ဆင်ခြင်း။

## အကိုးအကားများ

- [Ada16b] Adam. Auto Running Programs-GUI. 2016. url: <http://www.raspberry-projects.com/pi/pi-operating-systems/raspbian/auto-running-programs-gui>.
- [Dad10] Jive Dadson. OpenCV Image and wxImage Conversion. 2010. url: <https://stackoverflow.com/a/2241517>.
- [Qt17] Qt. Licensing - Before you begin, make the right license choice. 2017. url: <https://www1.qt.io/licensing/>.

- [Zet13] ZetCode. wxWidgets tutorial. 2013. url:  
<http://zetcode.com/gui/wxwidgets/>.
- [wxW12] wxWiki. WxWidgets Compared To Other Toolkits. 2012. url: [https://wiki.wxwidgets.org/WxWidgets\\_Compared\\_To\\_Other\\_Toolkits](https://wiki.wxwidgets.org/WxWidgets_Compared_To_Other_Toolkits).
- [wxW14] wxWiki. Compiling and getting started. 2014. url: [https://wiki.wxwidgets.org/Compiling\\_and\\_getting\\_started](https://wiki.wxwidgets.org/Compiling_and_getting_started).
- [wxW15] wxWiki. Bindings. 2015. url: <https://wiki.wxwidgets.org/Bindings>.
- [wxW98] wxWidgets. wxWindows Library Licence. 1998. url: <https://www.wxwidgets.org/about/licence/>.

## အခန်း ၇

# Serial Port

Serial communication မှာ data byte တွေရဲ့ bit တွေကို အစီအစဉ်လိုက် တစ်ခုပြီး တစ်ခု ပိုတဲ့ အတွက် ဝါယာ တွေ အများကြီး သုံးစရာ မလို တော့ပဲ ဝါယာ တစ်ခု နဲ့တင် ပိုလို ရတဲ့ အားသာချက် ရှိပါ တယ်။ ပိုတဲ့ အခါ byte ကနေ bit တစ်ခု ချင်းစီ အနေ နဲ့ serial ပြောင်းပြီး ထွက်လာ ဖို့ လက်ခံတဲ့ အခါ တစ်ခု ပြီး တစ်ခု ဝင်လာ တဲ့ serial bit တွေကို byte အဖြစ် ပြန် တည်ဆောက် ဖို့ အတွက် UART (universal asynchronous receiver-transmitter) ကို သုံးကြ ပါတယ်။ UART တစ်ခု မှာ transmit (TX) လို ခေါ်တဲ့ အထွက် တစ်ခု နဲ့ receive (Rx) လို ခေါ်တဲ့ အဝင် တစ်ခု ပါရှု ပါတယ်။

## ၇.၁ UART

UART တစ်ခု ဟာ serial bit stream ပြောင်းပေး နိုင်ရုံ တင် မက ပဲ start bit, parity bit, stop bit တွေ ထည့်ပေးခြင်း စတဲ့ လုပ်ငန်း တွေကို လည်း Asynchronous Serial Communication Protocol နဲ့ ကိုက်ညီ အောင် ဖြည့်စွက် လုပ်ဆောင် ပေးပါ တယ်။

### ၇.၁.၁ Start Bit

UART တစ်ခု က ပိုစရာ data မရှိပဲ idle ဖြစ်နေ ရင် output ကို 1 မှာထား ပါတယ်။ အဲဒီ အခါ လက်ခံ မယ့် UART ဟာ idle ဖြစ်လို ထုတ်ထား တဲ့ 1 လား၊ ပိုပေး နေတဲ့ ဒေတာ က 1 လား ခွဲခြား သိဖို့ လိုလာ ပါတယ်။ အဲဒါကို ဖြောင်းဖို့ အတွက် 0 bit တစ်ခု ကို start bit အနေနဲ့ သုံး ပါတယ်။ လက်ခံ နေတဲ့ UART က 0 ရောက် မလာ မချင်း ရှိနေ တဲ့ 1 ကို idle လို ယူဆ ပါတယ်။ ပထမ ဆုံး 0 ရောက်လာ တာနဲ့ အဲဒါ ကို start bit လို ယူဆပြီး နောက်ဝင် လာမယ့် bit တစ်ခု က စပြီး data အနေနဲ့ လက်ခံ ပါတယ်။

### ၇.၁.၂ Baud Rate

Data bit တစ်ခု နဲ့ တစ်ခု ကြားက အချင် အကွာ အငေး ကို baud rate က သတ်မှတ် ပါတယ်။ ဥပမာ baud rate က 1 kHz ဆိုရင်၊ bit တစ်ခု အတွက် time period က  $1/(1 \text{ kHz}) = 1 \text{ ms}$  ကြာမှာ ဖြစ် ပါတယ်။ အသုံး များတဲ့ baud rate တွေကတော့ 9600 တို့၊ 115200 တို့ ဖြစ် ပါတယ်။

### ၇.၁.၃ Data Bits

Data ပိုတဲ့ အခါ ပိုတဲ့ UART နဲ့ လက်ခံ တဲ့ UART တို့ရဲ့ data bit အရေ အတွက် ခြင်း တူဖို့ လို ပါတယ်။ ပိုတဲ့ အခါ 5 bit data ကနေ 8 bit data ထိ ပိုတတ် ပါတယ်။ ဥပမာ ASCII data ဆိုရင် ပိုတာ 7 bit ပဲ ရှိတဲ့ အတွက် လက်ခံ ရင်လဲ 7 bit လက်ခံ ဖို့ လို ပါတယ်။ Data bits တွေကို ပိုတဲ့ အခါ LSB ကနေ အရင် စပို ပါတယ်။ အသုံး များတာ ကတော့ 8 bit ပါ။

### ၇.၁.၄ Parity Bit

လက်ခံ လိုက်တဲ့ data က မှန်ကန် ကိုက်ညီ မှု ရှိရဲ့ လား ပြန်စစ်ဖို့ အတွက် parity bit ကိုလဲ data bits တွေရဲ့ နောက်မှာ အပို ထည့်ပေး တတ် ပါတယ်။ 1 အရေ အတွက် စုံ ကဏ္ဍား ဖြစ်အောင် ထည့်ပေး ရင် Even parity ၊ မ ကဏ္ဍား ဖြစ်အောင် ပေါင်းထည့် ပေးရင် Odd parity ပါ။ ပိုတဲ့ UART နဲ့ လက်ခံ တဲ့ UART parity ခြင်းလည်း ကိုက်ညီ ဖို့ လို ပါတယ်။ များသော အားဖြင့် Parity ထည့် သုံးလေ့ မရှိ ပါဘူး။

### ၇.၁.၅ Stop Bit

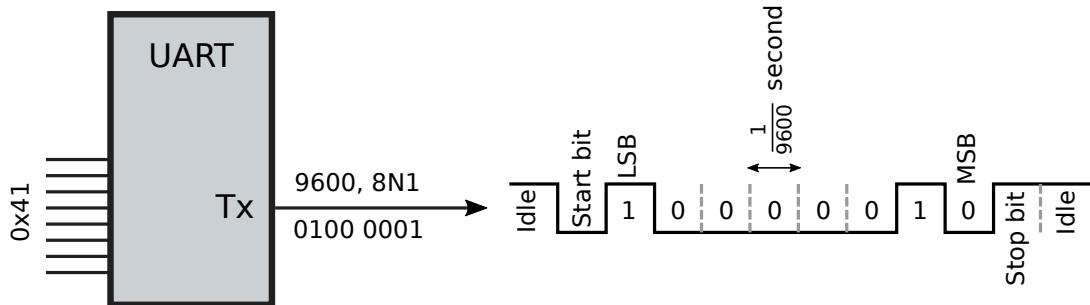
8 bits ရှိတဲ့ data byte တစ်ခု နဲ့ တစ်ခု ကြားမှာ ခြား ပေးဖို့ အတွက် stop bits တွေကို သုံး ပါတယ်။ 1 ကို stop bit အနေ နဲ့ သုံး ပါတယ်။ Stop bit ကို တစ်ခု၊ သို့မဟုတ် နှစ်ခု သုံး လို ရ ပါတယ်။ data တွေ parity bit တွေ ပို့ပြီး တဲ့ အခါ နောက်မှာ 1 တစ်ခု ပေါင်း ပေး၊ ဒါမှ မဟုတ် နှစ်ခု ပေါင်းထည့် ပေး တာပါ။ အသုံး များတာ တော့ stop bit တစ်ခု ထဲ ပါပဲ။

အောက်က ပုံ ၇.၁ မှာ နမူနာ အနေ နဲ့ data byte 0x41 ကို UART တစ်ခု ကနေ 9600, 8N1 နဲ့ ပိုတာ ကို ပြထား ပါတယ်။ ဆိုလို တာက

- Baud rate = 9600
- Data bit = 8 bits
- Parity = No parity (N= No parity, E= Even parity, O= Odd parity)

- Stop bit= 1 stop bit

လို ဆိုလို တာပါ။ No parity ဖြစ်တဲ့ အတွက် data ကို ပိုမြီး တဲ့ အခါ parity bit ကို မထည့် ပဲ stop bit တန်းလာ ပါတယ်။

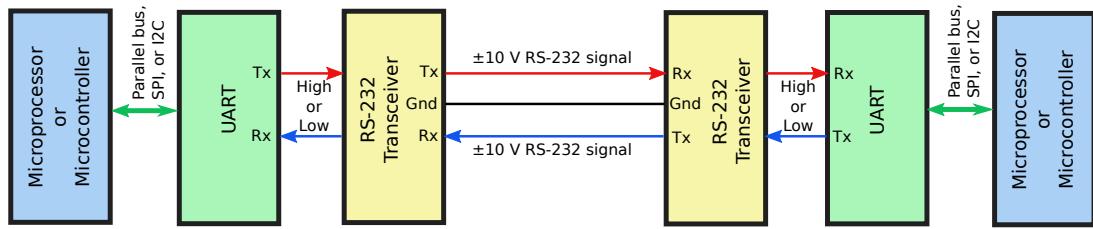


ပုံ ဂ.၁: UART တစ်ခု မှ data byte (0x41) ကို serial bit stream (9600, 8N1) အဖြစ် ပြောင်းပေးခြင်း။

## ၇.J RS-232

UART တစ်ခု က ပုံမှန် အားဖြင့် 1 အတွက် high level voltage ( 5 V, 3.3 V စတာ တွေ ) ထုတ် ပေးပြီး၊ 0 အတွက် ဆိုရင် low level votage ( 0 V ) ထုတ်ပေး ပါတယ်။ များသော အားဖြင့် UART တစ်ခု က ထွက် လာတဲ့ voltage signal ကို external device တွေနဲ့ ဆက်ဖို့ သင့်တော် တဲ့ signaling levels တွေ အဖြစ် ပြောင်းဖို့ transceiver တစ်ခုခဲ့ပြီး ထုတ်ပေး လေ့ ရှိ ပါတယ်။ အသုံး များတဲ့ standard တစ်ခု ကတော့ RS-232 (Recommended Standard 232) ပါ။ RS-232 transceiver တစ်ခုရဲ့ အလုပ်က 1 အတွက် -10 V နဲ့ 0 အတွက် +10 V အဖြစ် အပြန် အလုန် ပြောင်းပေး တာပါ။ ပုံမှန်က  $\pm 10$  V ဆိုပေမယ့် အပေါင်းအနှစ် 3 V ကနေ 25 V အထိက valid ဖြစ်ပါတယ်။ အသုံး များတဲ့ transceiver တွေ ကတော့ MAX232 တို့၊ MAX202 တို့ ဖြစ် ပါတယ်။

ပုံ မှာ RS-232 ဆက်သွယ် အသုံး ပြုပုံ နမူနာ တစ်ခု ကို ပြထား ပါတယ်။ ပုံမှန် အားဖြင့် transmit နဲ့ receive data လိုင်း တွေကို ပဲ သုံးလေ့ ရှိတဲ့ အတွက် Tx တစ်လိုင်း၊ Rx တစ်လိုင်း နဲ့ Ground ဝါယာ တစ်လိုင်း စုစုပေါင်း သုံးလိုင်း သုံးဖို့လိုပါတယ်။



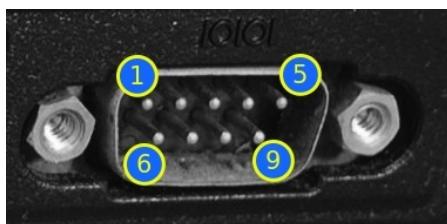
ပုံ ၇.၂: RS-232 ဆက်သွယ် အသုံးပြုခြင်း။

RS-232 ရဲ Tx နဲ Rx ကို တစ်ခါ တစ်လေ ခွဲခြား ဖို့လို လာမြီ ဆိုရင် ကြိုး တွေကို open လုပ်ပြီး မိတ္တ ကို သုံးပြီး အလွယ် တကူ တိုင်း ကြည့်နိုင် ပါတယ်။ Tx နဲ Ground ကြားမှာ ပုံမှန် အားဖြင့် -10V လောက် တွေ ရမှာ ဖြစ်ပြီး Rx နဲ Ground ကြားမှာ ပို မရှိ ပါဘူး။ RS232 ကို unbalanced lines လို ဆို ပါတယ်။ Tx ဝါယာနဲ Ground ဝါယာ ကြားက ဖိုက transmit voltage ဖြစ်ပြီး Rx ဝါယာ နဲ Ground ဝါယာ ကြားက ဖိုက receive voltage ဖြစ်ပါတယ်။

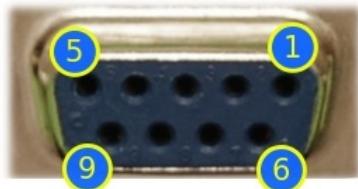
### ၇.၂.၁ Handshaking

RS232 communication မှာ ပုံမှန် အားဖြင့် data transmit line (Tx) နဲ data receive line (Rx) ကိုပဲ သုံး ပေမယ့် တစ်ခု နဲ တစ်ခု handshaking လုပ်ဖို့ အတွက် control နဲ status lines တွေ ကိုလဲ သုံးတတ် ကြ ပါတယ်။ RS232 device တွေ အကြောင်း ပြောတဲ့ အခါ မှာ ကွန်ပျူဗာ ကို DTE (Data Terminal Equipment) လို ခေါ်ပြီး ကွန်ပျူဗာ နဲ ဆက်သုံး တဲ့ Modem သို့ device ကို DCE (Data Circuit-Terminating Equipment) လို ခေါ် ပါတယ်။

ပုံ ၇.၃ မှာ ပြထား တဲ့ အတိုင်း ကွန်ပျူဗာ (DTE) မှာ RS232 အတွက် Male DB9 ပါပြီး သူနဲ ဆက်သုံး ရမယ့် (DCE) device မှာ Female DB9 connector ပါလေ့ရှိ ပါတယ်။



(a) Male DB9 connector on DTE



(b) Female DB9 connector on DCE

ပုံ ၇.၃: RS-232 အတွက် DB9 connector များ။

DTE (ဂုန်ပျိုးတာ) ဘက်က male connector ရဲ pin connection တွေက အောက်ပါ အတိုင်း ဖြစ်ပါတယ်။

1. CD- Carrier Detect (input)
2. Rx- Receive Data (input)
3. Tx- Transmit Data (output)
4. DTR- Data Terminal Ready (output)
5. GND- System Ground
6. DSR- Data Set Ready (input)
7. RTS- Request To Send (output)
8. CTS- Clear To Send (input)
9. RI - Ring Indicator (input)

DCE (Device) ဘက်က female connector ရဲ pin connection တွေကို တော့ အောက်က တာရင်းမှာ တွေ့နိုင်ပါတယ်။

1. CD- Carrier Detect (output)
2. Tx- Transmit Data (output)
3. Rx- Receive Data (input)
4. DTR- Data Terminal Ready (intput)
5. GND- System Ground
6. DSR- Data Set Ready (output)
7. CTS- Clear To Send (input)
8. RTS- Request To Send (output)

### 9. RI - Ring Indicator (output)

Primary Communication lines တွေ ထဲမှာ ဆို Tx နဲ့ RTS က အတွက် ဖြစ်ပြီး Rx နဲ့ CTS က အဝင် ဖြစ် ပါတယ်။ Status and Control lines တွေမှာ ဆိုရင် တော့ DTR က အတွက် လိုင်း ဖြစ်ပြီး DSR နဲ့ CD က အဝင် လိုင်း ဖြစ် ပါတယ်။

**RTS** Request To Send signal ကတော့ DTE ကနေ DCE ကို ပိုချင်တဲ့ အကြောင်း လှမ်း request လုပ်တာပါ။ ပုံမှန် အားဖြင့် logic 1 အနုတ် ဖို့မှာပဲ ရှိပြီး request လုပ်တာ နဲ့ logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။

**CTS** Clear To Send signal ကတော့ DCE က ပိုမို ready ဖြစ်တဲ့ အတွက် စ ပိုတော့ ဆိုတဲ့ အကြောင်း DTE ကို အကြောင်း ပြန် တာပါ။ ပုံမှန် အားဖြင့် logic 1 အနုတ် ဖို့မှာပဲ ရှိပြီး DCE ကနေ အကြောင်း ပြန်ဖို့ logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။

**DSR** DCE Ready (Data Set Ready) ကတော့ DCE (device) က turn on ဖြစ်ပြီး ready ဖြစ် နေပြီ ဆိုတဲ့ အကြောင်း DTE ကွန်ပျူးတာ ကို လှမ်းပြော တာပါ။ Ready ဖြစ်ပြီ ဆို DCE ကနေ logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။

**DTR** DTE Ready ကတော့ DTE ကွန်ပျူးတာ ကနေ communication ကို လုပ်ချင် လို့ အသင့် ပြင်တော့ လို့ DCE (device) ကို လှမ်းပြော တာပါ။ ပုံမှန် အားဖြင့် logic 1 အနုတ် ဖို့မှာပဲ ရှိပြီး DTE ကနေ DTR လိုင်းကို true လုပ်တဲ့ အခါ logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။

**CD** Carrier Detect (Received Line Signal Detector) ကတော့ DCE က modem ဖြစ်တဲ့ အခါမှာ သုံး ပါတယ်။ တယ်လီဖုန်း လိုင်း ကို တခြား ဘက် အဝေးမှာ ရှိတဲ့ modem က ဖြေ လိုက်တဲ့ answer tone ကို ရတဲ့ အခါမှာ ဒီဖက် modem က ကွန်ပျူးတာ ကို logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။

**RI** Ring Indicator ကတော့ ring signal ကို ရတဲ့ အခါ modem က ကွန်ပျူးတာ ကို logic 0 အပေါင်း ဖို့ ထုတ်ပေး ပါတယ်။ Ring signal ရတဲ့ အချိန်ပဲ အပေါင်း ထုတ် ပေးပြီး ring signal အချင်းချင်း ကြားနဲ့ ring signal မရှိတဲ့ အချိန် တွေမှာ အနုတ် ဖြစ်နေ ပါတယ်။

## ၇.၃ Hardware များ

အခု နောက်ပိုင်း desktop နဲ့ laptop ကွန်ပျူးတာ တွေမှာ serial port ပါ မလာ တော့ပဲ USB port တွေပဲ ပါတာ များ ပါတယ်။ အဲဒီ အတွက် RS232 port တွေ ပါတဲ့ PCI card တွေ ဈေးပေါပေါ့ နဲ့ ဝယ်တပ် လို့

ရပါတယ်။ Laptop တွေ အတွက် ပါ အဆင်ပြု တာက တော့ ပဲ ၇.၆ မှာ ပြထားတဲ့ USB to RS232 converter လေး တွေပါ။

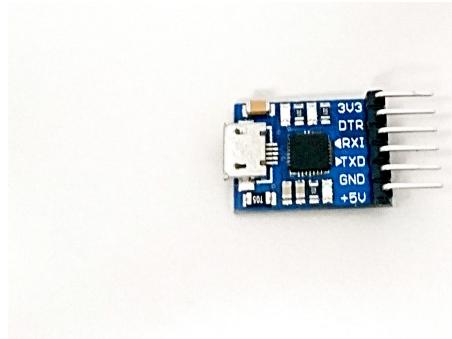


ပဲ ၇.၃: USB to RS232 converter တစ်ခု။

အဲဒီ USB to RS232 converter တွေက RS232 သုံးထား တဲ့ ပစ္စည်း၊ ကိရိယာ တွေနဲ့ ဆက်သွယ် အသုံးပြုဖို့ အတွက် အဆင်ပြု ပေမယ့် Raspberry Pi တို့လို UART ပဲ ပါပြီး 3.3V ပဲသုံးတဲ့ ကိရိယာ တွေ အတွက်တော့ ±12 V ထွက်တဲ့ RS232 pin တွေနဲ့ တိုက်ရှိက် ဆက်သုံးလို့ မရ ပါဘူး။ ဒါကြောင့် Raspberry Pi ကို RS232 ပစ္စည်း တွေနဲ့ ဆက်သုံး ချင်ရင် MAX202 တို့လို့ RS232 transceiver chip တစ်ခု ကြားမှာ ခံစိုးလိုပါတယ်။ Host computer နဲ့ Raspberry Pi ကို တိုက်ရှိက် ဆက်သွယ် ချင်ရင်တော့ USB to UART converter တွေဖြစ်တဲ့ FTDI cable (ပဲ ၇.၅) တို့၊ CP2102 USB to UART bridge (ပဲ ၇.၆) တိုကို သုံးနိုင် ပါတယ်။



ပဲ ၇.၅: FTDI cable တစ်ခု။



ပုံ ၇.၆: CP2102 USB to UART bridge တစ်ခု။

FTDI cable အမျိုးမျိုး ရှိဖြီး ပုံ ၇.၅ မှာ ပြထားတဲ့ ကြိုးက 3.3V TTL voltage level နဲ့ AliExpress က ဝယ်ထားတဲ့ ဈေးပေါ် တဲ့ အမျိုးအစား ပါ။ သူ အတွက် pin connection တွေကို တော့ အောက်က စာရင်းမှာ တွေ့နှင့် ပါတယ်။

၁။ အနက်ရောင်ကြိုး: GND - System Ground

၂။ အပြောရောင်ကြိုး: CTS- Clear To Send (input)

၃။ အနီရောင်ကြိုး: VCC - 5 V

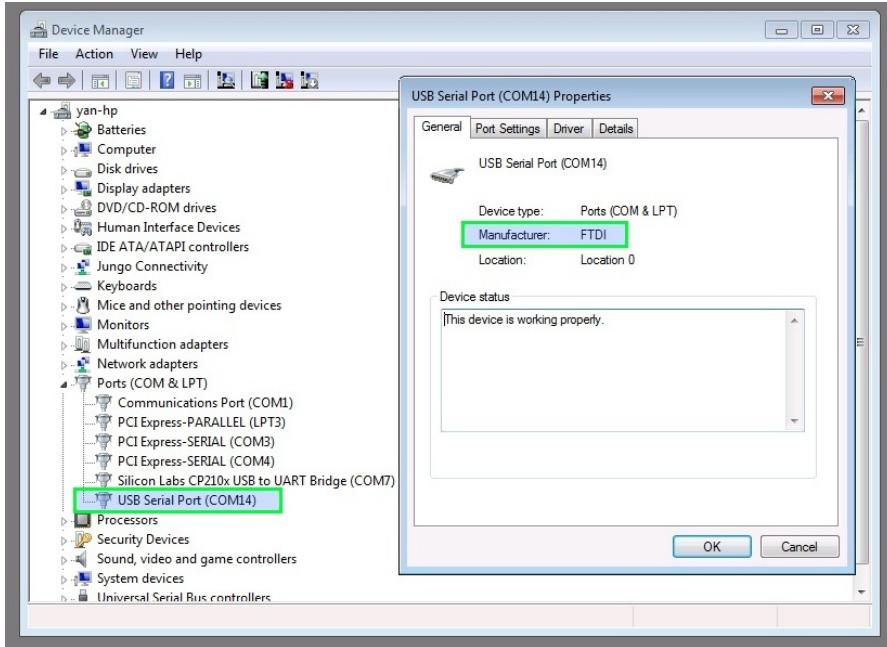
၄။ အစိမ်းရောင်ကြိုး: Tx- Transmit Data (output)

၅။ အဖြူရောင်ကြိုး: Rx- Receive Data (input)

၆။ အဝါရောင်ကြိုး: RTS- Request To Send (output)

### ၇.၂.၁ Windows တွင်အသုံးပြုခြင်း

FTDI cable ကို Windows စက်တွေ မှာ တပ်ဆင် တဲ့ အခါ ဘာ device driver မှ တပ်ဆင် စရာ မလို ပဲ တန်းပြီး အလုပ် လုပ် တာကို တွေ့ရ ပါတယ်။ သူရဲ့ COM port နာမည် ကို device manager မှာ တွေ့နှင့် ပါတယ် (ပုံ ၇.၇)။

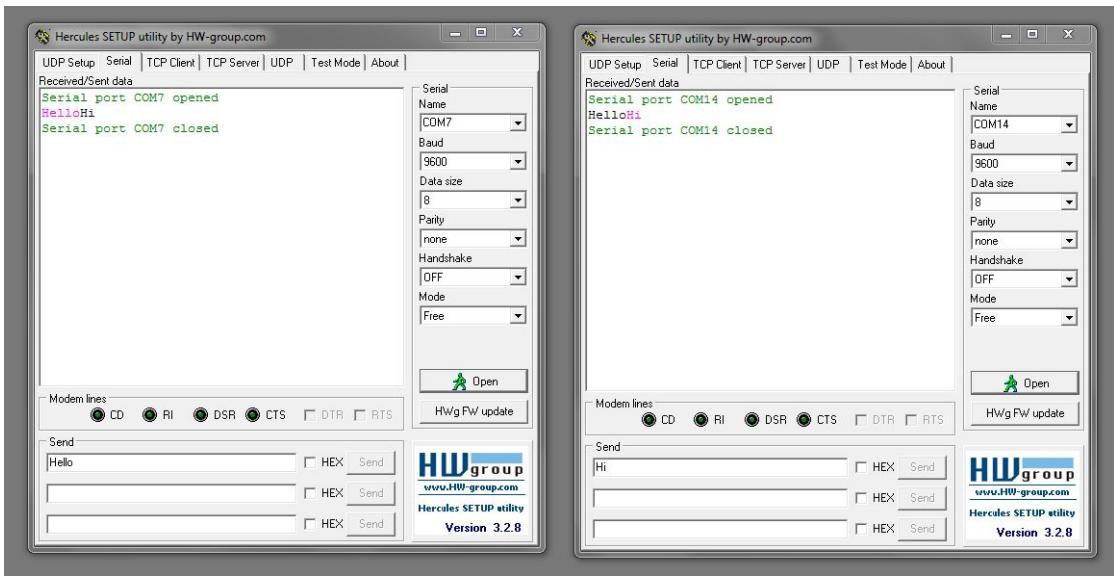


ပုံ ၇.၇: Device manager တွင် တွေ့နိုင်သော COM port နာမည်။

CP2102 အတွက် တော့ သူရဲ့ driver ကို Silicon Labs ရဲ့ ဝက်ဘ်စာမျက်နှာ ([www.silabs.com  
/products/development-tools/software/usb-to-uart-bridge-vcp-drivers](http://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers)) ကနေ download လုပ်ပြီး တပ်ဆင် ဖို့ လိုပါတယ်။ အခါ နောက် မှာတော့ device manager မှာ Silicon Labs CP210x USB to UART bridge လို့ တွေ့ရ မှာပါ။

သူတို့ကို သုံးပြီး ဒေတာ တွေ စမ်းပို့ ကြည့်ဖို့ အတွက် စက်နှစ်လုံး ပဲဖြစ်ဖြစ်၊ စက်တစ်လုံး ထဲမှာ device နှစ်ခု တပ်ပြီး ပဲ ဖြစ်ဖြစ်၊ အချင်းချင်း ဆက်ပြီး ပိုကြည့် နိုင် ပါတယ်။ အနည်းဆုံး ပါယာ သုံးခု ဆက်သွယ်ဖို့ လိုပါတယ်။ Ground အချင်းချင်း ဆက်၊ နောက် တစ်ခု ရဲ့ TX ကို တဗြား တစ်ခု ရဲ့ RX နဲ့ အပြန်အလုန် ဆက်သွယ်ဖို့ လိုပါတယ်။ အလုံး မဟုတ် ရင်လည်း device တစ်ခု ထဲကို ပဲ သူရဲ့ TX ကို RX နဲ့ ပြန်ဆက် ပေးပြီး loop back အနေ နဲ့ လည်း စမ်းချင် ရင် စမ်းကြည့် လို့ ရပါတယ်။

စမ်းသပ် သုံးစွဲ ကြည့်ဖို့ application အမျိုးမျိုး ရှိပြီး [www.hw-group.com/products/hercules/index\\_en.html](http://www.hw-group.com/products/hercules/index_en.html) မှာ free ရရှိနိုင်တဲ့ Hercules Utility က ကောင်းမွန် အဆင်ပြု တာကို တွေ့ရ ပါတယ်။ Hercules ကို Windows ကွန်ပျူတာ မှာ တပ်ဆင် လိုက်ပြီး၊ ပရိုဂရမ် ကို ဖွင့်ပြီး တဲ့ အခါ serial tab မှာ port number, baud rate စတာတွေ သတ်မှတ် ပြီး စမ်းသပ် ပေးပို့ ထားတာ ကို ပုံ ၇.၈ မှာ နမူနာ အနေနဲ့ တွေ့နိုင် ပါတယ်။



ံ ၇.၈: Hercules ဖြင့် serial port များကို စမ်းသပ်ခြင်း။

### ၇.၃.၂ Linux တွင်အသုံးပြုခြင်း

Linux ပေါ်မှာ တော့ FTDI cable ရော့၊ CP2012 ရော့ ဘာမှ ထပ်လုပ် စရာ မလိုပဲ တန်း သုံးလို့ ရတာ ကို တွေ့ရ ပါတယ်။ Serial port ကဲ နာမည် ကို dmesg သုံးပြီး အောက်က စာရင်း နဲ့ ပုံ ၇.၉ မှာ ပြထား သလို ကြည့်နိုင် ပါတယ်။

```
$ dmesg | grep -ie FTDI
```

```
yan@linux:~$ dmesg | grep FTDI
[89986.547018] usb 1-9: Manufacturer: FTDI
[89987.590138] usbserial: USB Serial support registered for FTDI USB Serial Device
[89987.590380] ftdi_sio 1-9:1.0: FTDI USB Serial Device converter detected
[89987.590742] usb 1-9: FTDI USB Serial Device converter now attached to ttyUSB0
yan@linux:~$
```

ံ ၇.၉: FTDI cable ၏ serial port ကို စစ်ခြင်း။

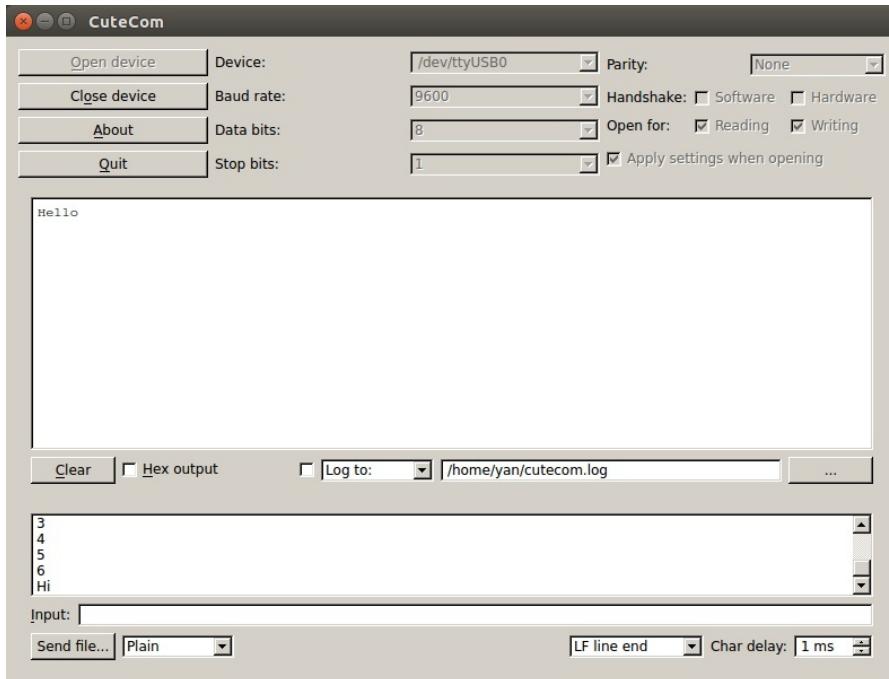
CP2102 အတွက် လည်း အောက်က command နဲ့ စစ် ကြည့်နိုင် ပါတယ်။

```
$ dmesg | grep -ie cp210*
```

Linux ပေါ်မှာ လည်း COM port ကို သုံးဖို့ application အမျိုးမျိုး ရှိပြီး cutecom , GtkTerm စတဲ့ application တွေက အသုံးပြု ရတာ လွယ်ကူ အဆင်ပြေ ပါတယ်။ GtkTerm ကို Ubuntu Software မှာ Serial Port Terminal ဆိုတဲ့ နာမည် နဲ့ တွေ့နိုင် သလို၊ စာရင်း ၇.၁ မှာ ပြထား သလို terminal မှာ superuser အနေနဲ့ တပ်ဆင် အသုံးပြု လိုလည်း ရ ပါတယ်။ CuteCom ကို တပ်ဆင် တဲ့ run တဲ့ အခါ မှာလည်း superuser အနေနဲ့ အောက်က စာရင်း ၇.၂ နဲ့ ပုံ ၇.၁၀ မှာ ပြထား သလို တပ်ဆင် အသုံးပြု လိုရ ပါတယ်။

```
1 $ sudo apt install gtkterm
2 $ sudo gtkterm
```

စာရင်း ၇.၁: GtkTerm ကိုတပ်ဆင်အသုံးပြုခြင်း။



ပုံ ၇.၁၀: CuteCom ကိုအသုံးပြုခြင်း။

```
1 $ sudo apt install cutecom
2 $ sudo cutecom
```

စာရင်း ၇.၂: CuteCom ကိုတပ်ဆင်အသုံးပြုခြင်း။

တကယ်တော့ serial port တွေကို အသုံးပြု ဖို့ အတွက် dialout ဆိုတဲ့ user group ထဲမှာ ပါရှိ နေဖို့လိုတာပါ။ နောက်ပိုင်း တချို့ system တွေမှာ တော့ dialout အစား tty လည်း ဖြစ်နိုင် ပါတယ်။ အဲဒါ ကို အောက်ပါ အတိုင်း စစ် ကြည့်နိုင် ပါတယ်။

```
$ ls -l /dev/ttyUSBO
```

အဲဒီ အခါ crw-rw-- 1 root dialout စသဖြင့် တွေ့နိုင် ပြီး character device, root နဲ့ dialout group တွေပဲ ရေးဖတ် လို့ရပြီး တခြား သူတွေ သုံးလို မရ ဘူး လို ဆိုလို ပါတယ်။ လက်ရှိ current user က dialout ထဲမှာ ပါမ ပါ အောက်က အတိုင်း စစ်နိုင် ပါတယ်။

```
$ id
```

မပါ သေးရင် အောက်က command သုံးပြီး username နေရာ မှာ လက်ရှိ user ရဲ့ နာမည် နဲ့ ထည့်နိုင် ပါတယ်။

```
$ sudo usermod -a -G dialout username
```

## ၇.၄ RPi ၏ serial port ကိုသုံးခြင်း

Raspberry Pi 3 မှာ UART device နှစ်ခု ပါပြီး ပြည့်စုံ ကောင်းမွန် တဲ့ ttyAMA0 က bluetooth အတွက် သုံးထား ပါတယ်။ ကျေန်တဲ့ ttyS0 လို ခေါ်တဲ့ UART ကို GPIO header ရဲ့ pin8 (UART0\_TXD) နဲ့ pin10 (UART0\_RXD) မှာ ဆက်သွယ်ပြီး အသုံးပြု နိုင် ပါတယ်။ အဲဒီ ကျေန်တဲ့ UART က parity ကို လည်း အထောက်အပံ့ မပေး သလို၊ Baudrate ကလည်း system clock ကနေယူတွက် ထားတာ မို့ CPU ရဲ့ clock frequency ပြောင်းတဲ့ အခါ တွေမှာ လည်း ပြဿနာ တွေရှိ ပါတယ်။ ဒါကြောင့် Raspberry Pi နဲ့ seriar port ကို သုံးဖို့ လိုလာ ရင် ရှေ့မှာ ဖော်ပြ ခဲ့ သလို FTDI cable တို့ CP2102 တို့ကို သူ့ရဲ့ USB မှာ တပ်ပြီး သုံးတာက အလွယ်ကူဆုံး နဲ့ အဆင် အပြောင်း ပါ။

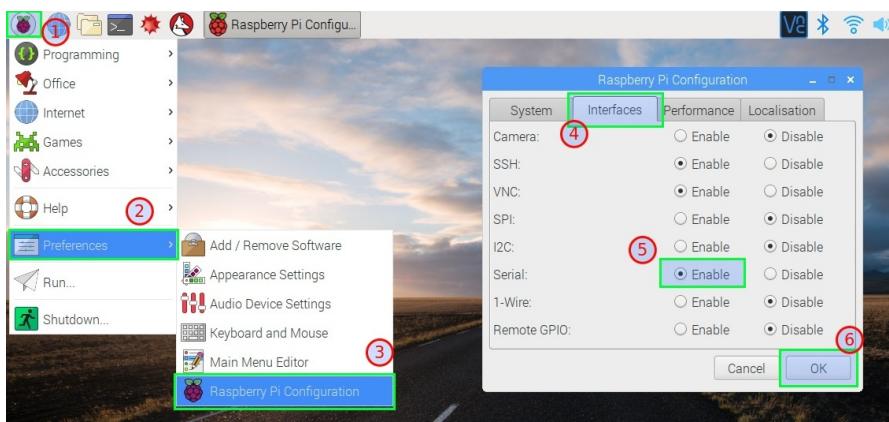
Raspberry Pi ရဲ့ GPIO header က ttyS0 ကို သုံးလို ရတဲ့ ပုံစံ နှစ်မျိုး ရှိပါတယ် [Eli17a]။

**Linux Console:** PC ကို GPIO header ရဲ့ serial port နဲ့ ဆက်ပြီး RPi အတွက် Linux console အနေနဲ့ အသုံးပြု နိုင်ပါတယ်။ အဲဒါ က boot လုပ်တဲ့ အချိန် မှာ ရှိတဲ့ ပြဿနာ တွေ၊ video နဲ့ network တွေ မရ တော့တဲ့ အခါမျိုးတွေ မှာ အသုံးကျ ပါတယ်။

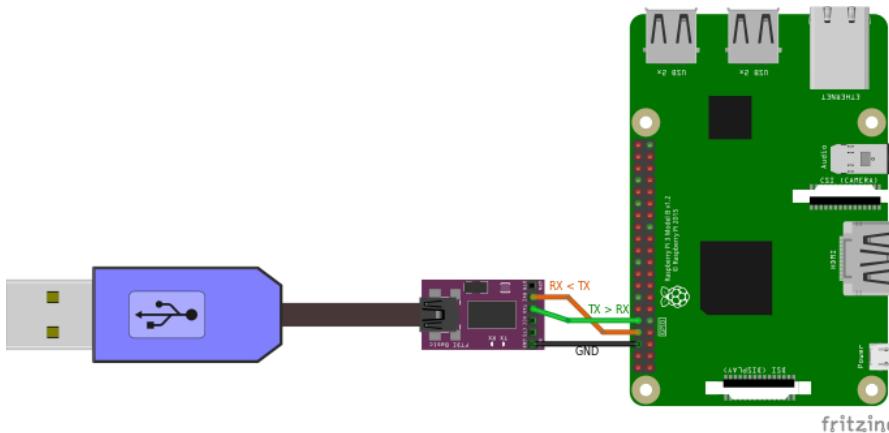
**Serial Interface:** RPi အတွက် serial interface အနေနဲ့ တစ်ခြား device တွေနဲ့ ဆက်သွယ် အသုံးပြုနိုင် ပါတယ်။

### ၇.၄.၁ Linux Console

Raspberry Pi ရဲ့ serial port ကို Console အနေနဲ့ ဆက်သွယ် အသုံးပြုဖို့ အတွက် ပုံ ၇.၁၁ မှာ ပြထားသလို Applications menu → Preferences → Raspberry Pi Configuration → Interfaces tab → Serial မှာ Enable ကို ရွေးပြီး reboot လုပ်ဖို့ လိုပါတယ်။ PC နဲ့ Raspberry Pi ကို FTDI cable ဖြစ်ဖြစ်သုံးပြီး ပုံ ၇.၁၂ မှာ ပြထားသလို ဆက်သွယ် လိုက် ပါမယ်။



ပုံ ၇.၁၁: Serial console ကို အသုံးပြုနိုင်ရန် configure လုပ်ခြင်း။



ပုံ ၇.၁၂: PC နဲ့ RPi ၏ serial port ကို ဆက်သွယ်ခြင်း။

အဲဒီနောက် PC မှာ ရှုမှုံးဖြင့်ဖြစ်ပွားသူများဖြင့် RPi အတွက် console အနေနဲ့ လုပ်လိုက်ပါ။ ဒါနောက် မှတ်တမ်းဆိုင်ရာ အတွက် serial interface အတွက် settings တွေကို အောက်က စာရင်းမှာ ပြထားပါတယ်။

**Baud rate:** 115200

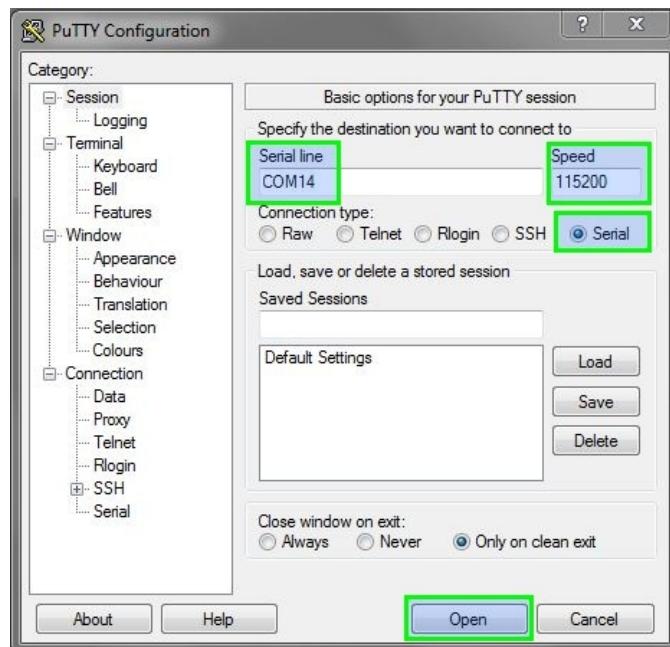
**Data bits:** 8

**Parity:** N

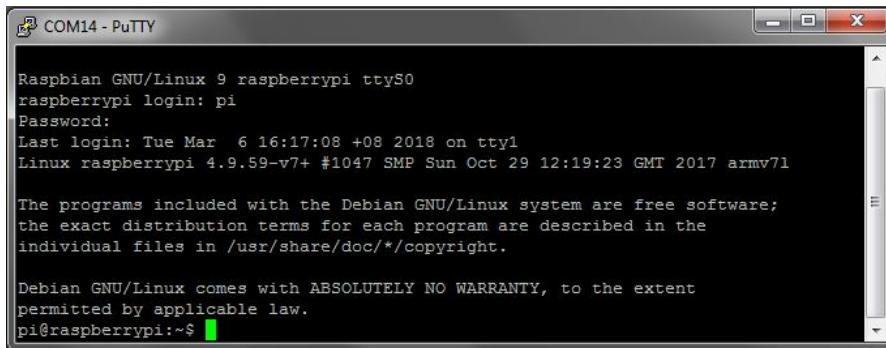
**Stop bits:** 1

**Flow control:** None

PuTTY မှာ connection type အတွက် Serial ကို ရွေး၊ baud rate ကို 115200 ကို သတ်မှတ်၏ Port နာမည် ကို device manager မှာ တွေ့နိုင် တဲ့ နာမည် ကို ထည့်ဖြီး Open ကို နိုပ်နိုင် ပါတယ် (ပုံ ၇.၁၃)။ ပြီးတဲ့ အခါ RPi ကို ပါဝါ ဖွင့်လိုက် ရင် ပုံ ၇.၁၄ မှာ ပြထား သလို console ကို အသုံးပြုနိုင် တာကို တွေ့ရ မှာပါ။



ပုံ ၇.၁၃: PuTTY သုံး၍ serial port ကို ဆက်သွယ်ခြင်း။



ပုံ ၇.၁၄: PuTTY ရှိ console။

အကယ်၍ Linux PC အတွက် ဆိုရင်တော့ CuteCom , GtkTerm စိုးအပြင် screen ဆိုတဲ့ application ကိုလည်း အောက် က အတိုင်း ရှိက်ပြီး သုံးလို့ရ ပါတယ်။

```
sudo apt-get install screen
screen /dev/ttyUSB0 115200
```

Screen ကို အဆုံးသတ် ချင်ရင် ctrl+a နှိပ်ပြီး capital K ကို နှိပ်၊ y ကို နှိပ်ပြီး အဆုံးသတ် နှင့် ပါတယ်။ ရှိပြီး သား screen session တွေ အကုန် အဆုံးသတ် ချင်ရင် တော့

```
screen -ls | grep pts | cut -d. -f1 | awk '{print $1}' | xargs kill
```

ကို ထည့် နှင့် ပါတယ်။

#### ၇.၄.J Serial Interface

Raspberry Pi ရဲ့ serial port ကို console အနေနဲ့ မသုံးပဲ serial interface အနေနဲ့ ပဲ သုံးမယ် ဆိုရင် Applications menu → Preferences → Raspberry Pi Configuration → Interfaces tab → Serial မှာ Disable ကို ရွေးဖို့ လိုပါတယ်။ ပြီးတဲ့ အခါ /boot/config.txt ကို အောက်က အတိုင်း edit လုပ်ဖို့ဖွင့်နှင့် ပါတယ်။

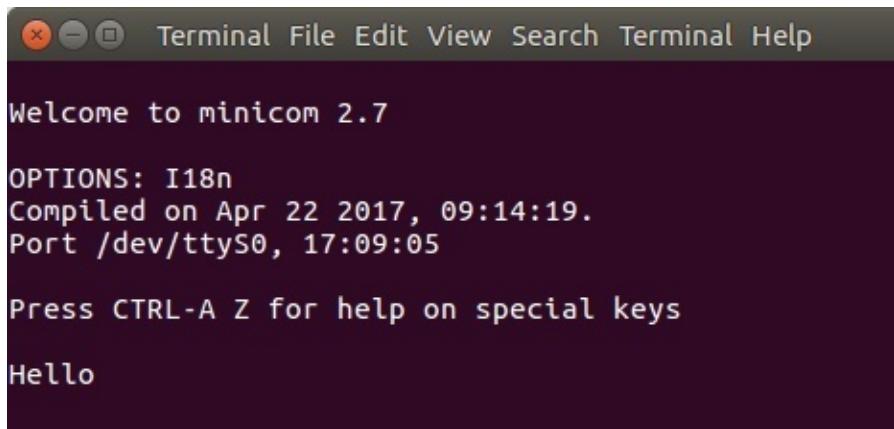
```
$ sudo nano /boot/config.txt
```

အဲဒီ ဖို့ ထဲမှာ enable\_uart=1 ဆိုတဲ့ စာကြောင်းကို ဖြည့်ပြီး တဲ့ အခါ Ctrl+O နဲ့ သိမ်း၊ Ctrl+X နဲ့

ထွက်ပြီး RPi ကို reboot လုပ်လိုက်ပါမယ်။ အဲဒီနောက် မှာတော့ ttyS0 ကို GtkTerm, CuteCom စတဲ့ အဆင်ပြုရာ application တစ်ခု ခု ဖွင့်ပြီး သုံးလို ရတာ ကို တွေ့ရ ပါလိမ့်မယ်။ SSH တို့လို terminal ပေါ်မှာ သုံးပြီး ပို့ချင်ရင် လည်း minicom တို့လို application ကို RPi ပေါ်မှာ အောက်က အတိုင်း တပ်ဆင် အသုံးပြု နိုင် ပါတယ်။

```
sudo apt install minicom
minicom -b 9600 -D /dev/tty04
```

ပြီးတဲ့ အခါ host ကွန်ပူးတာ နဲ့ RPi တို့ကြား အပြန်အလှန် ဆက်သွယ် ပေးပို့ လို ရတာကို ပုံ ရော်မှာ ပြထားတဲ့ အတိုင်း တွေ့နိုင် ပါတယ်။ Minicom မှာ ရိုက်ထည့် တဲ့ စာလုံးကို ပေါ်ချင်ရင် local echo ကို enable လုပ်နိုင် ပါတယ်။ CTRL-A ကို နှိပ်ပြီး တဲ့ အခါ z ကို နှိပ်လိုက်ပြီး၊ command summary ပေါ်လာရင် e ကို နှိပ်လိုက် တဲ့ အခါ local echo ON သွား ပါလိမ့်မယ်။ ထွက်ချင်ရင် တော့ CTRL-A → z → x → enter ကို နှိပ် နိုင် ပါတယ်။



ပုံ ရော်: Minicom ကို သုံးခြင်း

## ၇.၅ Programming Serial Port

တစ်ခြား device တွေလို ပါပဲ။ Linux ပေါ်မှာ serial port ကို device file အနေနဲ့ ဖွင့်ပြီး သုံးနိုင် ပါတယ် [Wik16]။ Serial port တစ်ခု ကို သုံးဖို့ အတွက် သူ့နဲ့ သက်ဆိုင်ရာ ttyS0, ttyUSB0 စတဲ့ device file ကို ဖွင့်ဗျာ၊ ရေးတာ၊ ဖတ်တာ တွေကို လုပ်နိုင် ပါတယ်။ အဲဒါ တွေ အတွက် terminal နဲ့ ပတ်သက်တဲ့ သတ်မှတ် ချက်တွေ ပါတဲ့ termios.h၊ ဖိုင် ထိန်းချုပ်မှု နဲ့ဆိုင်တဲ့ fcntl.h၊ UNIX စနစ်တွေ အတွက်

unistd.h စာတွဲ header ဖိုင်တွေကို ထည့်ပါမယ်။

```
#include <termios.h>
#include <fcntl.h>
#include <unistd.h>
```

Serial port ရဲ့ configuration တွေကို struct termios ဆိုတဲ့ data structure ကို သုံးပြီး သတ်မှတ် နိုင်ပါတယ်။ အဲဒီ မှာ data bits, parity, stop bits စာတွေ တွေကိုလို သလို ပြုပြင် နိုင်ပါတယ်။

```
struct termios settings;
memset(&settings, 0, sizeof(settings));
settings.c_iflag = 0;
settings.c_oflag = 0;
settings.c_cflag = CREAD | CLOCAL; //see termios.h for more information
settings.c_cflag |= CS8; // 8 data bits
settings.c_cflag |= PARENB; //no parity
//1 stop bit if CSTOPB is not set
```

Serial port ကို ဖွဲ့စွဲ အခါ open ကို သုံးပြီး အောက်က အတိုင်း ဖွဲ့စွဲနိုင်ပါတယ်။ Read/Write အတွက် Non-blocking mode နဲ့ ဖွဲ့စွဲမယ် လို ဆိုလို တာပါ။ အဲလို မဟုတ်ပဲ Blocking mode ဆိုရင် ရေးတဲ့ ဖတ်တဲ့ အခါ လုပ်ဆောင် မူ မပြီး မချင်းပရိုကရမဲ့ က ရပ်စောင့် နေပါမယ်။

```
int fd; //file descriptor
fd = open(SERIAL_PATH, O_RDWR | O_NONBLOCK);
```

Serial port ကို ဒေတာ တွေ ရေးလို ဖတ်ဖို့ အတွက် တော့ write နဲ့ read ဖန်ရှင် တွေကို သုံးနိုင်ပါတယ်။ Device file ရယ်၊ ရေးလို၊ သိမ်းလို တဲ့ character buffer ရယ်၊ ရေးဖတ်မယ့် အရေး အတွက် နဲ့ အောက်ပါ အတိုင်း သုံးလို ရ ပါတယ်။ အမှန် တကယ် ရေးလို ဖတ်လို ရလိုက်တဲ့ အရေး အတွက် က return အနေ နဲ့ ပြန်ရ ပါတယ်။

```
char buffer_send[32] = "Test\r\n";
char buffer_recv[32] = {0};
int write_ret = write(fd, buffer_send, 6);
int read_ret = read(fd, buffer_recv, 6);
```

Serial port ကို ပိတ်ဖို့ အတွက်တော့ close ကို သုံးနိုင် ပါတယ်။

```
close(fd);
fd=-1;
```

နှမူနာ အနေနဲ့ serial port ကို အသုံးပြုတဲ့ ရိုးရှင်းတဲ့ [SerialSimple.cpp](#) ဆိုတဲ့ ပရိဂရမ် လေးကို စာရင်း ၇.၃ မှာ ပြထား ပါတယ်။

```
1 // File: SerialSimple.cpp
2 // Description: A simple program to use serial port
3 // Author: Yan Naing Aye
4
5 #include <stdio.h>
6 #include <termios.h>
7 #include <unistd.h>
8 #include <fcntl.h>
9 #include <string.h>
10
11 #define BAUDRATE B115200
12 // #define SERIAL_PATH "/dev/ttyUSBO"
13 #define SERIAL_PATH "/dev/ttyS0"
14
15 int main() {
16     int fd; //file descriptor
17     struct termios settings;
18     char buffer_send[32] = "Test\r\n";
19     char buffer_recv[32] = {0};
20
21 //open serial port
22     printf("Opening serial port.\n");
23     memset(&settings, 0, sizeof(settings));
24     settings.c_iflag = 0;
25     settings.c_oflag = 0;
26     settings.c_cflag = CREAD | CLOCAL; //see termios.h for more information
```

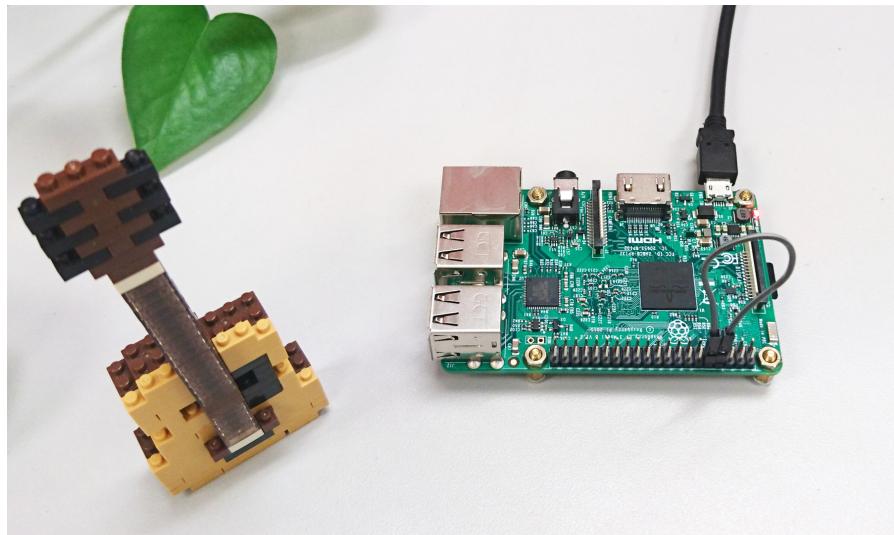
```
27     settings.c_cflag |= CS8;// 8 data bits
28     settings.c_cflag |= PARENB;//no parity
29 //1 stop bit if CSTOPB is not set
30     settings.c_lflag = 0;
31     settings.c_cc[VMIN] = 1;
32     settings.c_cc[VTIME] = 0;
33     fd = open(SERIAL_PATH, O_RDWR | O_NONBLOCK);
34     if (fd == -1) {
35         printf("Error in opening serial port!\n");
36         return -1;
37     }
38     printf("Port opened successfully.\n");
39     cfsetospeed(&settings, BAUDRATE);
40     cfsetispeed(&settings, BAUDRATE);
41     tcsetattr(fd, TCSANOW, &settings);
42     int flags = fcntl(fd, F_GETFL, 0);
43     fcntl(fd, F_SETFL, flags | O_NONBLOCK);
44
45 //write
46     printf("Writing: %s\n",buffer_send);
47     int write_ret = write(fd,buffer_send,strlen(buffer_send));
48     printf("Written: %d\n",write_ret);
49
50 //wait a while
51     usleep(1000000);
52
53 //read
54     int read_ret = read(fd,buffer_recv,strlen(buffer_send));
55     printf("Received: %s\n",buffer_recv);
56     printf("Read: %d\n",read_ret);
57
58 //close
59     printf("Closing serial port.\n");
60     close(fd);
61     fd=-1;
62
```

```

63     return 0;
64 }
```

စာရင်း ၇.၃: SerialSimple.cpp

သူကို စမ်းကြည့် ဖို့ အတွက် ပုံ ၇.၁၆ မှာ ပြထားသလို Raspberry Pi ၏ TX pin နဲ့ RX pin ကို loop back ပြန်ဆက် ပြီး run ကြည့်နိုင် ပါတယ်။ ရလာ တဲ့ ရလဒ် ကို ပုံ ၇.၁၇ မှာ ပြထားပါတယ်။



ပုံ ၇.၁၆: TX နဲ့ RX ကို loop back ဆက်သွယ်ခြင်း။

```

pi@raspberrypi:~/SerialSimple
File Edit Tabs Help
pi@raspberrypi:~ $ cd SerialSimple
pi@raspberrypi:~/SerialSimple $ ./bar
Building
Running
Opening serial port.
Port opened successfully.
Writing: Test

Written: 6
Received: Test

Read: 6
Closing serial port.
pi@raspberrypi:~/SerialSimple $
```

ပုံ ၇.၁၇: SerialSimple.cpp ၏ ရလဒ်။

### ၇.၅.၁ Linux နှင့် Windows အတွက် Serial Class

Raspberry Pi အတွက် ပါမက ပဲ သူနဲ့ အပြန် အလှန် ဆက်သွယ်မယ့် host computer တွေ အတွက် ပါ serial port ကို သုံးဖို့ လိုတတ် ပါတယ်။ အဲဒီ အတွက် Linux ၁ Windows စက်တွေ နဲ့ Raspberry Pi အတွက် ပါ သုံးနိုင်တဲ့ ကျွန်ုတ် ဖန်တီး ထားတဲ့ C++ class library လေး တစ်ခု ကို နမူနာ ပြောချင် ပါတယ် [Lab17; Den95]။ ပထမ နမူနာ အနေနဲ့ ရိုးရှင်းတဲ့ C++ console program လေး တစ်ခု ကို ဖော်ပြ ထားပြီး၊ နောက်ထပ် GUI application နမူနာ အနေနဲ့ wxWidgets ကို သုံးထား တဲ့ program ကိုပါ ဖော်ပြထား ပါတယ်။ Serial class ရဲ့ source code (`ce_serial.h`) ကို နောက်ဆက်တဲ့ A - စာရင်း ၁.၃ မှာ တွေ့နိုင် ပါတယ်။

### ၇.၅.၂ Console

ကွန်ပျူးတာ ရဲ့ serial port ကို ဖွင့်ပြီး ဒေတာ ပို့ ပြီးတော့ character တစ်လုံး ကို ပြန် ဖတ်ပြ တဲ့ ရိုးရှင်းတဲ့ `serialcon.cpp` ဆိုတဲ့ နမူနာ လေးကို စာရင်း ၇.၄ မှာ ပြထား ပါတယ်။

```

1 //File: serialcon.cpp
2 //Description: Serial communication console program for Windows and Linux
3 //WebSite: http://cool-emerald.blogspot.sg/2017/05/serial-port-programming-in-
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 #include<stdio.h>
8 #include "ce_serial.h"
9 using namespace std;
10 int main()
11 {
12
13 #if defined (__WIN32__) || defined(_WIN32) || defined(WIN32) || defined(
14     __WINDOWS__) || defined(__TOS_WIN__)
15     Serial com("\\\\.\COM1",9600,8,'N',1); //Windows
16 #else
17     Serial com("/dev/ttys0",9600,8,'N',1); //Linux
18 #endif

```

```

19 printf("Opening port %s.\n", com.GetPort().c_str());
20 if (com.Open() == 0) {
21     printf("OK.\n");
22 }
23 else {
24     printf("Error.\n");
25     return 1;
26 }
27
28 bool successFlag;
29 printf("Writing.\n");
30 char s []="Hello";
31 successFlag=com.Write(s); //write string
32 successFlag=com.WriteChar('!'); //write a character
33
34 printf("Waiting 3 seconds.\n");
35 delay(3000); //delay 5 sec to wait for a character
36
37 printf("Reading.\n");
38 char c=com.ReadChar(successFlag); //read a char
39 if (successFlag) printf("Rx: %c\n", c);
40
41 printf("Closing port %s.\n", com.GetPort().c_str());
42 com.Close();
43 return 0;
44 }
```

စာရင်း ၇.၄: serialcon.cpp

အစ မှာ #include "ce\_serial.h" လို ကြော်ပြီး Serial class ကို ထည့်လိုက် ပါတယ်။ ပြီးတော့ 'com' လို ခေါ်တဲ့ object တစ်ခု ကို port number, baud rate စတဲ့ တန်ဖိုး တွေနဲ့ ကြော်ပြီး ပါတယ်။ ဒီနမူနာ အတွက် ဆိုရင် Windows မှာ COM1 ကို သုံးမှာ ဖြစ်ပြီး၊ Linux မှာ ttyS0 ကို သုံးမှာ ဖြစ် ပါတယ်။ Linux မှာ USB device ကို သုံးမယ် ဆိုရင်တော့ ttyUSB0 ဖြစ်ကောင်း ဖြစ်နိုင် ပါတယ်။ Parity အတွက် ကဲ 'N','E','O','M', သို့ 'S' တို့ကို none, even, odd, mark, နဲ့ space တို့ အတွက် သုံးလို ရ ပါတယ်။

Comm port ကိုဖွင့်ဖို့အတွက် com.Open() ကိုသုံးနိုင်ပါတယ်။ Write နဲ့ WriteChar methods တွေကိုသုံးပြီး null terminated string ဒါမှမဟုတ် character တစ်လုံးကိုပို့နိုင်ပါတယ်။ ReadChar method နဲ့ character ကိုဖတ်တဲ့အခါစောင့်မနေပဲ non-blocking ပုံစံနဲ့ဖတ်မှာဖြစ်တဲ့အတွက် character ရှိမနေရင် successFlag က false လို့ပြုပါမယ်။ Close method နဲ့com ကိုဝိတ်ပါတယ်။

Control နဲ့status လိုင်းတွေကိုလည်းအသုံးပြုနိုင်ပါတယ်။ RTS နဲ့DTR lines တွေကိုcontrolလုပ်နိုင်ပြီး CTS တို့၊ DSR တို့လို့Status lines တွေကိုလည်းဖတ်နိုင်ပါတယ်။ ဒါပရိုဂရမ် "serialcon.cpp" ကိုUbuntu Linux ပေါ်မှာအောက်ကအတိုင်းcompiledလုပ်၊ runလုပ်နိုင်ပါတယ်။

```
g++ serialcon.cpp -o serialcon
sudo ./serialcon
```

Windowsပေါ်မှာတော့ Visual Studio ဒါမှမဟုတ် tdm-gcc တို့၊ mingw တို့လို့ compiler တစ်ခုခုကိုသုံးပြီးအောက်ကလိုပျိုးcomplieလုပ်ပြီးrunနိုင်ပါတယ်။

```
g++ serialcon.cpp -o serialcon.exe -std=c++11
.\serialcon.exe
```

## ၇.၅.၃ GUI

နောက်ထပ်နမူနာတစ်ခုအနေနဲ့wxWidgets နဲ့GUI application တစ်ခုဖန်တီးပြီးserial port ကိုအသုံးပြုတဲ့အကြောင်းဆွေးနွေးပါမယ်။ File menu ကနေသုံးမယ့်serial port၊ baud rate စေားတွေကိုရွေးချယ်သတ်မှတ်လို့ရပြီးလက်ခံရရှိတဲ့ဒေတာတွေကိုtext box မှာဖော်ပြုပေးနိုင်ပါတယ်။ အပေါ်ဘက်ကtext box ထမှာပို့ချင်တဲ့text ကိုထည့်ပြီးsend ဆလုတ်ကိုနှိပ်ပြီးပို့နိုင်ပါတယ်။ Control နဲ့status လိုင်းတွေကိုအသုံးပြုပုံကိုပါပြထားပါတယ်။ ဒါwxserial.cppဆိုတဲ့နမူနာပရိုဂရမ်ကိုစာရင်း[၇.၅](#)မှာဖော်ပြထားပါတယ်။

```
1 //File: wxserial.cpp
2 //Description: Serial communication for wxWidgets
3 //WebSite: http://cool-emerald.blogspot.sg/2017/05/serial-port-programming-in
           -c-with.html
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
```

```
6
7 // For compilers that support precompilation, includes "wx/wx.h".
8 #include "wx/wxprec.h"
9
10 #ifdef __BORLANDC__
11     #pragma hdrstop
12 #endif
13
14 // for all others, include the necessary headers (this file is usually all
15 // you
16 // need because it includes almost all "standard" wxWidgets headers)
17 #ifndef WX_PRECOMP
18     #include "wx/wx.h"
19 #endif
20
21 // the application icon (under Windows and OS/2 it is in resources and even
22 // though we could still include the XPM here it would be unused)
23 #ifndef wxHAS_IMAGES_IN_RESOURCES
24     #include "sample.xpm"
25 #endif
26
27 #include <wx/numdlg.h>
28
29 #include "ce_serial.h"
30
31 // Define a new application type, each program should derive a class from
32 // wxApp
33 class MyApp : public wxApp
34 {
35     public:
36         virtual bool OnInit();
37     };
38
39 class MyFrame : public wxFrame
40 {
41     public:
```

```
40 // ctor(s)
41     MyFrame(const wxString& title);
42     wxButton *btnSend;
43     wxTextCtrl *txtSend;
44     Serial com;
45     wxTimer m_timer;
46     wxTextCtrl *txtRx;
47     wxCheckBox *chkRTS;
48     wxCheckBox *chkDTR;
49     wxCheckBox *chkCTS;
50     wxCheckBox *chkDSR;
51     wxCheckBox *chkRI;
52     wxCheckBox *chkCD;
53     // event handlers (these functions should _not_ be virtual)
54     void OnQuit(wxCommandEvent& event);
55     void OnAbout(wxCommandEvent& event);
56     void OnOpen(wxCommandEvent& event);
57     void OnClose(wxCommandEvent& event);
58     void SelPort(wxCommandEvent& event);
59     void SetDataSize(wxCommandEvent& event);
60     void SetParity(wxCommandEvent& event);
61     void SetStopBits(wxCommandEvent& event);
62     void SetBaud(wxCommandEvent& event);
63     void OnSend(wxCommandEvent& event);
64     void OnTimer(wxTimerEvent& event);
65     void ProcessChar(char ch);
66     void ClearText(wxCommandEvent& event);
67     void OnChkRTS(wxCommandEvent& event);
68     void OnChkDTR(wxCommandEvent& event);
69     void UpdateCommStatus();
70 private:
71
72 };
73
74 // IDs for the controls and the menu commands
75 const int ID_BTNSEND = 101;
```

```
76 const int ID_TXTSEND = 102;
77 const int ID_CHKRTS = 103;
78 const int ID_BAUDRATE = 103;
79 const int ID_TIMER = 104;
80 const int ID_TXTRX = 105;
81 const int ID_CHKDTR = 106;
82 const int ID_SELPORT = 107;
83 const int ID_CHKCTS = 108;
84 const int ID_CHKDSR = 109;
85 const int ID_CHKRI = 110;
86 const int ID_CHKCD = 111;
87 const int ID_DATASIZE = 112;
88 const int ID_PARITY = 113;
89 const int ID_STOPBITS = 114;
90
91 enum
92 {
93     Button_Send = ID_BTNSEND,
94     Txt_Send = ID_TXTSEND,
95     Chk_RTS = ID_CHKRTS,
96     Serial_Baud = ID_BAUDRATE,
97     Timer1 = ID_TIMER,
98     Txt_Rx = ID_TXTRX,
99     Chk_DTR = ID_CHKDTR,
100    Serial_Port = ID_SELPORT,
101    Serial_DataSize = ID_DATASIZE,
102    Serial_Parity = ID_PARITY,
103    Serial_StopBits = ID_STOPBITS,
104    Txt_Clear = wxID_CLEAR,
105    Serial_Open = wxID_OPEN,
106    Serial_Close = wxID_CLOSE,
107    Minimal_Quit = wxID_EXIT,
108
109    Minimal_About = wxID_ABOUT
110
111};
```

## 7.0. PROGRAMMING SERIAL PORT

JOR

```
112
113 IMPLEMENT_APP(MyApp)
114 // 'Main program' equivalent: the program execution "starts" here
115 bool MyApp::OnInit()
116 {
117     // call the base class initialization method, currently it only parses a
118     // few common command-line options but it could be do more in the future
119     if ( !wxApp::OnInit() )
120         return false;
121
122     // create the main application window
123     MyFrame *frame = new MyFrame(wxT("Serial Com"));
124
125     // and show it (the frames, unlike simple controls, are not shown when
126     // created initially)
127     frame->Show(true);
128
129     // success: wxApp::OnRun() will be called which will enter the main
130     // message
131     // loop and the application will run. If we returned false here, the
132     // application would exit immediately.
133     return true;
134 }
135
136 // frame constructor
137 MyFrame::MyFrame(const wxString& title)
138     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(390, 280),
139               wxDEFAULT_FRAME_STYLE ^ wxRESIZE_BORDER), m_timer(this, ID_TIMER)
140 {
141     // set the frame icon
142     SetIcon(wxICON(sample));
143
144 #if wxUSE_MENUS
145     // create a menu bar
146     wxMenu *fileMenu = new wxMenu;
```

```
147 //Edit menu
148 wxMenu *editMenu = new wxMenu;
149
150 // the "About" item should be in the help menu
151 wxMenu *helpMenu = new wxMenu;
152
153 helpMenu->Append(Minimal_About, wxT("&About\tF1"), wxT("Show about dialog
154 "));
155 fileMenu->Append(Serial_Open, wxT("&Open\tAlt-O"), wxT("Open serial port
156 "));
157 fileMenu->Append(Serial_Close, wxT("&Close\tAlt-C"), wxT("Close serial port
158 "));
159 editMenu->Append(Txt_Clear, wxT("Clea&r\tAlt-R"), wxT("Clear text"));
160 fileMenu->Append(Serial_Port, wxT("&Serial Port\tAlt-S"), wxT("Select
161 serial port"));
162 fileMenu->Append(Serial_Baud, wxT("&Baud Rate\tAlt-B"), wxT("Set baud rate
163 "));
164 fileMenu->Append(Serial_DataSize, wxT("&Data Size\tAlt-D"), wxT("Set data
165 size"));
166 fileMenu->Append(Serial_Parity, wxT("&Parity\tAlt-P"), wxT("Set parity"));
167 fileMenu->Append(Serial_StopBits, wxT("S&top Bits\tAlt-t"), wxT("Set stop
168 bits"));
169 fileMenu->Append(Minimal_Quit, wxT("E&xit\tAlt-X"), wxT("Quit this program
170 "));
171
172 // now append the freshly created menu to the menu bar...
173 wxMenuBar *menuBar = new wxMenuBar();
174 menuBar->Append(fileMenu, wxT("&File"));
175 menuBar->Append(editMenu, wxT("&Edit"));
176 menuBar->Append(helpMenu, wxT("&Help"));
177
178 // ... and attach this menu bar to the frame
179 SetMenuBar(menuBar);
180
181 #endif // wxUSE_MENUS
182
```

```

175 #if wxUSE_STATUSBAR
176     // create a status bar just for fun (by default with 1 pane only)
177     CreateStatusBar(2);
178     SetStatusText(wxT("Serial Communication"));
179 #endif // wxUSE_STATUSBAR
180     btnSend = new wxButton(this, Button_Send, wxT("Send"), wxPoint(5, 5), wxSize
181         (100, 25));
182     txtSend = new wxTextCtrl(this, Txt_Send, wxT("Hello!"), wxPoint(120,5), wxSize
183         (250,25));
184     //lblRx = new wxStaticText(this, ID_LBLRX, wxT("Rx:"), wxPoint(5, 75),
185     //wxSize(35, 25));
186     txtRx = new wxTextCtrl(this, Txt_Rx, wxT(""), wxPoint(5, 35), wxSize(365,
187         125), wxTE_MULTILINE);
188     chkRTS = new wxCheckBox(this, Chk_RTS, wxT("RTS"), wxPoint(5, 170),
189     wxDefaultSize);
190     chkDTR = new wxCheckBox(this, Chk_DTR, wxT("DTR"), wxPoint(55, 170),
191     wxDefaultSize);
192     chkCTS = new wxCheckBox(this, ID_CHKCTS, wxT("CTS"), wxPoint(155, 170),
193     wxDefaultSize);
194     chkDSR = new wxCheckBox(this, ID_CHKDSR, wxT("DSR"), wxPoint(205, 170),
195     wxDefaultSize);
196     chkRI = new wxCheckBox(this, ID_CHKRI, wxT("RI"), wxPoint(255, 170),
197     wxDefaultSize);
198     chkCD = new wxCheckBox(this, ID_CHKCD, wxT("CD"), wxPoint(305, 170),
199     wxDefaultSize);

200     Connect(Button_Send, wxEVT_COMMAND_BUTTON_CLICKED, wxCommandEventHandler(
201         MyFrame::OnSend));
202     Connect(Minimal_About, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
203         MyFrame::OnAbout));
204     Connect(Minimal_Quit, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
205         MyFrame::OnQuit));
206     Connect(Serial_Open, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
207         MyFrame::OnOpen));
208     Connect(Serial_Close, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
209         MyFrame::OnClose));

```

```
196     Connect(Serial_Port, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
197         MyFrame::SelPort));
198     Connect(Serial_Baud, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
199         MyFrame::SetBaud));
200     Connect(Serial_DataSize, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
201         MyFrame::SetDataSize));
202     Connect(Serial_Parity, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
203         MyFrame::SetParity));
204     Connect(Serial_StopBits, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
205         MyFrame::SetStopBits));
206     Connect(Timer1, wxEVT_TIMER, wxTimerEventHandler(MyFrame::OnTimer));
207     Connect.Txt_Clear, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
208         MyFrame::ClearText));
209     Connect(Chk_RTS, wxEVT_COMMAND_CHECKBOX_CLICKED, wxCommandEventHandler(
210         MyFrame::OnChkRTS));
211     Connect(Chk_DTR, wxEVT_COMMAND_CHECKBOX_CLICKED, wxCommandEventHandler(
212         MyFrame::OnChkDTR));
213     //Bind(wxEVT_MENU, &MyFrame::OnClose, this, Serial_Close);
214     m_timer.Start(250);
215     chkCTS->Disable();
216     chkDSR->Disable();
217     chkRI->Disable();
218     chkCD->Disable();
219 }
220
221 // event handlers
222 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
223 {
224     // true is to force the frame to close
225     Close(true);
226 }
227
228 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
229 {
230     wxMessageBox(wxString::Format(
```

```

224             wxT("Serial Communication! \n ")
225             wxT("Author: Yan Naing Aye \n ")
226             wxT("Web: https://github.com/yan9a/serial")
227         ),
228         wxT("About Serial Comm"),
229         wxOK | wxICON_INFORMATION,
230         this);
231     }
232
233 void MyFrame::OnOpen(wxCommandEvent& WXUNUSED(event))
234 {
235     if(com.Open()) txtRx->AppendText(wxString::Format(wxT("Error opening port %
236         s.\n"), com.GetPort()));
237     else txtRx->AppendText(wxString::Format(wxT("Port %s is opened.\n"), com.
238         GetPort()));
239 }
238
239 void MyFrame::OnClose(wxCommandEvent& WXUNUSED(event))
240 {
241     com.Close();
242     txtRx->AppendText(wxString::Format(wxT("Port %s is closed.\n"), com.GetPort
243         ()));
244 }
244
245 void MyFrame::SelPort(wxCommandEvent& WXUNUSED(event))
246 {
247     if (com.IsOpened()) {
248         txtRx->AppendText(wxString::Format(wxT("Close Port %s first.\n"), com.
249             GetPort()));
250     }
250     else {
251         wxString cdev=wxString::Format(wxT("%s"), com.GetPort());
252         wxString device = wxGetTextFromUser(wxT("Enter the port"), wxT("Set Port"
253             ), cdev);
254         string str = device.ToStdString();
255         if (str.length() > 0) {

```

```
255     com.SetPort(str);
256 }
257
258     txtRx->AppendText(wxString::Format(wxT("Port: %s\n"), com.GetPort()))
259     ;
260 }
261
262 void MyFrame::SetParity(wxCommandEvent& WXUNUSED(event))
263 {
264     if (com.IsOpened()) {
265         txtRx->AppendText(wxString::Format(wxT("Close Port %s first.\n"), com.
266         GetPort()));
267     }
268     else {
269         wxString cdev = wxString::Format(wxT("%c"), com.GetParity());
270         wxString parity = wxGetTextFromUser(wxT("Enter the parity ( N, E, O, M,
271         or S )"), wxT("Set Parity"), cdev);
272     #else
273         wxString parity = wxGetTextFromUser(wxT("Enter the parity ( N, E, or O )
274         ), wxT("Set Parity"), cdev);
275     #endif
276
277     string pstr = parity.ToString();
278     if (pstr.length() > 0) {
279         com.SetParity(pstr.at(0));
280     }
281     txtRx->AppendText(wxString::Format(wxT("Parity: %c\n"), com.GetParity()))
282     ;
283 }
284
285 void MyFrame::SetBaud(wxCommandEvent& WXUNUSED(event))
286 {
287     if (com.IsOpened()) {
```

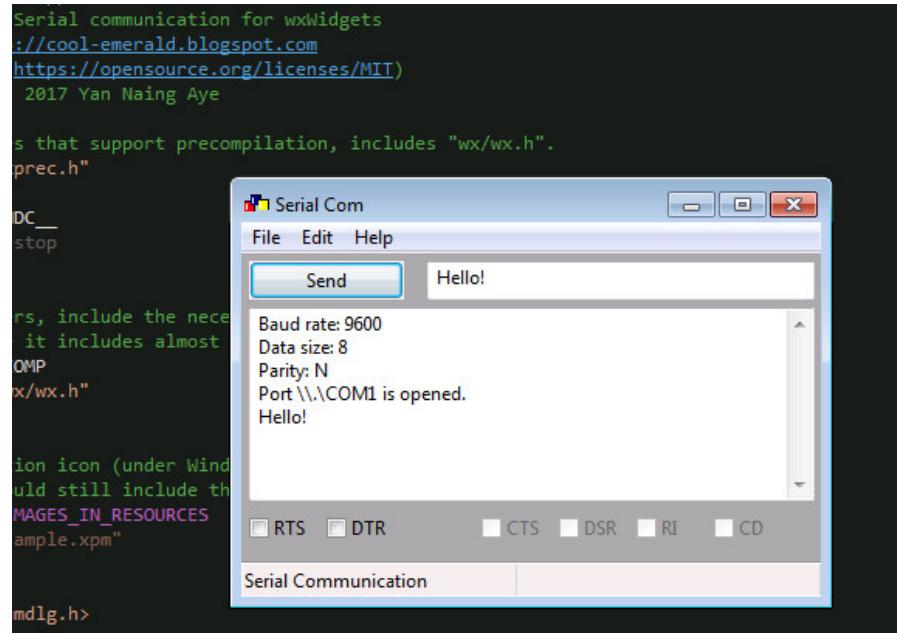
```
286     txtRx->AppendText(wxString::Format(wxT("Close port %s first.\n"), com.
287     GetPort())));
288 }
289 else {
290     long n = wxGetNumberFromUser(wxT("Enter the baud rate"), wxT("Baud rate")
291     , wxT("Set Baud Rate"), com.GetBaudRate(), 0, 1000000);
292     if (n >= 0) {
293         com.SetBaudRate(n);
294     }
295     txtRx->AppendText(wxString::Format(wxT("Baud rate: %ld\n"), com.
296     GetBaudRate())));
297 }
298
299 void MyFrame::SetDataSize(wxCommandEvent& WXUNUSED(event))
300 {
301     if (com.IsOpened()) {
302         txtRx->AppendText(wxString::Format(wxT("Close port %s first.\n"), com.
303         GetPort()));
304     }
305     else {
306         long n = wxGetNumberFromUser(wxT("Enter the data size"), wxT("Data Size")
307         , wxT("Set Data Size"), com.GetDataSize(), 5, 8);
308         if (n >= 0) {
309             com.SetDataSize(n);
310         }
311         txtRx->AppendText(wxString::Format(wxT("Data size: %ld\n"), com.
312         GetDataSize())));
313     }
314 }
```

```
315 }
316 else {
317     long n = wxGetNumberFromUser(wxT("Enter the number of stop bits"), wxT("Data Size"),
318                                 wxT("Set stop bits"), long(com.GetStopBits()), 1, 2);
319     if (n > 0) {
320         com.SetStopBits(float(n));
321     }
322     txtRx->AppendText(wxString::Format(wxT("Stop bits: %ld\n"), long(com.
323     GetStopBits())));
324 }
325 void MyFrame::OnSend(wxCommandEvent& WXUNUSED(event))
326 {
327     wxString str = txtSend->GetValue();
328     wxCharBuffer buffer = str.ToUTF8();
329     if (com.Write(buffer.data())) {
330         txtRx->AppendText(str);
331     }
332     else {
333         txtRx->AppendText(wxT("Write error.\n"));
334     }
335 }
336
337 void MyFrame::OnTimer(wxTimerEvent& WXUNUSED(event))
338 {
339     char ch; bool r;
340     do {ch = com.ReadChar(r); if (r) ProcessChar(ch);} while (r);
341     UpdateCommStatus();
342 }
343
344 void MyFrame::ProcessChar(char ch)
345 {
346     txtRx->AppendText(wxString::Format(wxT("%c"), ch));
347 }
```

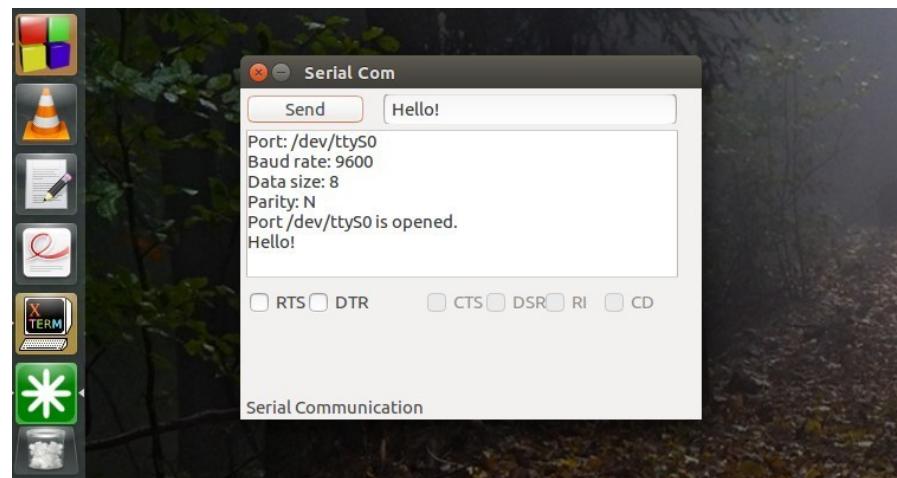
```

349 void MyFrame::ClearText(wxCommandEvent& WXUNUSED(event))
350 {
351     txtRx->Clear();
352 }
353
354 void MyFrame::OnChkRTS(wxCommandEvent& WXUNUSED(event))
355 {
356     if (!com.SetRTS(chkRTS->IsChecked())) {
357         txtRx->AppendText(wxT("RTS error.\n"));
358     }
359 }
360
361 void MyFrame::OnChkDTR(wxCommandEvent& WXUNUSED(event))
362 {
363     if (!com.SetDTR(chkDTR->IsChecked())) {
364         txtRx->AppendText(wxT("DTR error.\n"));
365     }
366 }
367
368 void MyFrame::UpdateCommStatus()
369 {
370     bool s;
371     bool v;
372     v = com.GetCTS(s);
373     if (s) chkCTS->SetValue(v);
374     v = com.GetDSR(s);
375     if (s) chkDSR->SetValue(v);
376     v = com.GetRI(s);
377     if (s) chkRI->SetValue(v);
378     v = com.GetCD(s);
379     if (s) chkCD->SetValue(v);
380 }
```

Windows ပေါ်မှာ Visual Studio 2017 နဲ့ wxserial.cpp ကို run လိုက်တဲ့ အခါ ပုံ ၇.၁၈ မှာ ပြထားသလို တွေ့နိုင် ပါတယ်။



ပုံ ၇.၁၉: Serial class သုံးထားသော wxWidgets GUI application ကို Visual Studio 2017 ဖွင့် run ခြင်း။



ပုံ ၇.၂၀: Serial class သုံးထားသော wxWidgets GUI application ကို Ubuntu ပေါ်ပွင့် run ခြင်း။

အဲဒီ wxserial.cpp program ကိုပဲ ဘာမှ မပြင်ပဲ Debian အခြေဖြူ Linux တွေပေါ်မှာ အောက်ကအတိုင်း built လုပ်၊ run လုပ်လိုရပါတယ်။ ပရိဂရမ် ရဲ့ GUI ကို ပုံ ၇.၁၉ မှာ တွေ့နိုင်ပါတယ်။

```
g++ wxserial.cpp `wx-config --cxxflags --libs` -o wxserial -DNDEBUG
sudo ./wxserial
```

## ၇.၆ Byte Stuffing

Device တစ်ခု ကနေ တစ်ခု ကို data တွေ ပို့တဲ့ လက်ခံ တဲ့ အခါမှာ frame တွေ တည်ဆောက် ပြီး အပိုင်းလိုက် message အနေနဲ့ ပို့တာ က ပိုပြီး ကောင်းမွန် အဆင်ပြေ လျှော့ ပါတယ်။ ရှိုးရှင်းတဲ့ byte stuffing မူကဲ တစ်ခု ကို နမူနာ အနေနဲ့ ရွှေ့ပြီး data bytes တွေကို frame ဆောက် ပြီး ပိုဖို့ နဲ့ လက်ခံဖို့ အတွက် လုပ်ပါ မယ်။ Frame ရဲ့ အစ နဲ့ အဆုံးကို သတ်မှတ်ဖို့ အတွက် control characters တွေ အဖြစ် 0x02 နဲ့ 0x03 ကို start of text (STX) ရယ်၊ end of text (ETX) ရယ် လို့ ထားလိုက် ပါမယ်။ လက်ခံရရှိတဲ့ ဒေတာ မှာ အမှား ပါမပါ စစ်ဖို့ အတွက် data bytes တွေရဲ့ exclusive-or တန်ဖိုးကို ETX နောက်မှာ အဆုံးသတ် checksum အနေနဲ့ ထည့်လိုက်ပါမယ်။ ပိုကောင်းတဲ့ error detection အနေနဲ့ တော့ CRC ကို သုံးလို့ ရပါတယ် [CE09]။ ဥပမာ data byte နှစ်လုံး

```
0x30 0x31
```

ကို ပိုဖို့ ဆိုရင်၊ ပိုရမယ့် frame က

```
0x02 0x30 0x31 0x03 0x01
```

ဖြစ်ပါတယ်။ ပိုချင်တဲ့ data bytes တွေ မစခင် မှာ 0x02 ကို STX အနေနဲ့ ထည့်ထား လိုက်တဲ့ အတွက် လက်ခံ မယ့် device က STX ကို ရတာ နဲ့ ဒေတာ တွေ ရဲ့ အစ မှန်း သိနိုင် သွားပါတယ်။ အဲဒီ လိုပဲ ဒေတာ တွေရဲ့ အဆုံးမှာ 0x03 က ETX အနေနဲ့ ထည့်ထား တဲ့ အတွက်၊ လက်ခံ တဲ့ ဘက်က ETX ကို ရတာ နဲ့ ဒေတာ frame ရဲ့ အဆုံး မှန်း သိပြီး processing တွေ လုပ်နိုင် ပါတယ်။ ပြီးတဲ့ ရရှိတဲ့ data byte တွေက မှန်ကန်မှု ရှိမရှိ ကို သူတို့ရဲ့ exclusive-or (0x02 ⊕ 0x03) ဖြစ်တဲ့ 0x01 ကို တွက်ကြည့် ပြီး checksum အနေနဲ့ နောက်ဆုံးက နေ ထည့်ထား တဲ့ တန်ဖိုး နဲ့ ပြန် တိုက်ကြည့် နိုင် ပါတယ်။

အကယ်၍ ပို့ရမယ့် ဒေတာမှာ control characters တွေ အနေနဲ့ သုံးထားတဲ့ 0x02 တို့၊ 0x03 တို့ ပါလာရင် ဘယ်လို လုပ်မလဲ လို့ မေးစရာရှိ ပါတယ်။ အဲဒီ အတွက် နောက်ထပ် control character တစ်ခု လိုလာ ဖြေး၊ 0x10 ကို Data Link Escape (DLE) အနေနဲ့ သတ်မှတ် ထား ပါတယ်။

နောက်ထပ် ဥပမာ အနေနဲ့ data byte ငါးလုံး ဖြစ်တဲ့

```
0x30 0x02 0x65 0x10 0x03
```

အတွက် frame တစ်ခု ဆောက်ကြည့် ပါမယ်။ ဒေတာထဲမှာ STX တို့၊ ETX တို့၊ DLE တို့ ကိုတွေ့တိုင်း control character မဟုတ်ကြောင်း သိအောင် ရှေ့မှာ DLE တစ်လုံးကို အပို ထည့်ပေး မှာပါ။ ဒါဆို ပို့ရမယ့် frame ၏

```
0x02 0x30 0x10 0x02 0x65 0x10 0x10 0x10 0x03 0x03 0x44
```

ဖြစ်ပါတယ်။ လက်ခံ ရရှိတဲ့ device ၏ DLE ကို တွေ့တိုင်း သူ့နောက် ကပ်ရပ် byte ကိုပဲ ဒေတာ အနေနဲ့ ယူဆ ပါတယ်။ Byte stuffing ကို CRC16 နဲ့ အသုံးပြု ဖြေး ဒေတာ တွေပို့တဲ့ လက်ခံတဲ့ နမူနာ C++ ကုဒ် frame.h ကို နောက်ဆက်တဲ့ A စာရင်း ၁.၄ မှာ တွေ့နိုင်ပါတယ်။

## အကိုးအကားများ

- [CE09] Cool-Emerald. “CRC Calculation in VB and C”. In: (2009). url: <http://coolemerald.blogspot.com/2009/09/crc-calculation-in-vb-and-c.html>.
- [Den95] Allen Denver. Serial Communications. 1995. url: <https://msdn.microsoft.com/en-us/library/ff802693.aspx>.
- [Eli17a] Elinux. RPi Serial Connection. 2017. url: [https://elinux.org/RPi\\_Serial\\_Connection](https://elinux.org/RPi_Serial_Connection).
- [Lab17] Silicon Labs. AN197: Serial Communications Guide for the CP210x. 2017. url: <https://www.silabs.com/documents/public/application-notes/an197.pdf>.

[Wik16] Wikibooks. Serial Programming/termios. 2016. url: [https://en.wikibooks.org/wiki/Serial\\_Programming/termios](https://en.wikibooks.org/wiki/Serial_Programming/termios).

## အခန်း ၈

# Database

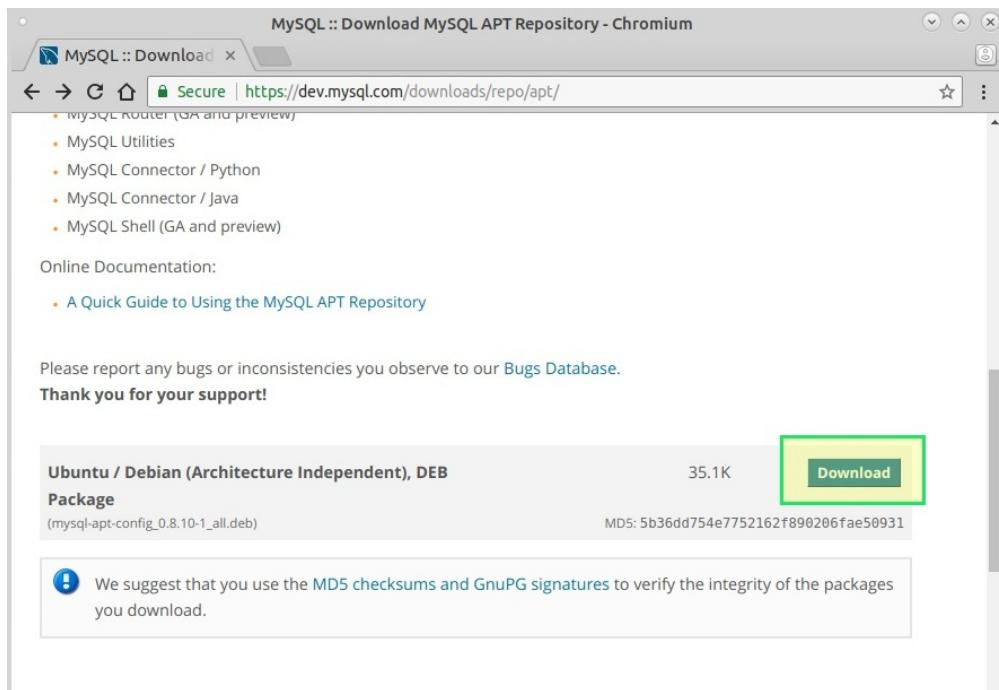
Database server တွကို ဆက်သွယ် အသုံးပြု တဲ့ C++ application တွေ ရေးသား တဲ့ အကြောင်း ဈွေးနွေး ပါမယ်။ MySQL (မိုင် အက်စ် ကျူး အယ်) က Oracle ရဲ့ open source ဖြစ်တဲ့ database management system တစ်ခု ဖြစ် ပါတယ်။ သူက freeရနိုင်ပြီး robust ဖြစ်ရုံး မက ဘဲ platforms တော်တော် များများ ပေါ်မှာ အလုပ်လုပ် ပါတယ်။ Odroid လို့ 32 bit armv7 architecture SBC လေးတွေ ပေါ်မှာ တောင် source ကနေ build လုပ်ပြီး run နိုင်ပြီး functionality တွေ စုံသလို့ user interface မျိုးစုံ ရှိတာ ဖို့ နမူနာ အနေနဲ့ MySQL database server ကို ဆက်သွယ် အသုံးပြု ပါမယ်။

MySQL server ကို traditional အတိုင်း relational database ပုံစံ နဲ့ သုံးလို့ ရသလို့ schema ကို အသေ သတ်မှတ် စရာ မလိုတဲ့ NoSQL လိုလည်း ခေါ်ကြတဲ့ document store အနေနဲ့ လည်း သုံးလို့ ရပါတယ်။ Relational database မှာ သိမ်းမယ့် table ရဲ့ column အားလုံး ကြိုးလိုး schema ကြိုးသတ်မှတ် ဖို့ လိုပေမယ့်၊ document store ကတော့ schema flexible ဖြစ်ပြီး JSON object နဲ့ ကိုယ်စားပြု ဖော်ပြ ပါတယ်။ ဒီ နေရာမှာ relational model ရဲ့ power နဲ့ document store ရဲ့ flexibility ကို ပေါင်းစပ် အသုံးပြု လို့ ရအောင် X DevAPI ကို သုံးပြီး application တွေ ပြုလုပ် တဲ့ အကြောင်း ဈွေးနွေး ပါမယ်။

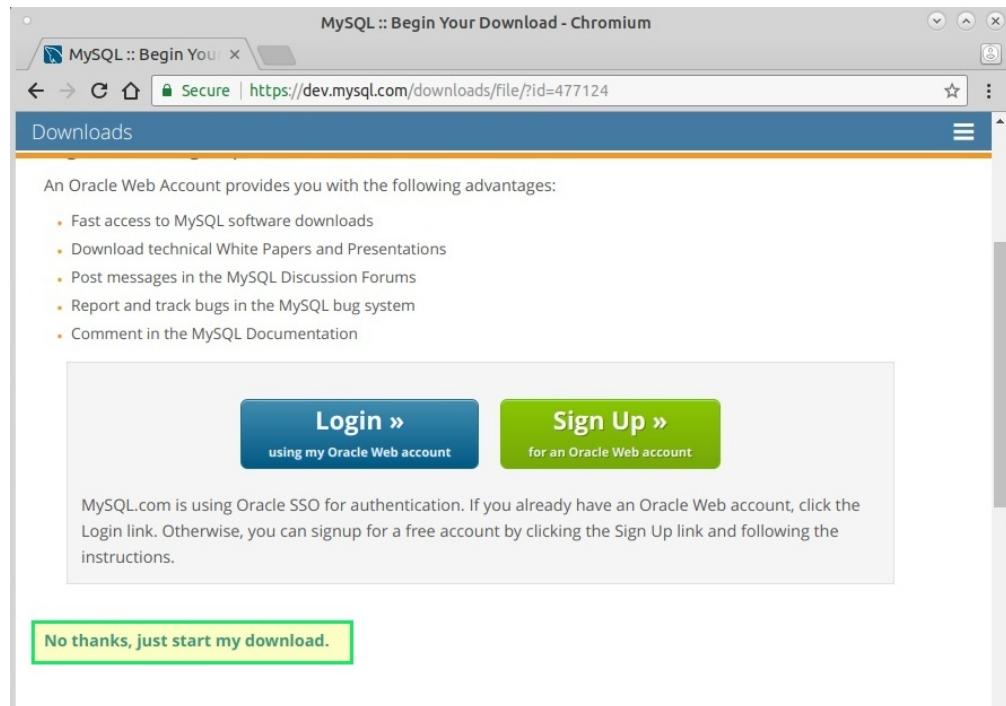
Linux Virtual Machine တစ်ခု လုပ်ပြီး MySQL server ကို အဲဒီ ပေါ်မှာ run နိုင်းထား လို့ ရပါတယ်။ နမူနာ အနေနဲ့ Ubuntu Linux ကို MySQL server အတွက် သုံးပြီး၊ Raspberry Pi ဘက် မှာတော့ အဲဒီ server ကို ဆက်သွယ် အသုံးပြုမယ့် client C++ application ကိုပဲ ဖန်တီး ပြီး၊ database ကို ကိုင်တွယ်အသုံးပြု မှာပါ။

## ၈.၁ တပ်ဆင်ခြင်း

X protocol နဲ့ ဆက်သွယ် အသုံးပြု မှု ကို အထောက် အပံ့ ပေးတဲ့ X Plugin ပါတဲ့ MySQL server 8.0 ကို Ubuntu ပေါ်မှာ တပ်ဆင် ဖို့ အတွက် သူရဲ့ Download MySQL APT Repository (<https://dev.mysql.com/downloads/repo/apt/>) စာမျက်နှာ ကို သွားပြီး Ubuntu / Debian (Architecture Independent), DEB Package ဆိုတဲ့ apt config package ကို download လုပ်လိုက် ပါမယ် [Ora18a; Bou17a]။ ဒီ နမူနာ မှာတော့ လက်ရှိ နောက်ဆုံး ထွက် mysql-apt-config\_0.8.10-1\_all.deb ကို သုံးပါမယ် (ပုံ ၈.၁)။ Oracle Web account ဖန်တီး ချင်ရင် ဖန်တီး နိုင် ပြီး No ကို နှိပ်ပြီး ကျော်သွား ရင်လည်း ရပါတယ် (ပုံ ၈.၂)။



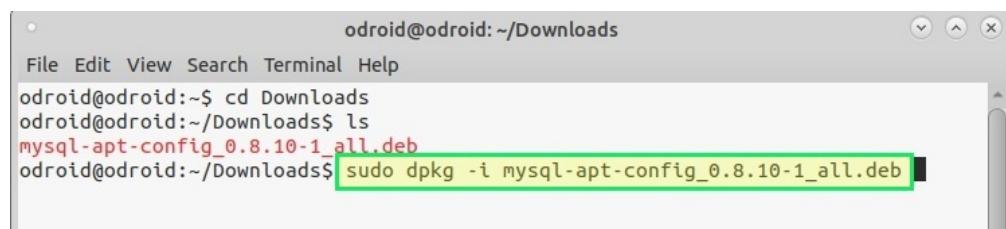
ပုံ ၈.၁: နောက်ဆုံးထွက် MySQL apt config package ကို ရယူခြင်း။



ပုံ ၈.၂: Oracle Web account မဖန်တီးရန် ရွေးချယ်ခြင်း နှင့် download ကို စတင်ခြင်း။

ပြီးတဲ့ အခါ အဲဒီ directory မှာ ပုံ ၈.၃ မှာပြ ထားတဲ့ အတိုင်း dpkg command ကို သုံးပြီး သူရဲ့ apt configure package ကို တပ်ဆင် ပါမယ်။

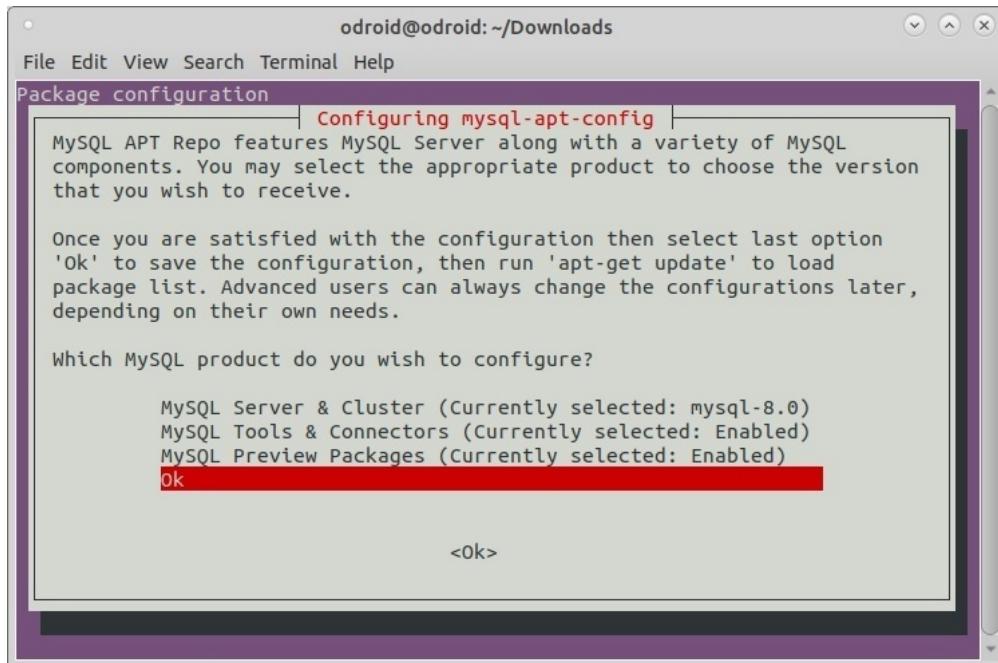
```
$ sudo dpkg -i mysql-apt-config_0.8.10-1_all.deb
```



ပုံ ၈.၃: Apt config package ကို တပ်ဆင်ခြင်း။

စိတ်ကြိုက် configure လုပ်ပြီး OK ကို နိုင်လိုက် တဲ့ အခါ တပ်ဆင်မှု လုပ်ဆောင်ပြီး command prompt ကို ပြန်ရောက် သွား ပါလိမ့် မယ်။ ဒါ နမူနာ မှာ တော့ mysql ရဲ့ API အသစ် ဖြစ်တဲ့ X DevAPI

ကို သုံးဖို့ ရည်ရွယ် တာမူ့ ပုံ ၈.၄ မှာ ပြထား သလို mysql-8.0 ကို ရွေးချယ် လိုက် ပါတယ်။ Tools နဲ့ preview packages တွေကို လည်း enabled လုပ် လိုက် ပါတယ်။



ပုံ ၈.၄: mysql-8.0 ကို ရွေးချယ်ခြင်း။

MySQL server ကို တပ်ဆင် ဖို့ အတွက် အောက်က command ကို သုံးနိုင် ပါတယ်။ တပ်ဆင် နေချိန် မှာ root password ကို သတ်မှတ် ဖို့ တောင်းရင် ထည့် ပေးနိုင် ပါတယ်။

```
$ sudo apt update
$ sudo apt install mysql-server
```

MySQL client နဲ့ စီမံခန့်ခွဲမှု အတွက် MySQL workbench တို့ကို တပ်ဆင် ဖို့ အတွက် လည်း အောက်က command ကို သုံးလိုက် ပါမယ်။

```
$ sudo apt install mysql-client mysql-workbench
```

### ၈.၁ Major Release Version ကို ရွေးချယ်ခြင်း

MySQL ကို တပ်ဆင်ပြီး တဲ့ အခါ တွေား major release version ကို ပြန် ပြောင်း ချင်ရင် ပြောင်းလို ရပြီး၊ အဲဒီ အတွက် အောက်ပါ အတိုင်း လုပ်ဆောင် နှင့် ပါတယ်။

```
$ sudo dpkg-reconfigure mysql-apt-config
$ sudo apt-get update
```

### ၈.၂ Source မှ တပ်ဆင်ခြင်း

#### ၈.၂.၁ လိုအပ်သည့်အရာများ

တစ်ချို့ platform ဥပမာ armhf တွေ အတွက် MySQL Server 8.0 က install လုပ်စရာ မရှိပဲ support မလုပ် ပါဘူး။ အဲဒီ ဆိုရင် ကိုယ့်ဟာ ကိုယ် MySQL ကို source ကနေ build လုပ်ပြီး၊ install လုပ်ဖို့ လို ပါတယ်။ Source ကနေ build လုပ်မယ် ဆိုရင် အောက်ပါ development tools တွေ စက်ထဲ မှာ ရှိနေဖို့ လိုအပ် ပါတယ်။

1. CMAKE
2. GCC 4.8 or higher
3. Boost C++ libraries
4. ncurses library
5. Git and bison (အကယ်၍ MySQL on GitHub source tree ကနေ build လုပ်မယ် ဆိုရင် )

GCC, CMAKE, Git, ncurses နဲ့ bison တို့ ကို အောက်ပါ အတိုင်း တပ်ဆင် ပါမယ်။

```
$ sudo apt update
$ sudo apt install build-essential
$ sudo apt install cmake
$ sudo apt install git
$ sudo apt install libncurses5-dev
$ sudo apt install bison
```

## ၈.J. SOURCE မှ တပ်ဆင်ခြင်း

JJO

Boost ကို တပ်ဆင် ဖို့ အတွက် အောက်ပါ link ကို သွားပြီး၊ Version 1.66.0 ဖြစ်တဲ့ boost\_1\_66\_0.tar.bz2 ကို ဒေါင်းလုပ် လုပ်လိုက် ပါမယ်။

<https://dl.bintray.com/boostorg/release/1.66.0/source/>

ပြီးတဲ့ အခါ /usr/local/ မှာ တပ်ဆင် ဖို့ အောက်ပါ command တွေကို သုံးနိုင် ပါတယ်။

```
$ cd /usr/local  
$ sudo tar --bzip2 -xf ~/Downloads/boost_1_66_0.tar.bz2
```

အဲဒီ လို Boost header တွေ ရရှိပြီး တဲ့နောက် MySQL ကို build လုပ်တဲ့ အချိန် CMAKE မှာ

```
-DWITH_BOOST=/usr/local/boost_1_66_0
```

ဆိုတဲ့ option ထည့် ပေးနိုင် ပါတယ်။

CMAKE မှာ SSL libraries တွေ မတွေ့ တဲ့ ပြဿနာ တက်နိုင် တာမို့ အောက်က အတိုင်း SSL ကို တပ်ဆင် ထားနိုင် ပါတယ်။

```
$ sudo apt install libssl-dev
```

## ၈.J.J Preconfiguration setup

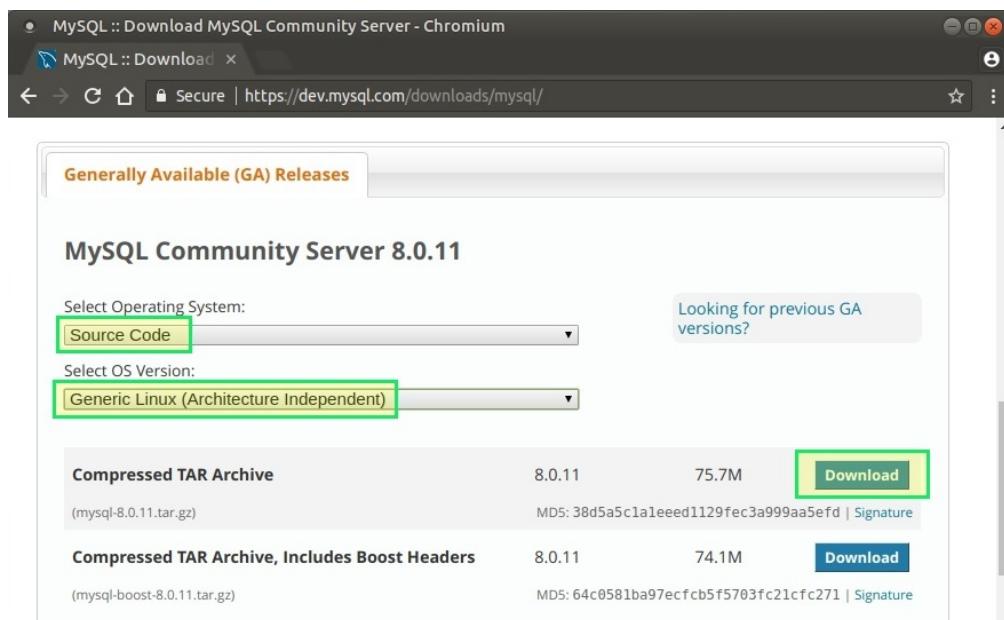
MySQL ကို build မလုပ် ခင် preconfiguration setup အနေ နဲ့ အောက်ပါ တို့ကို လုပ်ဆောင် ဖို့ လိုပါတယ် [Ora18b]။ အကယ်၍ mysql လို့ လည်း ခေါ်တဲ့ MySQL Server ကို run ဖို့ အတွက် စနစ် ထဲမှာ user နဲ့ group မရှိ သေးရင်၊ သူတို့ ကို ဖန်တီး ဖို့လို ပါတယ်။ mysql ဆိုတဲ့ user နဲ့ group ကို အောက်ပါ အတိုင်း ဖန်တီး နိုင် ပါတယ်။ mysql ဆိုတဲ့ နာမည် တွေကို မသုံးချင် ရင် တစ်ခြား နာမည် တွေကို သုံးလို လည်း ရ ပါတယ်။ တစ်ချို့ စနစ် တွေရဲ့ syntax က groupadd နဲ့ useradd အစား addgroup နဲ့ adduser လည်း ဖြစ်နိုင် ပါတယ်။

```
$ sudo groupadd mysql  
$ sudo useradd -r -g mysql -s /bin/false mysql
```

User ၊ ownership အတွက် ပဲ လိုတာ ဖြစ်ပြီး၊ login အတွက် မလို အပ် တာမို့ useradd command

မှာ -r နဲ့ -s /bin/false ဆိတဲ့ option တွေကို သုံးပြီး login permissions တွေ မရှိတဲ့ user ကို ဖန်တီးထားပါတယ်။ အဲဒီ options တွေကို အထောက် အပုံ မပေးတဲ့ စနစ် တွေပေါ်မှာ ဆိုရင် တော့ ချုန်လှပ်ခဲ့လို့ ရပါတယ်။

အဲဒီ နောက် [dev.mysql.com/downloads/mysql/](https://dev.mysql.com/downloads/mysql/) ကို သွားပြီး Select Operating System ဆိတဲ့ drop down box မှာ Source Code ကို ရွေး၊ Select OS Version ဆိတဲ့ နေရာ မှာ Generic Linux (Architecture Independent) ကို ရွေးပြီး တဲ့ နောက် Compressed TAR Archive ကို ပုံ ၈.၅ မှာ ပြထားသလို download လုပ်လိုက် ပါမယ်။ Oracle Account လုပ်ဖို့ မေးလာ တဲ့ အခါ account sign up လုပ်ချင်ရင် လုပ် လို့ ရပြီး၊ No thanks, just start my download. ကို နှိပ်ပြီး ရွှေဆက် သွားလို့ လည်းရပါတယ်။



ပုံ ၈.၅: MySQL Source ကို ရယူခြင်း။

## ၈.၂ Installation

ရလာ တဲ့ source ဖုန် mysql-8.0.11.tar.gz ကို extract လုပ်၊ ပြီးတဲ့ အခါ build လုပ်ပြီး install လုပ်ပါမယ်။

```
$ tar zxvf ~/Downloads/mysql-8.0.11.tar.gz
$ cd mysql-8.0.11
```

```
$ mkdir bld
$ cd bld
$ cmake .. -DWITH_BOOST=/usr/local/boost_1_66_0
$ make
$ sudo make install
```

### ၈.၂.၄ Postinstallation setup

MySQL ကို တပ်ဆင် ပြီးတဲ့ အခါ postinstallation setup အနေနဲ့ mysql system database ထဲက tables တွေ ပါ ပါတဲ့ data directory ကို initialize လုပ်ဖို့ လို ပါတယ်။ အဲဒီ အတွက် MySQL ကို တပ်ဆင် ထားတဲ့ directory ကို သွားပါ မယ်။ အဲဒီ မှာ ဖိုင်တွေ၊ အခန်းခွဲ တွေ အများကြီး တွေ့နိုင် ပြီး bin ဆိုတဲ့ အခန်း ထဲမှာ server နဲ့ client utility တွေ ရှိပါ တယ်။

```
$ cd /usr/local/mysql
```

Import/export operations တွေ ကို directory တစ်ခု မှာ ကန်သတ် ပေးတဲ့ secure\_file\_priv ဆိုတဲ့ system variable အတွက် directory တစ်ခု ကို ဖန်တီး ပါမယ်။ ပြီးတဲ့ အခါ ownership, group နဲ့ permissions တွေကို သတ်မှတ် မှာ ဖြစ်ပါ တယ်။

```
$ sudo mkdir mysql-files
$ sudo chown mysql:mysql mysql-files
$ sudo chmod 750 mysql-files
```

User တွေ MySQL server ကို ဆက်သွယ် အသုံးပြု ဖို့ ခွင့်ပြုချက် တွေကို သတ်မှတ် ပေးတဲ့ initial MySQL grant tables တွေ ပါတဲ့ mysql database အတွက် data directory ကို initialize လုပ် ပါမယ်။

```
$ sudo bin/mysqld --initialize --user=mysql
```

ဒါ အဆင့် ပြီးတဲ့ အခါ root user အတွက် temporary password ထုတ်ပေး တာကို ပုံ ၈.၆ မှာ ပြထား သလို တွေ့ရ မှာ ဖြစ်ပြီး ရေးမှတ် ထားနိုင် ပါတယ်။

```
odroid@odroid:/usr/local/mysql$ sudo bin/mysqlld --initialize --user=mysql
2018-07-16T02:20:31.664435Z 0 [System] [MY-013169] [Server] /usr/local/mysql/bi
n/mysqlld (mysqlld 8.0.11) initializing of server in progress as process 28762
2018-07-16T02:20:53.380425Z 5 [Note] [MY-010454] [Server] A temporary password
is generated for root@localhost: -Yp_Ug=9.z<B
2018-07-16T02:20:58.322305Z 0 [System] [MY-013170] [Server] /usr/local/mysql/bi
n/mysqlld (mysqlld 8.0.11) initializing of server has completed
odroid@odroid:/usr/local/mysql$
```

ပုံ ၈.၆: MySQL root user အတွက် ယာယိ password ထုတ်ပေးခြင်း။

အကယ်၍ secure connections တွေ အတွက် အလို အလျောက် အထောက် အပံ့ ကို သုံးချင် ရင် mysql\_ssl\_rsa\_setup ဆိုတဲ့ utility ကို သုံးပြီး default SSL နဲ့ RSA ဖိုင် တွေကို ဖန်တီး နိုင် ပါတယ်။

```
$ sudo bin/mysql_ssl_rsa_setup
```

ပြီးတဲ့ အခါ MySQL server ကို start လုပ် ဖို့ အတွက် mysqld\_safe ကို အောက်ပါ အတိုင်း သုံးနိုင် ပါတယ်။

```
$ sudo bin/mysqld_safe --user=mysql &
# Next command is optional
$ sudo cp support-files/mysql.server /etc/init.d/mysql.server
```

MySQL server ကို run တဲ့ အခါ root မဟုတ် တဲ့ account ဖြစ်ဖို့ အရေးကြီး ပါတယ်။ အဲဒီ အတွက် mysqld\_safe ကို root နဲ့ run ပြီး၊ –user ဆိုတဲ့ option ကို သုံးနိုင် ပါတယ်။ အဲလို မဟုတ်ရင် mysql အနေနဲ့ logged in လုပ်ပြီး မှ run ရမှာ ပါ။

MySQL server ရဲ့ temporary root password က expired ဖြစ်နေ မှာမို့ အပိုင်း ၈.၄ မှာ ခွေးနွေး ထား သလို mysql\_secure\_installation ကို သုံးပြီး configure ပြန်လုပ် နိုင် ပါတယ်။

```
$ sudo bin/mysql_secure_installation
```

## ၈.၂၅ Startup

Linux စနစ် တွေ အတွက် binary ဒါမ္မ မဟုတ် source ကနေ build လုပ်ရ တဲ့ MySQL distributions တွေမှာ server ကို mysqld\_safe သုံးပြီး စတင် ပေးတဲ့ mysql.server ဆိုတဲ့ script ပါ ပါတယ်။ အဲဒီ mysql.server ကိုသုံးပြီး server ကို manual စဖို့ ရပိုဖို အတွက် အောက်က command တွေကို သုံးနိုင်

## ၈.၂. SOURCE မှ တပ်ဆင်ခြင်း

ပါတယ်။

```
$ cd /usr/local/mysql
$ sudo support-files/mysql.server start
$ sudo support-files/mysql.server stop
```

MySQL ကို အလို အလျောက် စဖို့နဲ့ ရပို့ အတွက် ဆိုရင် တော့ mysql.server ကို mysql ဆိုတဲ့ နာမည် နဲ့ /etc/init.d directory ထဲကို ကူးထည့် ပြီး၊ executable ဖြစ်အောင် လုပ်ဖို့လိုပါ တယ်။

```
$ cd /usr/local/mysql
$ sudo cp support-files/mysql.server /etc/init.d/mysql
$ sudo chmod +x /etc/init.d/mysql
```

ပြီးတဲ့ အခါ update-rc.d ကို သုံးပြီး system startup မှာ run ဖို့ အတွက် activate လုပ်ပါ မယ်။

```
$ sudo update-rc.d mysql defaults
```

တခို့ Linux တွေ အတွက် ဆိုရင် တော့ update-rc.d ကို မသုံးရင် /etc/rc.local ထဲမှာ အောက်က command ကို ဖြည့်ပေး နိုင် ပါတယ်။

```
/bin/sh -c 'cd /usr/local/mysql; ./bin/mysqld_safe --user=mysql &'
```

mysql.server က သူအတွက် options တွေကို option ဖိုင် ထဲက [mysql.server] နဲ့ [mysqld] ဆိုတဲ့ section တွေက နေ ဖတ် ပါတယ်။ Option တွေကို ထပ်ဖြည့် ချင်ရင် /etc/my.cnf ဆိုတဲ့ ဖိုင် ထဲမှာ ထပ် ဖြည့်နိုင်၊ ပြင်နိုင် ပါတယ်။ အဲဒီ ဖိုင် ရဲ့ ပုံစံ နမူနာ ကို အောက်မှာ တွေ့နိုင် ပါတယ်။

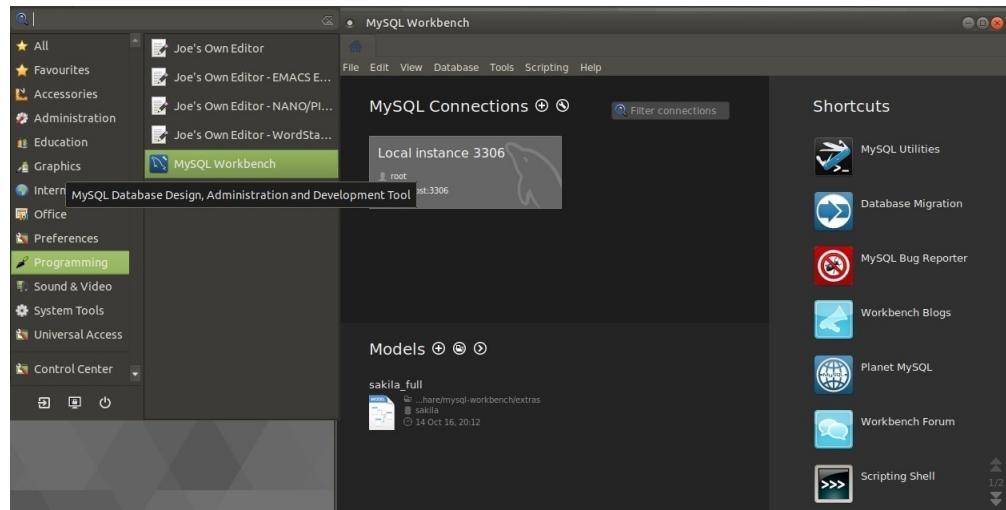
```
[mysqld]
datadir=/usr/local/mysql/var
socket=/var/tmp/mysql.sock
port=3306
user=mysql

[mysql.server]
basedir=/usr/local/mysql
```

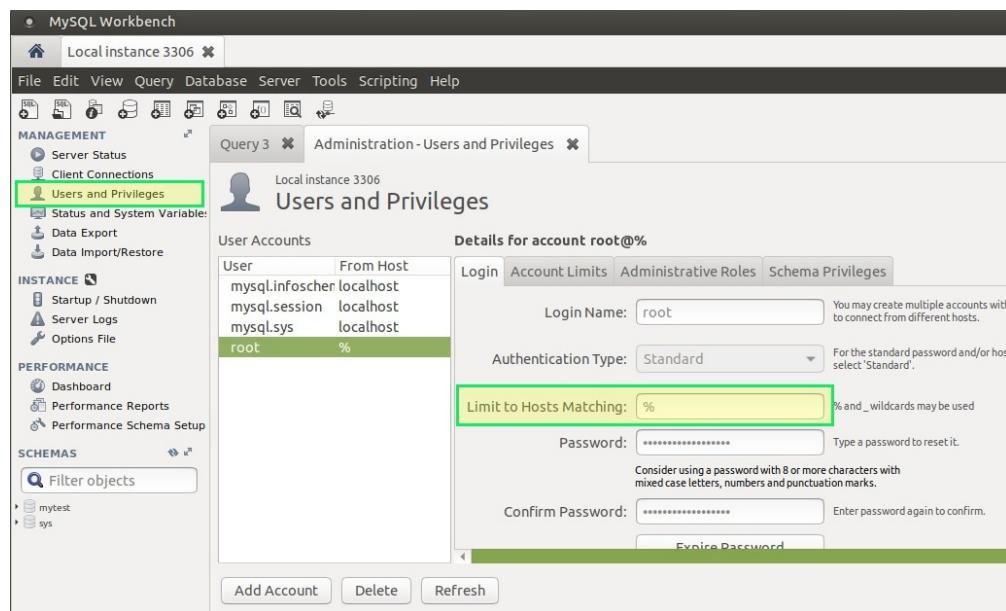
၂၂၇

အခန်း ၈. DATABASE

ပုံ ၈.၇ မှာ ပြထားသလို MySQL workbench ကို ဖွင့်ပြီ၊ root ကို တစ်ခြား စက်တွေ ကပါ ဆက်သွယ် အသုံးပြု လို ရအောင် Limit to Hosts Matching ကို ပုံ ၈.၈ အတိုင်း % ဖြစ်အောင် ပြပြင် နိုင်ပါတယ်။



ပုံ ၈.၇: MySQL Workbench ကိုသုံးခြင်း။



ပုံ ၈.၈: Host ကန်သတ်မှတ် ပြပြင်ခြင်း။

## ၈.၃ Root Password ကို reset လုပ်ခြင်း

တပ်ဆင် ပြီးတဲ့ အခါ အကြောင်း အမျိုးမျိုး ကြောင့် root password ကို reset လုပ်ချင်ရင် mysql service ကို အရင် ရပ်လိုက် ဖို့လို ပါမယ်။

```
$ sudo service mysql status
$ sudo service mysql stop
```

ပြီးရင် သူရဲ့ root password ကို reset လုပ်ဖို့ mysql safe daemon ကို --skip-grant-tables option နဲ့ စလိုက် ပါ မယ်။ အဲဒါ က ဘယ်သူ ကို မဆို password မလိုပဲ ဝင့်ခွင့် ပေးပြီး၊ ခွင့်ပြုချက် အပြည့် ပေးပါတယ်။ အဲဒီ လို server ကို --skip-grant-tables option နဲ့ စတာ ဟာ လုပ်မှုမှ မရှိ တာကြောင့် remote connection တွေကို တားဆိုဖို့ --skip-networking က အလို အလျောက် enable ဖြစ်သွား ပါလိမ့်မယ်။

```
$ sudo mysqld_safe --skip-grant-tables &
```

အကယ်၍ mysqld\_safe Directory '/var/run/mysqld' for UNIX socket file don't exists ဆိုတဲ့ error တက်နေရင် တော့ အောက်က အတိုင်း လုပ်ဆောင် နိုင် ပါတယ်။

```
$ sudo mkdir -p /var/run/mysqld
$ sudo chown mysql:mysql /var/run/mysqld
```

ပြီးတဲ့အခါ ခုနက command

```
$ sudo mysqld_safe --skip-grant-tables &
```

ကို ပြန် run လိုက်တဲ့ အခါ starting mysqld daemon လို ပြပြီး cursor လေး ပေါ်လာ တဲ့ အခါ mysql ကို root user အနေနဲ့ password မလိုပဲ ဝင်နိုင် ပါမယ်။

```
mysql -u root
```

အဲဒီ နောက် ပေါ်လာ တဲ့ mysql prompt မှာ semicolon တွေနဲ့ အမြဲ အဆုံးသတ် တဲ့ sql command တွေကို ထပ်ထည့် ပါမယ်။ အဲဒီက my-new-password နေရာ မှာ ကိုယ်သုံးချင်တဲ့ password

ကို သုံးနိုင် ပါတယ်။

```
> use mysql;
> update user set authentication_string=PASSWORD("my-new-password") where
    User="root";
> flush privileges;
> quit
```

ပြီးတဲ့ အခါ စက်ကို reboot လုပ်နိုင် ပါတယ်။

## ၈.၄ Configuration

တပ်ဆင် ပြီးတာ နဲ့ mysql server က လုပ်ဆောင် နေမှာ ဖြစ်ပြီး သူရဲ့ status ကို အောက်ပါ အတိုင်း ကြည့်နိုင် ပါတယ်။

```
$ sudo service mysql status
```

```
odroid@odroid:~$ sudo service mysql status
● mysql.service - MySQL Community Server
  Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: en
  Active: active (running) since Thu 2018-06-28 15:15:58 +08; 9min ago
    Main PID: 3587 (mysqld)
      CGroup: /system.slice/mysql.service
              └─3587 /usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid

Jun 28 15:15:57 odroid systemd[1]: Starting MySQL Community Server...
Jun 28 15:15:58 odroid systemd[1]: Started MySQL Community Server.
lines 1-9/9 (END)
```

ပုံ ၈.၆: MySQL Service status ကို စစ်ကြည့်ခြင်း။

Server ရဲ့ နားထောင် နေတဲ့ port တွေကို ကြည့်ဖို့ အောက်က command ကို သုံးနိုင် ပါတယ်။ အဲဒီ အခါ ပုံ ၈.၁၀ မှာ ပြထား သလို mysqld က 3306 နဲ့ xdevapi အတွက် 33060 မှာ listen လုပ်နေ တာကို တွေ့ရ မှာ ဖြစ် ပါတယ်။

```
$ sudo netstat -npl
```

```

File Edit View Search Terminal Help
sss@umate:~$ sudo netstat -npl
[sudo] password for sss:
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.53:53          0.0.0.0:*               LISTEN
388/systemd-resolve
tcp        0      0 127.0.0.1:631          0.0.0.0:*               LISTEN
3663/cupsd
tcp6       0      0 :::33060              :::*                  LISTEN
5934/mysqld
tcp6       0      0 :::3306               :::*                  LISTEN
5934/mysqld
tcp6       0      0 :::1:631              :::*                  LISTEN
3663/cupsd
udp        44544   0 127.0.0.53:53          0.0.0.0:*               LISTEN
388/systemd-resolve
udp        0      0 0.0.0.0:68            0.0.0.0:*               LISTEN
887/dhcclient

```

ပုံ ၈၁။ Network port များကို စစ်ကြည့်ခြင်း။

အဲဒီ နောက်မှာ အောက်ပါ အတိုင်း server ရဲ့ security နဲ့ ပတ်သတ် တဲ့ update တစ်ချို့ ကို သတ်မှတ် နိုင် ပါတယ်။

```
$ mysql_secure_installation
```

Source ကနေ build လုပ်ထား တာ ဆိုရင်တော့ binary တွေ ရှိတဲ့ directory ကို သွားဖို့လိုနိုင် ပြီး၊ အောက်ပါ command တွေ အတိုင်း သုံးနိုင် ပါတယ်။

```
$ cd /usr/local/mysql
$ sudo bin/mysql_secure_installation
```

အဲဒီ ကို run လိုက်တဲ့ အခါ root password ကို တောင်းရင် ထည့်ပေး လိုက်ပြီး၊ configuration တွေကို သတ်မှတ် ပါတယ်။ VALIDATE PASSWORD PLUGIN က password တွေရဲ့ strength ကို စစ်ပေးပြီး security ပိုကောင်း အောင် လုပ်ဆောင် ပေးနိုင် ပါတယ်။ ထည့်မယ် ဆိုရင် y ကို နှုပ်ပေး နိုင်ပြီး၊ မထည့်ချင် ရင်တော့ enter ပဲ ဖြစ်ဖြစ် key တစ်ခုခု ကို နှုပ် နိုင် ပါတယ်။ အဲဒီ နောက် root password ကို ပြန်ပြင် ချင်ရင် ပြင်နိုင် ပါတယ်။ နောက် တစ်ခု က MySQL ကို user account မရှုံးလည်း ဝင်လို့ ရအောင် စီမံ ထားတဲ့ anonymous user ပါ။ Root ကို remotely ဝင်လို့ မရအောင် ပိတ် ချင်လည်း ပိတ်နိုင် ပါတယ်။ စမ်းသပ် ဖို့ အတွက် test database ကို ဖယ်မယ် ဆိုရင် လည်း ဖယ်နိုင် ပါတယ်။ ပြီးတဲ့ အခါ အပြောင်း

အလည်း တွေ သက်ရောက်မှု ရှိအောင် privilege tables တွေကို reload လုပ်နိုင် ပါတယ်။

## ၈.၅ MySQL server ကို စမ်းသပ်ခြင်း

MySQL အတွက် utilities တွေကို /usr/local/mysql ရဲ့ bin ထဲမှာ တွေ့နိုင် ပါတယ်။ အဲဒီ directory ကို သွားပြီး ls ဆိုတဲ့ command နဲ့ list လုပ်ကြည့်နိုင် ပါတယ်။

### ၈.၅.၁ mysqladmin

mysqladmin က MySQL server ကို administer လုပ်ဖို့ အတွက် client တစ်ခုပါ။ သူကို သုံးပြီး server version ကို စစ်မယ် ဆိုရင် apt နဲ့ install လုပ်ခဲ့ရင် အောက်ပါ အတိုင်း စစ်ကြည့် နိုင် ပါတယ်။

```
$ sudo mysqladmin -u root -p version
```

Source ကနေ build လုပ်ထား တာ ဆိုရင်တော့ အောက်ပါ command တွေ အတိုင်း သုံးနိုင် ပါတယ်။

```
$ cd /usr/local/mysql
$ sudo bin/mysqladmin -u root -p version
```

Databases တွေကို ကြည့်ချင် ရင် တော့ mysqlshow နဲ့ ကြည့်လို ရပါတယ်။

```
$ sudo bin/mysqlshow -u root -p version
```

### ၈.၅.၂ mysql

mysql command line tool က ရိုးရှင်း တဲ့ SQL shell တစ်ခု ဖြစ်ပါ တယ်။ mysql ကို သုံးဖို့ အတွက် အောက်ပါ command နဲ့ ဝင်နိုင် ဖြီး ပေါ်လာ တဲ့ prompt မှာ mysql command တွေကို semicolon တွေနဲ့ အဆုံး သတ်ပြီး ထည့်နိုင် ပါတယ်။ Apt နဲ့ MySQL server ကို install လုပ်ခဲ့တာ ဆိုရင် တော့ binary ဖိုင်တွေ ရှိတဲ့ path ကို သွား စရာ မလိုပဲ mysql command ကို တန်းသုံး နိုင် ပါတယ်။

```
$ cd /usr/local/mysql
$ sudo bin/mysql -u root -p
```

## ၈၅. MYSQL SERVER ကို စမ်းသပ်ခြင်း

JRC

Database တွေကို အောက်ပါ အတိုင်း ကြည့်နိုင် ပါတယ်။

```
> SHOW DATABASES;
```

နဲ့မူနာ အနေနဲ့ database တစ်ခု ကို ဖန်တီး ကြည့်ဖို့ အတွက် CREATE DATABASE ကို သုံးကြည့် ပါမယ်။

```
> CREATE DATABASE mytest;
```

ပြီးတဲ့ အခါ database တွေကို ပြန် ပြကည့် တဲ့ အခါ ခုနာက ဖန်တီး လိုက် တဲ့ database ပါ ပေါ်လာ တာကို တွေ့နိုင် ပါတယ်။ နောက်ထပ် ထည့် မယ့် command တွေ အတွက် default database ကို mytest လို့ သတ်မှတ်ဖို့ USE ကို သုံးနိုင် ပါတယ်။

```
> USE mytest;
```

အဲဒီ နောက် table တစ်ခု ကို ဖန်တီး ကြည့်ဖို့ CREATE TABLE ကို သုံးလိုက် ပါမယ်။

```
> CREATE TABLE tbl1
(
    id INT unsigned NOT NULL AUTO_INCREMENT, # unique id for the record
    name  VARCHAR(150) NOT NULL, # name
    birthday DATE NOT NULL, # birthday
    PRIMARY KEY (id) # make the id the primary key
);
```

ဖန်တီး လိုက် တဲ့ table ကို SHOW TABLES နဲ့ ကြည့်နိုင် ပါတယ်။

```
> SHOW TABLES;
```

Table ရဲ့ column တွေကို ကြည့်ဖို့ DESCRIBE နဲ့ ကြည့်နိုင် ပါတယ်။

```
> DESCRIBE tbl1;
```

MySQL ကငော ထွက်ပြီး ပုံမှန် bash prompt ကို ပြန်သွား ဖို့ အတွက် exit ကို သုံး လို့လည်း ရ ပါတယ်။

```
$ exit
```

MySQL ရဲ့ လက်ရှိ plugins တွေကို အောက်ပါ အတိုင်း ကြည့်လို လည်း ရပါတယ်။

```
$ mysql -u root -p -e "show plugins"
```

## ၈.၆ Installing MySQL Connector/C++ from source

MySQL server ကို C++ application ကနေ သုံးဖို့ အတွက် MySQL Connector/C++ ကို အသုံးပြု နိုင် ပါတယ် [Ora18a]။ Connector C++ 8.0 ကို သုံးတဲ့ အတွက် C++ application တွေမှာ X DevAPI ကို သုံးလို ရပြီး၊ plain SQL queries တွေ လုပ်ဆောင် လို ရသလို၊ document store ကို သုံးထား တဲ့ MySQL server တွေကို လည်း X Plugin သုံးပြီး ဆက်သွယ် လို ရပါတယ်။

### ၈.၆.၁ Build Tools

Connector C++ ကို build လုပ်ဖို့ အတွက် C++11 ကို အထောက် အပံ့ ပေးနိုင် တဲ့ C++ compiler လိုပါတယ်။ Cross platform build tool တစ်ခု ဖြစ်တဲ့ CMake လည်း ရှိဖို့ လိုပါတယ်။ Git repository ကနေ ရယူဖို့ အတွက် git ကို လည်း တပ်ဆင် ထားဖို့ လိုမှာ ပါ။ အဲဒါ တွေ မရှိ သေးရင် အောက်ပါ အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt update
$ sudo apt install build-essential
$ sudo apt install cmake git
```

### ၈.၆.၂ ရယူခြင်း

အဲဒါ အတွက် Download Connector/C++ Web Page ကိုသွား၊ dropdown box မှာ source ကို ရွှေးပြီး download လုပ်၊ extract လုပ်လို ရသလို၊ git clone ကို သုံး ရင်လည်း ရပါတယ်။

Extract လုပ်ဖို့ အောက်ပါ command လိုမျိုး သုံးနိုင် ပါတယ်။ ပြီးတဲ့ အခါ ရလာ တဲ့ directory ကို သွားပါမယ်။

```
$ tar zxvf mysql-connector-c++-8.0.11-src.tar.gz
$ cd mysql-connector-c++-8.0.11-src
```

Git repository ကောင် clone လုပ်ဖို့ အတွက် တော့

```
$ git clone https://github.com/mysql/mysql-connector-cpp.git
```

ကို သုံးနှင့် ပါတယ်။

ပြီးတဲ့ အခါ ရလာ တဲ့ directory ကို သွားပြီး 8.0 branch ကို checkout လုပ်လိုက် ပါမယ်။

```
$ cd mysql-connector-cpp
$ git checkout 8.0
```

## ၈.၇ Configuring

Connector C++ 8.0 ကို configure လုပ်၊ build လုပ် ဖို့ အတွက် CMake ကို သုံးပါ မယ်။ Build လုပ်ဖို့ directory တစ်ခု ဖန်တီးပြီး အဲဒီ directory ကို သွားပါ မယ်။

```
$ mkdir build
$ cd build
```

Default install လုပ်မယ့် နေရာ က /usr/local/mysql/connector-c++-8.0 ဖြစ်ပြီး၊ CMAKE\_INSTALL\_PREFIX ကို သုံးပြီး စိတ်ကြော် သတ်မှတ် လိုလည်း ရပါတယ်။ ဘာမှ မပြောရင် dynamic (shared) libraries ကို build လုပ်မှာ ဖြစ်ပြီး၊ static libraries ကို build လုပ်ချင်ရင် -DBUILD\_STATIC=ON ကို သတ်မှတ် နိုင် ပါတယ်။

```
$ cmake -DCMAKE_INSTALL_PREFIX=/usr/local/mysql/connector-c++-8.0 ..
```

### ၈.၆.၅ Building

အဲဒီလို Configure လုပ်ဖြီး တဲ့ အခါ build လုပ်ဖို့ အတွက် အောက်ပါ command ကို သုံးနိုင် ပါတယ်။ Release သို့မဟုတ် Debug သတ်မှတ်ဖို့ -config option ကို သုံးနိုင် ပြီး၊ အဲဒီ option မထည့်ရင် Release ကို build လုပ်မှာ ဖြစ် ပါတယ်။

```
$ cmake --build . --config Release
```

Build လုပ်လိုက် တဲ့ အခါ

```
libmysqlcppconn8.so.1
```

ဆိုတဲ့ connector library ကို build directory ထဲမှာ ရလာ ပါမယ်။

### ၈.၆.၆ Installing

Build လုပ်ဖြီး တဲ့ အခါ ရလာတဲ့ Connector C++ 8.0 libraries ထွေကို install လုပ်ဖို့ အောက်က command ကို သုံး ပါမယ်။

```
$ sudo cmake --build . --target install --config Release
```

အဲဒီ နောက် /usr/local/mysql/connector-c++-8.0/ မှာ lib64 ဆိုတဲ့ directory မရှိ သေးရင် ဖန်တီး လိုက် ပါမယ်။ 32 bit architecture ထွေပေါ် မှာတော့ lib64 အစား lib လိုပဲ ခေါ်ပါမယ်။

```
$ sudo mkdir /usr/local/mysql/connector-c++-8.0/lib64
```

ပြီးတဲ့ အခါ အောက်က command ကို သုံးပြီး၊ library ကို ကူးပါမယ်။

```
$ sudo cp libmysqlcppconn8.so /usr/local/mysql/connector-c++-8.0/lib64/
$ sudo cp libmysqlcppconn8.so.1.8.0.11 /usr/local/mysql/connector-c++-8.0/
lib64/
```

ပြီးတဲ့ အခါ /etc/ld.so.conf.d/ ဆိုတဲ့ directory ထဲမှာ mysqlcppconn8.conf ဆိုတဲ့ ဖိုင် တစ်ခု ဖန်တီး ပြီး edit လုပ် ပါမယ်။

```
$ sudo nano /etc/ld.so.conf.d/mysqlcppconn8.conf
```

ပွင့်လာ တဲ့ nano editor မှာ အောက်ပါ အတိုင်း ဖြည့်ပြီး၊ ctrl+o နှင့် enter ခလုတ် ကို နိပ်ပြီး ဖိုင် ကို save လုပ်၊ ctrl+x နဲ့ ထွက်လိုက် ပါမယ်။

```
/usr/local/mysql/connector-c++-8.0/lib64/
```

အဲဒီ နောက် အောက်ပါ အတိုင်း ldconfig လုပ်ပါမယ်။

```
$ sudo ldconfig
```

### ၈.၆.၆ Connector C++ Application များ build လုပ်ခြင်း

MySQL Connector C++ 8.0 ၊ MySQL Server 8.0 ရဲ့ စိတ်လှပ်ရှား စရာ feature အသစ် တွေကို အသုံးပြု နိုင် ပါတယ်။ ဥပမာ database လုပ်ငန်း ဆောင်တာ တွေကို ရိုးရှင်း ပြီး၊ ထိထိ ရောက်ရောက် သုံးလို့ ရတဲ့ MySQL Document Store လိုမျိုး အသစ် တွေကို C++ ရဲ့ အားသာ ချက်တွေနဲ့ ပေါင်းပြီး သုံးလို့ ရသွား ပါတယ်။ X Protocol ကို အထောက် အပံ့ ပေးတာ မို့ သူ့ကို အသုံးပြု ပြီး X DevAPI ကို သုံးတဲ့ နမူနာ C++ ပရိုဂရမ် တစ်ခု ရေးကြည့် ပါမယ် [Ora18b]။ Connector C++ ကို သုံးဖို့ ပရိုဂရမ် အစမှာ အောက်ပါ ကုဒ်တွေကို ထည့်ပါမယ်။

```
#include <mysqlx/xdevapi.h>
using namespace mysqlx;
```

Database connection အတွက် X Plugin လုပ်ဆောင် မှု ရှိတဲ့ MySQL server တွေကို logical session တွေ တည်ဆောက် နိုင် ပါတယ်။ Session object တွေကို သုံးထား တဲ့ application တွေက single server ပေါ်မှာ ပဲ ဖြစ်ဖြစ်၊ database cluster ပေါ်မှာ ပဲ ဖြစ်ဖြစ် ကုဒ် ကို ပြန်ပြောင်း စရာ မလိုပဲ သုံးနိုင် ပါတယ်။ Database ကို connection တည်ဆောက် ဖို့ URI type string ကို သုံးပြီး

```
Session my_session("user:password@host:port");
```

လို့ ချိတ်ဆက် လို့ ရသလို

```
Session my_session("host",port,"user","password");
```

လို့လည်းဆက်လို့ရပါတယ်။ Session ချိတ်ဆက် လို့ရပြီးတဲ့ အခါ server မှာရှိတဲ့ database တွေကို list လုပ်ကြည့် ပါမယ်။ အဲဒီ အတွက် database list ကို getSchemas() method နဲ့ရယူ နိုင် ပါတယ်။

```
std::list<Schema> schemaList = my_session.getschemas();
```

နဲ့မူနာ mysqltest.cpp ပရိုက်ရမဲ့ ကို စာရင်း ၈.၁ မှာ ပြထား ပါတယ်။

```
1 #include <iostream>
2 #include <mysqlx/xdevapi.h>
3 using namespace std;
4 using namespace mysqlx;
5
6 int main()
7 try {
8     cout <<"Getting session..." << endl;
9     Session sess("root:password@127.0.0.1:33060");
10    //Session sess("localhost",33060,"root","password");
11    cout <<"Session accepted, getting schemas list ..." << endl;
12
13    //Get a list of all available schemas
14    std::list<Schema> schemaList = sess.getschemas();
15    cout <<"Available schemas in this session:" << endl;
16
17    //loop over all available schemas and print their name
18    for(Schema schema : schemaList) {
19        cout << schema.getName() << endl;
20    }
21 }
22 catch (const mysqlx::Error &err)
23 {
24     cout << "ERROR: " << err << endl;
25 }
```

စာရင်း ၈.၁: MySQL Connector C++ 8.0 ကို သုံး၍ database များကို list လုပ်ကြည့်ခြင်း။

C++ application တွေကို build လုပ်တဲ့ အခါ MySQL Connector C++ 8.0 ကို build လုပ်တဲန်းက သုံးခဲ့တဲ့ tools တွေကို ပဲ သုံးရဲ ပါမယ်။ Compiler version, runtime library, runtime linker configuration settings စတာ တွေ တူညီ ဖို့ လိုပါ တယ်။ C++11 ကို အထောက် အပံ့ ပေးဖို့ -std=c++11 ဆိုတဲ့ option ကို လည်း ထည့်ပေး ဖို့ လို ပါတယ်။ Build configuration ကိုလည်း Release ဒါမှ မဟုတ် Debug option က Connector C++ အတိုင်း ဖြစ်ဖို့ လိုပါ တယ်။ အဲဒီ နောက် အောက်က အတိုင်း build လုပ်ပြီး run နိုင် ပါတယ်။

```
$ g++ -std=c++11 -I /usr/local/mysql/connector-c++-8.0/include -L /usr/local/
    mysql/connector-c++-8.0/lib64 mysqltest.cpp -lmysqlcppconn8 -o mysqltest
$ ./mysqltest
```

Build လုပ်ဖို့ အတွက် make ဖိုင်၊ ဒါမှ မဟုတ် ရှိုးရှင်း တဲ့ scrpit ဖိုင် တစ်ခုခု ရေးထား ပြီး သုံးတာက ပို အဆင်ပြေ ပါတယ်။ ပရိုဂရမ် ရဲ့ ရလဒ် ကို ပုံ ၈.၁၁ မှာ ပြထား ပါတယ်။

```
pi@raspberrypi:~/mysqltestcpp $ ./mysqltest-bar.sh
Getting session...
Session accepted, getting schemas list ...
Available schemas in this session:
information_schema
mysql
mytest
performance_schema
sys
test
pi@raspberrypi:~/mysqltestcpp $ _
```

ပုံ ၈.၁၁: mysqltest.cpp ၏ ရလဒ်။

## အကိုးအကားများ

[Bou17a] Brian Boucheron. How To Install the Latest MySQL on Ubuntu 16.04. 2017.

url: <https://www.digitalocean.com/community/tutorials/how-to-install-the-latest-mysql-on-ubuntu-16-04>.

- [Ora18a] Oracle. MySQL Connector/C++ 8.0 Developer Guide. 2018. url: <https://dev.mysql.com/doc/connector-cpp/8.0/en/>.
- [Ora18b] Oracle. X DevAPI User Guide. 2018. url: <https://dev.mysql.com/doc/x-devapi-userguide/en/>.
- [Ora18a] Oracle. A Quick Guide to Using the MySQL APT Repository. 2018. url: <https://dev.mysql.com/doc/mysql-apt-repo-quick-guide/en/>.
- [Ora18b] Oracle. Installing MySQL Using a Standard Source Distribution. 2018. url: <https://dev.mysql.com/doc/refman/8.0/en/installing-source-distribution.html>.

## အခန်း ၉

# Internet of Things

အီလက်ထရောနစ် ပစ္စည်း တွေဖြစ်တဲ့ အိမ် အသုံးအဆောင် တွေ၊ ကိရိယာ တွေ၊ မော်တော်ယာ၏တွေ၊ အစရှိတဲ့ ပစ္စည်း ကိရိယာ၊ တန်ဆာပလာ တွေ network ချိတ်ဆက် အသုံးပြု တဲ့ ကို internet of things (IoT) လို ခေါ် ပါတယ်။ wxWidgets ကို သုံးပြီး UDP စတဲ့ protocol တွေ နဲ့ network ပေါ်မှာ ဒေတာ အပြန် အလုန် ပေးပို့ ဆက်သွယ် တဲ့ အကြောင်း ဆွေးနွေး ချင် ပါတယ်။ အင်တာနက် စတဲ့ network တွေ ပေါ်မှာ TCP ဒါမှုမဟုတ် UDP တွေသုံးပြီး စက်တစ်ခု နဲ့ တစ်ခု ဒေတာ တွေ အပြန် အလုန် ပို့ဖို့ အတွက် socket တွေကို အသုံးပြု နိုင် ပါတယ်။ ခေတ်ပေါ် operating system တွေ အားလုံးက socket layer ကို အထောက် အပံ့ ပေးကြ ပေမယ့် platform ပေါ်မှု တည်ပြီး socket ကို အသုံး ပြုရတဲ့ ပုံစံ တွေက အမျိုးမျိုး ကွဲပြား နိုင်ပါတယ်။ wxWidgets မှာ အောက်လုံး platform အတွက် ပူစရာ မလိုပဲ အလွယ် တကူ အသုံးပြုနိုင်တဲ့ socket class ပါ ပါတယ်။ အဲဒီ class ကို မတူညီတဲ့ ပုံစံ နည်းလမ်း အမျိုးမျိုး နဲ့ အသုံးပြုနိုင်ပြီး၊ အသုံးပြုပဲ နမူနာ တရာ့ကို အောက်မှာ ဆက်ပြီး ဖော်ပြထား ပါတယ်။

### ၉.၁ UDP

IP (Internet Protocol) network တွေ ပေါ်မှာ စက်တစ်ခု ကနေ တစ်ခု ကို UDP (User Datagram Protocol) သုံးပြီး (Datagram လိုလည်း ခေါ်ကြတဲ့) message တွေကို ပိုလို ရပါတယ်။ UDP က ရိုးရှင်း တဲ့ connectionless communication ပုံစံ ကို သုံးတဲ့ အတွက် ဒေတာ မပိုခင် ချိတ်ဆက် တာတွေ၊ handshake လုပ်တာတွေ၊ အမှား ပြင်တာ တွေ မပါပဲ ပေါ့ပါး မှု ရှိပြီး ပိုလိုက်တဲ့ ဒေတာ မှန်မမှန် ကိုပဲ checksum သုံးပြီး စစ်ပါတယ်။ ပိုလိုက်တဲ့ ဒေတာ က မှားသွား၊ ထပ်သွား လည်း ပြန်ပို စရာ မလို၊ ပြင်စရာ မလိုပဲ မြန်ဆန် သွက်လက် ဖို့ပဲ အရေးကြီး တဲ့ နေရာ (ဥပမာ Voice over IP) တွေက UDP နဲ့ သင့်တော်

ပါတယ်။

wxWidgets ကို ထည့်သွင်းတပ်ဆင်ပြီး တဲ့ အခါ samples ဆိုတဲ့ အခန်းထဲက sockets ထဲမှာ network ချိတ်ဆက် အသုံးပြုတဲ့ နမူနာ [GZ09] ထဲမှာ UDP သုံးတာ ပြထားပါတယ်။ အဲဒီ နမူနာ က တခြား TCP တွေကို ရော ပြည့်စုံ အောင် ပေါင်းပြ ထားတဲ့ အတွက် ရှုပ်ထွေး ခက်ခဲ မှု အနည်းငယ် ရှိတာ ရယ်၊ event ကို မသုံးပဲ data ပြန်မရ မခြင်း ရပ် နေတာ တွေ ရှိတာ ကြောင့်၊ သူ့ကို အခြေခံ ပြင်ဆင် ထားတဲ့ ဂိုပြီး ရှိရင်း လွယ်ကူတဲ့ UDP သီးသန် ce\_wx\_udp.cpp ဆိုတဲ့ နမူနာ တစ်ခု ကို စာရင်း ၉.၁ မှာ ဖော်ပြ ထားပါတယ်။

```

1 // File: ce_wx_udp.cpp
2 // Description: A simpler version of wxWidgets UDP sample
3 // Author: Yan Naing Aye
4 // Web: http://cool-emerald.blogspot.com
5 // MIT License (https://opensource.org/licenses/MIT)
6 // Copyright (c) 2018 Yan Naing Aye
7
8 // References
9 // [1] Guillermo Rodriguez Garcia, Vadim Zeitlin,
10 //      "Server for wxSocket demo",
11 //      https://github.com/wxWidgets/wxWidgets/blob/master/samples/sockets/
12 //          server.cpp, 2009.
13
14 #include "wx/wxprec.h"
15
16 #ifdef __BORLANDC__
17 #pragma hdrstop
18#endif
19
20#ifndef WX_PRECOMP
21#include "wx/wx.h"
22#endif
23
24#include "wx/socket.h"
25
26// this example is currently written to use only IP or only IPv6 sockets, it

```

```
27 // should be extended to allow using either in the future
28 #if wxUSE_IPV6
29     typedef wxIPV6address IPaddress;
30 #else
31     typedef wxIPV4address IPaddress;
32 #endif
33
34 #ifndef wxHAS_IMAGES_IN_RESOURCES
35 #include "./sample.xpm"
36 #endif
37
38 // Define a new application type, each program should derive a class from
39 // wxApp
40 class MyApp : public wxApp
41 {
42     public:
43         // override base class virtuals
44         // -----
45         // this one is called on application startup and is a good place for the app
46         // initialization (doing it here and not in the ctor allows to have an error
47         // return: if OnInit() returns false, the application terminates)
48     virtual bool OnInit();
49 };
50
51 // Define a new frame type: this is going to be our main frame
52 class MyFrame : public wxFrame
53 {
54     public:
55         // ctor(s)
56     MyFrame(const wxString& title);
57
58         // event handlers (these functions should _not_ be virtual)
59     void OnQuit(wxCommandEvent& event);
60     void OnAbout(wxCommandEvent& event);
61     void OnSend(wxCommandEvent& event);
```

```
62 void OnSocketEvent(wxSocketEvent& event);
63
64
65 wxDatagramSocket *sock;
66 wxButton *btnSend;
67 wxTextCtrl *txtSend;
68 wxTextCtrl *txtRx;
69 // any class wishing to process wxWidgets events must use this macro
70 wxDECLARE_EVENT_TABLE();
71 }
72
73 // constants
74 const int ID_BTNSEND = 101;
75 const int ID_TXTSEND = 102;
76 const int ID_TXTRX = 103;
77
78 // IDs for the controls and the menu commands
79 enum
80 {
81     Button_Send = ID_BTNSEND,
82     Txt_Send = ID_TXTSEND,
83     Txt_Rx = ID_TXTRX,
84     SOCKET_ID,
85     // menu items
86     Minimal_Quit = wxID_EXIT,
87
88     // it is important for the id corresponding to the "About" command to have
89     // this standard value as otherwise it won't be handled properly under Mac
90     // (where it is special and put into the "Apple" menu)
91     Minimal_About = wxID_ABOUT
92 };
93
94
95 // the event tables connect the wxWidgets events with the functions (event
96 // handlers) which process them. It can be also done at run-time, but for the
97 // simple menu events like this the static method is much simpler.
```

```
98 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
99 EVT_MENU(Minimal_Quit, MyFrame::OnQuit)
100 EVT_MENU(Minimal_About, MyFrame::OnAbout)
101 EVT_SOCKET(SOCKET_ID, MyFrame::OnSocketEvent)
102 wxEND_EVENT_TABLE()

103

104 // Create a new application object: this macro will allow wxWidgets to create
105 // the application object during program execution (it's better than using a
106 // static object for many reasons) and also implements the accessor function
107 // wxGetApp() which will return the reference of the right type (i.e. MyApp
108 // and
109 // not wxApp)
110 IMPLEMENT_APP(MyApp)

111 // 'Main program' equivalent: the program execution "starts" here
112 bool MyApp::OnInit()
113 {
114 // call the base class initialization method, currently it only parses a
115 // few common command-line options but it could be do more in the future
116 if (!wxApp::OnInit())
117 return false;
118
119 // create the main application window
120 MyFrame *frame = new MyFrame("wxWidgets UDP App");
121
122 // and show it (the frames, unlike simple controls, are not shown when
123 // created initially)
124 frame->Show(true);
125
126 // success: wxApp::OnRun() will be called which will enter the main message
127 // loop and the application will run. If we returned false here, the
128 // application would exit immediately.
129 return true;
130 }
131
132 // frame constructor
```

```
133 MyFrame::MyFrame(const wxString& title)
134 : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(390, 280),
135   wxDEFAULT_FRAME_STYLE ^ wxRESIZE_BORDER)
136 {
137   // set the frame icon
138   SetIcon(wxICON(sample));
139
140 #if wxUSE_MENUS
141   // create a menu bar
142   wxMenu *fileMenu = new wxMenu;
143
144   // the "About" item should be in the help menu
145   wxMenu *helpMenu = new wxMenu;
146   helpMenu->Append(Minimal_About, "&About\tF1", "Show about dialog");
147
148   fileMenu->Append(Minimal_Quit, "E&xit\tAlt-X", "Quit this program");
149
150   // now append the freshly created menu to the menu bar...
151   wxMenuBar *menuBar = new wxMenuBar();
152   menuBar->Append(fileMenu, "&File");
153   menuBar->Append(helpMenu, "&Help");
154
155   // ... and attach this menu bar to the frame
156   SetMenuBar(menuBar);
157 #endif // wxUSE_MENUS
158
159 #if wxUSE_STATUSBAR
160   // create a status bar just for fun (by default with 1 pane only)
161   CreateStatusBar(2);
162   SetStatusText("UDP using wxWidgets");
163 #endif // wxUSE_STATUSBAR
164   btnSend = new wxButton(this, Button_Send, wxT("Send"),
165     wxPoint(5, 5), wxSize(100, 25));
166   txtSend = new wxTextCtrl(this, Txt_Send, wxT("Hello!"),
167     wxPoint(120, 5), wxSize(250, 25));
168   txtRx = new wxTextCtrl(this, Txt_Rx, wxT(""), wxPoint(5, 35),
```

```
169         wxSize(365, 125), wxTE_MULTILINE);  
170  
171 Connect(Button_Send, wxEVT_COMMAND_BUTTON_CLICKED,  
172     wxCommandEvent::OnSend));  
173  
174 // Create the address - defaults to localhost:0 initially  
175 IPaddress addr;  
176 addr.AnyAddress();  
177 addr.Service(3000);  
178 txtRx->AppendText(wxString::Format(wxT("Creating UDP socket at %s:%u \n"),  
179     addr.IPAddress(), addr.Service()));  
180  
181 // Create the socket  
182 sock = new wxDatagramSocket(addr);  
183  
184 // We use IsOk() here to see if the server is really listening  
185 if (!sock->IsOk()) {  
186     txtRx->AppendText(wxString::Format(wxT("Could not listen at the specified  
187         port !\n")));  
188     return;  
189 }  
190  
191 IPaddress addrReal;  
192 if (!sock->GetLocal(addrReal)) {  
193     txtRx->AppendText(wxString::Format(wxT("ERROR: couldn't get the address we  
194         bound to. \n")));  
195 }  
196 else {  
197     txtRx->AppendText(wxString::Format(wxT("Server listening at %s:%u \n"),  
198         addrReal.IPAddress(), addrReal.Service()));  
199 }  
200  
201 // Setup the event handler  
202 sock->SetEventHandler(*this, SOCKET_ID);  
203 sock->SetNotify(wxSOCKET_INPUT_FLAG);  
204 sock->Notify(true);  
205 }
```

```
203
204
205 // event handlers
206 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
207 {
208 // true is to force the frame to close
209 Close(true);
210 }
211
212 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
213 {
214 wxMessageBox(wxString::Format
215 (
216 "wxWidgets UDP sample\n"
217 "\n"
218 "Author: Yan Naing Aye \n"
219 "Web: http://cool-emerald.blogspot.com"
220 ),
221 "About wxWidgets UDP sample",
222 wxOK | wxICON_INFORMATION,
223 this);
224 }
225
226 void MyFrame::OnSend(wxCommandEvent& WXUNUSED(event))
227 {
228 wxString str = txtSend->GetValue();
229 wxCharBuffer buffer = str.ToUTF8();
230 size_t txn = str.length();
231
232 IPaddress raddr;
233 raddr.Hostname("localhost");
234 //raddr.Hostname("192.168.2.71");
235 raddr.Service(3001);
236 if (sock->SendTo(raddr, buffer.data(), txn).LastCount() != txn)
237 {
238 txtRx->AppendText(wxT("Write error.\n"));

```

```
239     return;
240 }
241 else {
242     txtRx->AppendText("Tx: "+str+"\n");
243 }
244 }

245

246 void MyFrame::OnSocketEvent(wxSocketEvent& event)
247 {
248     IPaddress addr;
249     addr.Service(3000);
250     char buf[1024];
251     size_t n;
252     switch(event.GetSocketEvent())
253     {
254     case wxSOCKET_INPUT:
255         //txtRx->AppendText("OnSocketEvent: wxSOCKET_INPUT\n");
256         sock->Notify(false);
257         n = sock->RecvFrom(addr, buf, sizeof(buf)).LastCount();
258         if (!n) {
259             txtRx->AppendText("ERROR: failed to receive data \n");
260             return;
261         }
262         //txtRx->AppendText(wxString::Format(wxT("Received \"%s\" from %s:%u.\n"))
263         ,
264         //  wxString::From8BitData(buf, n),addr.IPAddress(),addr.Service()));
265         txtRx->AppendText("Rx: "+wxString::From8BitData(buf, n) + "\n");
266         sock->Notify(true);
267         break;
268     default: txtRx->AppendText("OnSocketEvent: Unexpected event !\n"); break;
269 }
```

օջախ: Q.○: ce\_wx\_udp.cpp

ပရိုဂရမ် အစမှာ wxWidgets နဲ့ socket အတွက် header ဖိုင်တွေ နဲ့ IP address အမျိုးအစား တွေကို အောက်ပါ အတိုင်း သတ်မှတ် နိုင်ပါတယ်။

```
#include "wx/wx.h"
#include "wx/socket.h"
#if wxUSE_IPV6
typedef wxIPV6address IPEndPoint;
#else
typedef wxIPV4address IPEndPoint;
#endif
```

ပြီးတဲ့ အခါ လိုချင်တဲ့ IP address + port number တွေနဲ့ UDP socket တစ်ခု ကို ဖန်တီးပြီး အသုံးပြုချင်တဲ့ event တွေကို သတ်မှတ် နိုင် ပါတယ်။

```
// Create the address - defaults to localhost:0 initially
IPEndPoint addr;
addr.AnyAddress();
addr.Service(3000);

// Create the socket
sock = new wxDatagramSocket(addr);

// Setup the event handler
sock->SetEventHandler(*this, SOCKET_ID);
sock->SetNotify(wxSOCKET_INPUT_FLAG);
sock->Notify(true);
```

ဒေတာ တွေလက်ခံ ရရှိတဲ့ အခါ OnSocketEvent ရဲ့ wxSOCKET\_INPUT အမျိုးအစား event မှာ RecvFrom method ကို သုံးပြီး ဖတ်နိုင် ပါတယ်။

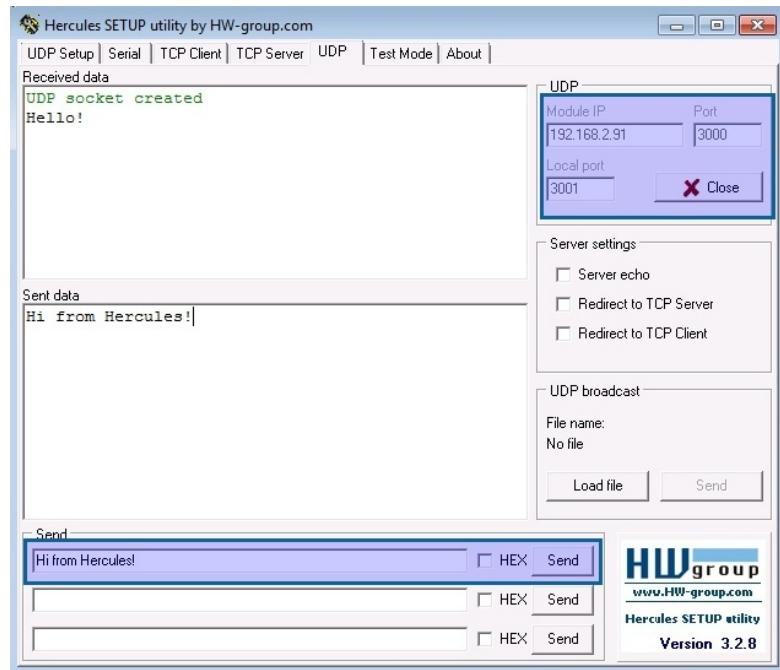
```
n = sock->RecvFrom(addr, buf, sizeof(buf)).LastCount();
```

ဒေတာ ပိုဖို အတွက် ပိုမယ့် remote host ရဲ့ address နဲ့ port number ကို သတ်မှတ်ပြီး၊ SendTo method နဲ့ ပို့နိုင် ပါတယ်။

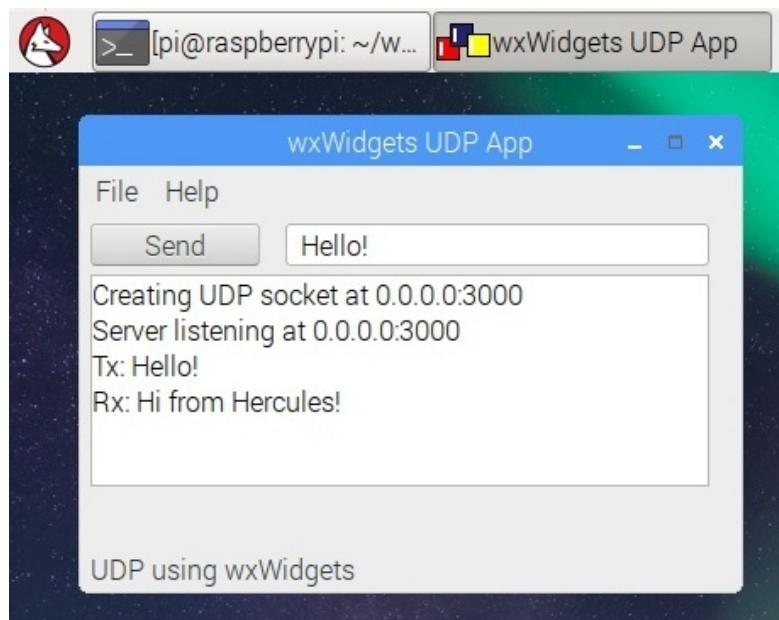
```
IPaddress raddr;
raddr.Hostname("localhost");
raddr.Service(3001);
sock->SendTo(raddr, buf,n)
```

ဒီ ပရိုဂရမ် ကို စမ်းကြည့်ဖို့ အတွက် [https://www.hw-group.com/products/hercules/index\\_en.html](https://www.hw-group.com/products/hercules/index_en.html) မှာ free ရနိုင်တဲ့ Hercules Utility ကို သုံးနိုင် ပါတယ်။ Hercules ကို သူနဲ့ ဆက်သွယ် စမ်းသပ်မယ့် Windows ကွန်ပျူော့ဘာ မှာ တပ်ဆင် လိုက်ပြီး ပရိုဂရမ် ကို ဖွင့်ပြီး တဲ့ အခါ IP address နဲ့ UDP port ကို သတ်မှတ်၊ နားထောင် ပါမယ်။ စာရင်း ၉.၁ က ပရိုဂရမ်ကို အောက်က အတိုင်း build လုပ်ပြီး run လိုက်တဲ့ အခါ သူတို့ အချင်းချင်း ဒေတာ တွေ အပြန်အလှန် ပို့နိုင် တာကို ပုံ ၉.၁ နဲ့ ပုံ ၉.၂ မှာ ပြထား သလို တွေ့နိုင် ပါတယ်။

```
g++ ce_wx_udp.cpp `wx-config --cxxflags --libs` -o ce_wx_udp
gksudo ./ce_wx_udp
```



ပုံ ၉.၁: Hercules utility ကို အသုံးပြုခြင်း။



ဦး ၉.၂: UDP ဖြင့် ဒေတာများ ပို့ခြင်း နှင့် လက်ခံခြင်း။

## ၉.၂ TCP

TCP (Transmission Control Protocol) က အဓိက ကျွတဲ့ network protocol တွေထဲမှာ တခု အပါအဝင် ဖြစ်ပါတယ်။ သူက ဒေတာ တွေပို့ တဲ့ အခါ စိတ်ချ ယုံကြည် ရပြီး၊ အစီအစဉ် တကျ ရောက်အောင်၊ အမှား မပါ အောင် ပို့ဆိုင် ပါတယ်။ TCP နဲ့ ဒေတာ တွေ ပို့ဆိုင် ဖို့ အရင်ဆုံး connection ကို ချိတ်ဆက်ဖို့ လိုပါတယ်။ အဲဒီ အတွက် သတ်မှတ်ထား တဲ့ port number ကို နားထောင် နေမယ့် server နဲ့ အဲဒီ ကို လှမ်း ဆက်သွယ် ပြီး ချိတ်ဆက်မှု ကို စတင် မယ့် client ဆိုပြီး နှစ်မျိုး ရှိပါ တယ်။

### ၉.၂.၁ TCP Server

TCP server က သတ်မှတ် ထားတဲ့ port number တစ်ခု မှာ passively နားထောင် နေပြီး၊ သူကို လာဆက် သွယ်တဲ့ client နဲ့ ဒေတာ တွေ အပြန်အလှန် ပို့ဆိုင် ပါတယ်။ Client တွေ ဆီက ဒေတာ တွေကို လက်ခံ ဖော်ပြပြီး၊ ပြန်ပို့ပေး၊ ပြီးတော့ လက်ရှိ ဆက်သွယ် နေတဲ့ client အရေအတွက် တွေကို ပါဖော်ပြ ပေးတဲ့ `ce_wx_tcp_server.cpp` ဆိုတဲ့ TCP server နမူနာ [GZ09] တစ်ခု ကို စာရင်း ၉.၂ မှာ ဖော်ပြထား ပါတယ်။

```
// File: ce_wx_tcp_server.cpp
```

```
2 // Description: A simple wxWidgets TCP server sample
3 // Author: Yan Naing Aye
4 // Web: http://cool-emerald.blogspot.com
5 // MIT License (https://opensource.org/licenses/MIT)
6 // Copyright (c) 2018 Yan Naing Aye
7
8 // References
9 // [1] Guillermo Rodriguez Garcia, Vadim Zeitlin, "Server for wxSocket demo",
10 //      https://github.com/wxWidgets/wxWidgets/blob/master/samples/sockets/
11 //          server.cpp, 2009.
12 // [2] Julian Smart and Kevin Hock, "Cross-Platform GUI Programming with
13 //      wxWidgets,"
14 //      Pearson Education, Inc. 2006. ISBN: 0-13-147381-6.
15
16 #ifdef __BORLANDC__
17 #pragma hdrstop
18#endif
19
20#ifndef WX_PRECOMP
21#include "wx/wx.h"
22#endif
23
24#include "wx/socket.h"
25
26// this example is currently written to use only IP or only IPv6 sockets, it
27// should be extended to allow using either in the future
28#if wxUSE_IPV6
29typedef wxIPV6address IPEndPoint;
30#else
31typedef wxIPV4address IPEndPoint;
32#endif
33
34#ifndef wxHAS_IMAGES_IN_RESOURCES
35#include "./sample.xpm"
```

```
36 #endif
37
38 class MyApp : public wxApp
39 {
40 public:
41
42 virtual bool OnInit();
43 };
44
45 // Define a new frame type: this is going to be our main frame
46 class MyFrame : public wxFrame
47 {
48 public:
49 // ctor(s)
50 MyFrame(const wxString& title);
51 ~MyFrame();
52 // event handlers (these functions should _not_ be virtual)
53 void OnQuit(wxCommandEvent& event);
54 void OnAbout(wxCommandEvent& event);
55 void OnServerEvent(wxSocketEvent& event);
56 void OnSocketEvent(wxSocketEvent& event);
57 private:
58
59 wxSocketServer *sock;
60 wxTextCtrl *txtRx;
61 int numClients;
62 // any class wishing to process wxWidgets events must use this macro
63 wxDECLARE_EVENT_TABLE();
64 };
65
66 // IDs for the controls and the menu commands
67 enum
68 {
69 ID_TXTRX=101,
70 SOCKET_ID,
71 SERVER_ID,
```

```
72 // menu items
73 Minimal_Quit = wxID_EXIT,
74
75 Minimal_About = wxID_ABOUT
76 };
77
78 // event tables and other macros for wxWidgets
79 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
80 EVT_MENU(Minimal_Quit, MyFrame::OnQuit)
81 EVT_MENU(Minimal_About, MyFrame::OnAbout)
82 EVT_SOCKET(SOCKET_ID, MyFrame::OnSocketEvent)
83 EVT_SOCKET(SERVER_ID, MyFrame::OnServerEvent)
84 wxEND_EVENT_TABLE()
85
86 IMPLEMENT_APP(MyApp)
87
88 // 'Main program' equivalent: the program execution "starts" here
89 bool MyApp::OnInit()
90 {
91 if ( !wxApp::OnInit() )
92 return false;
93 MyFrame *frame = new MyFrame("wxWidgets TCP Server");
94 frame->Show(true);
95 return true;
96 }
97
98 // frame constructor
99 MyFrame::MyFrame(const wxString& title)
100 : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(390, 280),
101 wxDEFAULT_FRAME_STYLE ^ wxRESIZE_BORDER)
102 {
103 // set the frame icon
104 SetIcon(wxICON(sample));
105
106 #if wxUSE_MENUS
107 // create a menu bar
```

```
108 wxMenu *fileMenu = new wxMenu;
109
110 // the "About" item should be in the help menu
111 wxMenu *helpMenu = new wxMenu;
112 helpMenu->Append(Minimal_About, "&About\tF1", "Show about dialog");
113
114 fileMenu->Append(Minimal_Quit, "E&xit\tAlt-X", "Quit this program");
115
116 // now append the freshly created menu to the menu bar...
117 wxMenuBar *menuBar = new wxMenuBar();
118 menuBar->Append(fileMenu, "&File");
119 menuBar->Append(helpMenu, "&Help");
120
121 // ... and attach this menu bar to the frame
122 SetMenuBar(menuBar);
123 #endif // wxUSE_MENUS
124
125 #if wxUSE_STATUSBAR
126 // create a status bar just for fun (by default with 1 pane only)
127 CreateStatusBar(2);
128 SetStatusText("TCP server using wxWidgets");
129 #endif // wxUSE_STATUSBAR
130 txtRx = new wxTextCtrl(this, ID_TXTRX, wxT(""), wxPoint(5, 5),
131                     wxSize(365, 125), wxTE_MULTILINE);
132
133 // Create the address - defaults to localhost:0 initially
134 IPaddress addr;
135 addr.AnyAddress();
136 addr.Service(3000);
137 txtRx->AppendText(wxString::Format(wxT("Creating server at %s:%u \n"))
138 ,addr.IPAddress(), addr.Service()));
139
140 // Create the socket
141 sock = new wxSocketServer(addr);
142
143 // We use IsOk() here to see if the server is really listening
```

```
144 if (!sock->IsOk()){
145     txtRx->AppendText(wxString::Format(wxT("Could not listen at the specified
146         port !\n")));
147     return;
148 }
149 IPaddress addrReal;
150 if (!sock->GetLocal(addrReal)){
151     txtRx->AppendText(wxString::Format(wxT("ERROR: couldn't get the address we
152         bound to. \n")));
153 }
154 else{
155     txtRx->AppendText(wxString::Format(wxT("Server listening at %s:%u \n"),
156         addrReal.IPAddress(), addrReal.Service())));
157 }
158 // Setup the event handler and subscribe to connection events
159 sock->SetEventHandler( *this, SERVER_ID);
160 sock->SetNotify(wxSOCKET_CONNECTION_FLAG);
161 sock->Notify(true);
162 numClients = 0;
163 }
164
165 MyFrame::~MyFrame()
166 {
167     // No delayed deletion here, as the frame is dying anyway
168     delete sock;
169 }
170
171 // event handlers
172 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
173 {
174     // true is to force the frame to close
175     Close(true);
176 }
177
```

```
178 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
179 {
180     wxMessageBox(wxString::Format
181 (
182     "wxWidgets TCP server sample\n"
183     "\n"
184     "Author: Yan Naing Aye \n"
185     "Web: http://cool-emerald.blogspot.com"
186 ),
187     "About wxWidgets TCP server sample",
188     wxOK | wxICON_INFORMATION,
189     this);
190 }
191
192 void MyFrame::OnServerEvent(wxSocketEvent& event)
193 {
194     txtRx->AppendText(wxT("OnServerEvent: "));
195     wxSocketBase *sockBase;
196
197     switch (event.GetSocketEvent())
198     {
199         case wxSOCKET_CONNECTION: txtRx->AppendText(wxT("wxSOCKET_CONNECTION\n"));
200             break;
201         default: txtRx->AppendText(wxT("Unexpected event !\n")); break;
202     }
203
204     // Accept new connection if there is one in the pending
205     // connections queue, else exit. We use Accept(false) for
206     // non-blocking accept (although if we got here, there
207     // should ALWAYS be a pending connection).
208
209     sockBase = sock->Accept(false);
210
211     if (sockBase)
212     {
213         IPaddress addr;
```

```
213     if (!sockBase->GetPeer(addr))
214     {
215         txtRx->AppendText(wxT("New connection from unknown client accepted.\n"))
216     }
217     else
218     {
219         txtRx->AppendText(wxString::Format(wxT("New client connection from %s:%
220 u accepted \n")),
221             addr.IPAddress(), addr.Service()));
222     }
223 }
224 else
225 {
226     txtRx->AppendText(wxT("Error: couldn't accept a new connection \n"));
227     return;
228 }
229
230 sockBase->SetEventHandler(*this, SOCKET_ID);
231 sockBase->SetNotify(wxSOCKET_INPUT_FLAG | wxSOCKET_LOST_FLAG);
232 sockBase->Notify(true);
233
234 numClients++;
235 SetStatusText(wxString::Format(wxT("%d clients connected"), numClients), 1)
236 ;
237 }
238
239 void MyFrame::OnSocketEvent(wxSocketEvent& event)
240 {
241     txtRx->AppendText(wxT("OnSocketEvent: "));
242     wxSocketBase *sockBase = event.GetSocket();
243
244     // First, print a message
245     switch (event.GetSocketEvent())
246     {
247         case wxSOCKET_INPUT: txtRx->AppendText(wxT("wxSOCKET_INPUT\n")); break;
```

```

246     case wxSOCKET_LOST: txtRx->AppendText(wxT("wxSOCKET_LOST\n")); break;
247     default: txtRx->AppendText(wxT("Unexpected event !\n")); break;
248   }
249
250   // Now we process the event
251   switch (event.GetSocketEvent())
252   {
253     case wxSOCKET_INPUT:
254     {
255       // We disable input events, so that the test doesn't trigger
256       // wxSocketEvent again.
257       sockBase->SetNotify(wxSOCKET_LOST_FLAG);
258
259       // Receive data from socket and send it back. We will first
260       // get a byte with the buffer size, so we can specify the
261       // exact size and use the wxSOCKET_WAITALL flag. Also, we
262       // disabled input events so we won't have unwanted reentrance.
263       // This way we can avoid the infamous wxSOCKET_BLOCK flag.
264
265       sockBase->SetFlags(wxSOCKET_WAITALL);
266
267       // Read the size @ first byte
268       unsigned char len;
269       sockBase->Read(&len, 1);
270       char buf[256];
271       // Read the message
272       wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
273       if (!lenRd) {
274         txtRx->AppendText(wxT("Failed to read message.\n"));
275         return;
276       }
277       else {
278         txtRx->AppendText(wxString::Format(wxT("Read %d bytes.\n"), lenRd));
279       }
280
281       txtRx->AppendText(wxString::Format(wxT("Rx: %s \n"), wxString::FromUTF8(

```

```
buf, len));  
282  
283     // Write it back  
284     len = 2;  
285     buf[0] = 'O';  
286     buf[1] = 'K';  
287     sockBase->Write(&len,1);  
288     sockBase->Write(buf, len);  
289     txtRx->AppendText("Tx: " + wxString::From8BitData(buf, len) + "\n");  
290     // Enable input events again.  
291     sockBase->SetNotify(wxSOCKET_LOST_FLAG | wxSOCKET_INPUT_FLAG);  
292     break;  
293 }  
294 case wxSOCKET_LOST:  
295 {  
296     numClients--;  
297  
298     // Destroy() should be used instead of delete wherever possible,  
299     // due to the fact that wxSocket uses 'delayed events' (see the  
300     // documentation for wxPostEvent) and we don't want an event to  
301     // arrive to the event handler (the frame, here) after the socket  
302     // has been deleted. Also, we might be doing some other thing with  
303     // the socket at the same time; for example, we might be in the  
304     // middle of a test or something. Destroy() takes care of all  
305     // this for us.  
306  
307     txtRx->AppendText(wxT("Deleting socket.\n"));  
308     sockBase->Destroy();  
309     break;  
310 }  
311 default:  
312 }  
313  
314 SetStatusText(wxString::Format(wxT("%d clients connected"), numClients),  
315 1);  
315 }
```

---

### စာရင်း ၉.J: ce\_wx\_tcp\_server.cpp

အစ frame constructor မှာ socket server တစ်ခု ကို အောက်က အတိုင်း သတ်မှတ် လိုပဲ port number နဲ့ ဖန်တီးပြီး၊ ဆက်သွယ်မှု တစ်ခု ရောက်လာရင် လုပ်ဆောင်ဖို့ event handler ကို သတ်မှတ် နိုင်ပါတယ်။

```
// Create the address - defaults to localhost:0 initially
IPaddress addr;
addr.AnyAddress();
addr.Service(3000);

// Create the socket
sock = new wxSocketServer(addr);

// Setup the event handler and subscribe to connection events
sock->SetEventHandler( *this, SERVER_ID );
sock->SetNotify(wxSOCKET_CONNECTION_FLAG);
sock->Notify(true);
```

OnServerEvent မှာ ရောက်လာတဲ့ ဆက်သွယ်မှု ကို လက်ခံပြီး ရင် ရလာတဲ့ socket အတွက် ဒေတာတွေ ဝင်လာတဲ့ အခါ ဒါမှမဟုတ် ဆက်သွယ်မှု ပြတ်တောက် သွားတဲ့ အခါ လုပ်ဆောင် ဖို့ event handler တွေကို သတ်မှတ် ပါမယ်။ နောက်ပြီး စုစုပေါင်း client အရေအတွက် ကို ဖော်ပြနိုင် ပါတယ်။

```
wxSocketBase *sockBase;
sockBase = sock->Accept(false);

sockBase->SetEventHandler( *this, SOCKET_ID );
sockBase->SetNotify(wxSOCKET_INPUT_FLAG | wxSOCKET_LOST_FLAG);
sockBase->Notify(true);

numClients++;
SetStatusText(wxString::Format(wxT("%d clients connected"), numClients), 1);
```

ဒေတာ တွေ ဝင်လာရင် OnSocketEvent နဲ့ လက်ခံ ရယူ တဲ့ ပုံစံ က လက်ရှိ socket ရဲ့ setting တွေပေါ် မူတည်ပြီး အမျိုးမျိုး ဖြစ်နိုင် ပါတယ်။ Setting တွေ သတ်မှတ် တာ မမှန်ရင် ပရိုဂရမ် ရပ်သွား နိုင်တာကြောင့် မှန်မှန် ကန်ကန် သတ်မှတ်ဖို့ အရေးကြီး ပြီး၊ အဲဒီ အကြောင်း အသေးစိတ်ကို Julian Smart ရဲ့ Cross-Platform GUI Programming with wxWidgets [SH06] ဆိုတဲ့ စာအုပ် အဆိုဒ် ၁၈ မှာ ဖော်ပြု ထားတာကို ဖတ်ကြည့် သင့်ပါတယ်။

ဒီ နမူနာ မှာတော့ ပထမ ဆုံး byte မှာ message ရဲ့ အရွယ် အစား ကို ပို့ပေးဖို့လိုပြီး၊ အဲဒီ အရေ အတွက် မရ မချင်း စောင့်ပြီး ဖတ်တဲ့ wxSOCKET\_WAITALL ကို အသုံးပြု ထား ပါတယ်။

```
sockBase->SetFlags(wxSOCKET_WAITALL);

// Read the size @ first byte
unsigned char len;
sockBase->Read(&len, 1);

char buf[256];
// Read the message
wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
```

ဒေတာ တွေ လက်ခံ ရရှိပြီးတဲ့ အခါ OK ဆိုတဲ့ message ကို သူရဲ့ အရွယ် အစား 2 ကို ရှုံးဆုံး byte မှာ ထည့်ပြီး client ဆိုကို အကြောင်း ပြန်ပါ မယ်။

```
len = 2;
buf[0] = 'O';
buf[1] = 'K';
sockBase->Write(&len, 1);
sockBase->Write(buf, len);
```

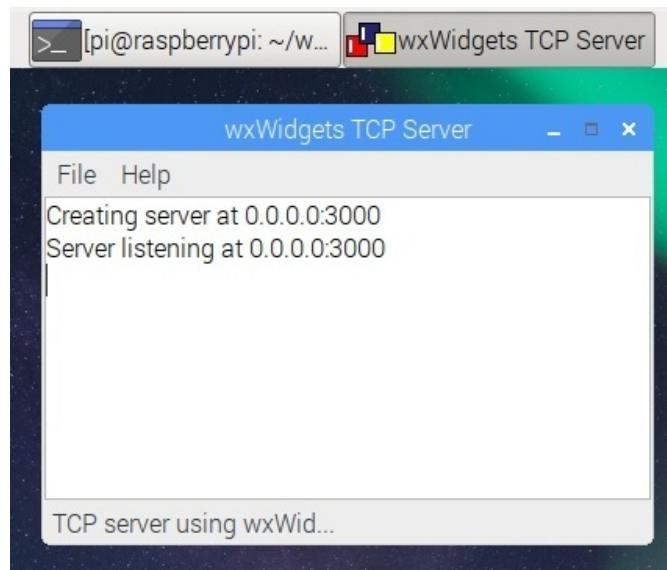
Connection ကို ဖြတ်တောက် လိုက်တဲ့ အခါ မှာ wxSOCKET\_LOST ဆိုတဲ့ OnSocketEvent ဖြစ်တဲ့ အခါမှာ ပြတ်တောက် သွားတဲ့ socket ကို ဖျက်လိုက် ပါမယ်။

```
sockBase->Destroy();
```

ပရိုဂရမ် ကို အောက်က command တွေနဲ့ build လုပ်ပြီး၊ run လိုက်တဲ့ အခါ ပုံ ၉.၃ မှာ ပြထား

သလို client ရဲ့ ဆက်သွယ်မှု ကို နားထောင် နေတာ တွေ့နိုင် ပါတယ်။

```
g++ ce_wx_tcp_server.cpp `wx-config --cxxflags --libs` -o ce_wx_tcp_server
gksudo ./ce_wx_tcp_server
```



ပုံ ၉၃: TCP server

### ၉.၂.၂ TCP Client

TCP client က လိပ်စာ တစ်ခု က port number တစ်ခု မှာ နားထောင် နေတဲ့ server ကို သွားရောက် ဆက်သွယ်ပြီး၊ ဒေတာ တွေ အပြန်အလှန် ပို့နိုင် ပါတယ်။ Connection တစ်ခု ကို ပြုလုပ်ပြီး ဒေတာ တွေကို ပိုပေး၊ server က ပြန်ပို တာကို လက်ခံ ဖော်ပြ ပေးတဲ့ [ce\\_wx\\_tcp\\_client.cpp](#) ဆိုတဲ့ TCP client နမူနာ [Gar99] တစ်ခု ကို စာရင်း ၉၃ မှာ ဖော်ပြထား ပါတယ်။

```
1 // File: ce_wx_tcp_client.cpp
2 // Description: A simple wxWidgets TCP client sample
3 // Author: Yan Naing Aye
4 // Web: http://cool-emerald.blogspot.com
5 // MIT License (https://opensource.org/licenses/MIT)
6 // Copyright (c) 2018 Yan Naing Aye
7 // References
```

```
8 // [1] Guillermo Rodriguez Garcia, "Client for wxSocket demo,"  
9 //     https://github.com/wxWidgets/wxWidgets/blob/master/samples/sockets/  
10 //      client.cpp, 1999.  
11 // [2] Julian Smart and Kevin Hock, "Cross-Platform GUI Programming with  
12 //      wxWidgets,"  
13 //      Pearson Education, Inc. 2006. ISBN: 0-13-147381-6.  
14  
15  
16 #ifdef __BORLANDC__  
17 #pragma hdrstop  
18 #endif  
19  
20 #ifndef WX_PRECOMP  
21 #include "wx/wx.h"  
22 #endif  
23  
24 #include "wx/socket.h"  
25  
26 #if wxUSE_IPV6  
27 typedef wxIPV6address IPaddress;  
28 #else  
29 typedef wxIPV4address IPaddress;  
30 #endif  
31  
32 #ifndef wxHAS_IMAGES_IN_RESOURCES  
33 #include "./sample.xpm"  
34 #endif  
35  
36 class MyApp : public wxApp  
37 {  
38 public:  
39     virtual bool OnInit();  
40 };
```

```
42 // Define a new frame type: this is going to be our main frame
43 class MyFrame : public wxFrame
44 {
45 public:
46     // ctor(s)
47     MyFrame(const wxString& title);
48     ~MyFrame();
49     // event handlers (these functions should _not_ be virtual)
50     void OnQuit(wxCommandEvent& event);
51     void OnAbout(wxCommandEvent& event);
52
53     // event handlers for Socket menu
54     void OnOpenConnection(wxCommandEvent& event);
55     void OnCloseConnection(wxCommandEvent& event);
56     void OnSend(wxCommandEvent& event);
57     void OnSocketEvent(wxSocketEvent& event);
58
59     // convenience functions
60     void UpdateStatusBar();
61
62 private:
63
64     wxSocketClient *sock;
65     wxButton *btnSend;
66     wxTextCtrl *txtSend;
67     wxTextCtrl *txtRx;
68     wxMenu *fileMenu;
69     wxMenu *helpMenu;
70     // any class wishing to process wxWidgets events must use this macro
71     wxDECLARE_EVENT_TABLE();
72 };
73
74 // IDs for the controls and the menu commands
75 enum
76 {
77     ID_BTNSEND=101,
```

```
78 ID_TXTSEND ,
79 ID_TXTRX ,
80 SOCKET_ID ,
81 CLIENT_OPEN=wxID_OPEN ,
82 CLIENT_CLOSE=wxID_CLOSE ,
83 // menu items
84 Minimal_Quit = wxID_EXIT ,
85 Minimal_About = wxID_ABOUT
86 };
87
88 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
89 EVT_MENU(Minimal_Quit, MyFrame::OnQuit)
90 EVT_MENU(Minimal_About, MyFrame::OnAbout)
91 EVT_SOCKET(SOCKET_ID, MyFrame::OnSocketEvent)
92 EVT_MENU(CLIENT_OPEN, MyFrame::OnOpenConnection)
93 EVT_MENU(CLIENT_CLOSE, MyFrame::OnCloseConnection)
94 wxEND_EVENT_TABLE()
95
96 IMPLEMENT_APP(MyApp)
97
98 // 'Main program'
99 bool MyApp::OnInit()
100 {
101     if (!wxApp::OnInit())
102         return false;
103
104     // create the main application window
105     MyFrame *frame = new MyFrame("wxWidgets TCP Client");
106
107     frame->Show(true);
108     return true;
109 }
110
111 // frame constructor
112 MyFrame::MyFrame(const wxString& title)
113     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(390, 280)
```

```
114         , wxDEFAULT_FRAME_STYLE ^ wxRESIZE_BORDER)  
115     {  
116         // set the frame icon  
117         SetIcon(wxICON(sample));  
118  
119 #if wxUSE_MENUS  
120         // create a menu bar  
121         fileMenu = new wxMenu;  
122  
123         // the "About" item should be in the help menu  
124         helpMenu = new wxMenu;  
125         helpMenu->Append(Minimal_About, "&About\tF1", "Show about dialog");  
126  
127         fileMenu->Append(CLIENT_OPEN, "&Open session\tAlt-O", "Connect to server");  
128         fileMenu->Append(CLIENT_CLOSE,"&Close session\tAlt-C","Close connection");  
129         fileMenu->Append(Minimal_Quit, "E&xit\tAlt-X", "Quit this program");  
130  
131         // now append the freshly created menu to the menu bar...  
132         wxMenuBar *menuBar = new wxMenuBar();  
133         menuBar->Append(fileMenu, "&File");  
134         menuBar->Append(helpMenu, "&Help");  
135  
136         // ... and attach this menu bar to the frame  
137         SetMenuBar(menuBar);  
138 #endif // wxUSE_MENU  
139  
140 #if wxUSE_STATUSBAR  
141         // create a status bar just for fun (by default with 1 pane only)  
142         CreateStatusBar(2);  
143         SetStatusText("TCP client using wxWidgets");  
144 #endif // wxUSE_STATUSBAR  
145         btnSend = new wxButton(this, ID_BTNSEND, wxT("Send"),  
146             wxPoint(5, 5), wxSize(100, 25));  
147         txtSend = new wxTextCtrl(this, ID_TXTSEND, wxT("Hello!"),  
148             wxPoint(120, 5), wxSize(250, 25));  
149         txtRx = new wxTextCtrl(this, ID_TXTRX, wxT(""),
```

```
150     wxPoint(5, 35), wxSize(365, 125), wxTE_MULTILINE);
151
152 Connect(ID_BTNSEND, wxEVT_COMMAND_BUTTON_CLICKED,
153         wxCommandEvent::OnSend));
154
155 // Create the socket
156 sock = new wxSocketClient();
157
158 // Setup the event handler and subscribe to most events
159 sock->SetEventHandler(*this, SOCKET_ID);
160 sock->SetNotify(wxSOCKET_CONNECTION_FLAG |
161                  wxSOCKET_INPUT_FLAG |
162                  wxSOCKET_LOST_FLAG);
163 sock->Notify(true);
164 }
165
166 MyFrame::~MyFrame()
167 {
168     // No delayed deletion here, as the frame is dying anyway
169     delete sock;
170 }
171
172 // event handlers
173 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
174 {
175     // true is to force the frame to close
176     Close(true);
177 }
178
179 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
180 {
181     wxMessageBox(wxString::Format
182 (
183     "wxWidgets TCP client sample\n"
184     "\n"
185     "Author: Yan Naing Aye \n"
```

```
186     "Web: http://cool-emerald.blogspot.com"
187 ),
188     "About wxWidgets TCP client sample",
189     wxOK | wxICON_INFORMATION,
190     this);
191 }
192
193 void MyFrame::OnSend(wxCommandEvent& WXUNUSED(event))
194 {
195     wxString str = txtSend->GetValue();
196     wxCharBuffer buffer = str.ToUTF8();
197     size_t txn = str.length();
198
199     unsigned char len;
200     len = txn;
201     sock->Write(&len, 1); //send the length of the message first
202     if (sock->Write(buffer.data(), txn).LastCount() != txn)
203     {
204         txtRx->AppendText(wxT("Write error.\n"));
205         return;
206     }
207     else {
208         txtRx->AppendText("Tx: " + str + "\n");
209     }
210 }
211
212 void MyFrame::OnOpenConnection(wxCommandEvent& WXUNUSED(event))
213 {
214     // Create the address - defaults to localhost:0 initially
215     IPaddress addr;
216     //addr.AnyAddress();
217     addr.Hostname("localhost");
218     addr.Service(3000);
219     txtRx->AppendText(wxString::Format(wxT("Trying to connect to %s:%u \n"),
220         addr.IPAddress(), addr.Service())));
221 }
```

```
222 fileMenu->Enable(CLIENT_OPEN, false);
223 fileMenu->Enable(CLIENT_CLOSE, false);
224 // we connect asynchronously and will get a wxSOCKET_CONNECTION event when
225 // the connection is really established
226 //
227 // if you want to make sure that connection is established right here you
228 // could call WaitOnConnect(timeout) instead
229
230 sock->Connect(addr, false);
231
232 //update status
233 UpdateStatusBar();
234 }
235
236 void MyFrame::OnCloseConnection(wxCommandEvent& WXUNUSED(event))
237 {
238     sock->Close();
239
240 //update status
241 UpdateStatusBar();
242 }
243
244 void MyFrame::OnSocketEvent(wxSocketEvent& event)
245 {
246     txtRx->AppendText(wxT("OnSocketEvent: "));
247     wxSocketBase *sockBase = event.GetSocket();
248
249 // First, print a message
250     switch (event.GetSocketEvent())
251     {
252         case wxSOCKET_INPUT: txtRx->AppendText(wxT("wxSOCKET_INPUT\n")); break;
253         case wxSOCKET_LOST: txtRx->AppendText(wxT("wxSOCKET_LOST\n")); break;
254         case wxSOCKET_CONNECTION: txtRx->AppendText(wxT("wxSOCKET_CONNECTION\n"));
255             break;
256         default: txtRx->AppendText(wxT("Unexpected event !\n")); break;
257     }
258 }
```

```
257
258 // Now we process the event
259 switch (event.GetSocketEvent())
260 {
261 case wxSOCKET_INPUT:
262 {
263 // We disable input events, so that the test doesn't trigger
264 // wxSocketEvent again.
265 sockBase->SetNotify(wxSOCKET_LOST_FLAG);
266
267 // Receive data from socket and send it back. We will first
268 // get a byte with the buffer size, so we can specify the
269 // exact size and use the wxSOCKET_WAITALL flag. Also, we
270 // disabled input events so we won't have unwanted reentrance.
271 // This way we can avoid the infamous wxSOCKET_BLOCK flag.
272
273 sockBase->SetFlags(wxSOCKET_WAITALL);
274
275 // Read the size @ first byte
276 unsigned char len;
277 sockBase->Read(&len, 1);
278 char buf[256];
279
280 // Read the message
281 wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
282 if (!lenRd) {
283 txtRx->AppendText(wxT("Failed to read message.\n"));
284 return;
285 }
286 else {
287 txtRx->AppendText(wxString::Format(wxT("Read %d bytes.\n"), lenRd));
288 }
289
290 txtRx->AppendText(wxString::Format(wxT("Rx: %s \n"),
291 wxString::FromUTF8(buf, len)));
292
293 // Enable input events again.
294 sockBase->SetNotify(wxSOCKET_LOST_FLAG | wxSOCKET_INPUT_FLAG);
```

```

293     break;
294 }
295 default:;
296 }

297
298 //update status
299 UpdateStatusBar();
300 }

301
302 void MyFrame::UpdateStatusBar()
303 {
304     fileMenu->Enable(CLIENT_OPEN, !sock->IsConnected());
305     fileMenu->Enable(CLIENT_CLOSE, sock->IsConnected());
306     if (sock->IsConnected()) {
307         //SetStatusText(wxString::Format(wxT("%s:%u"),
308         //    addr.IPAddress(), addr.Service()), 1);
309         SetStatusText(wxString::Format(wxT("Connected")), 1);
310     }
311     else {
312         SetStatusText(wxString::Format(wxT("Not connected")), 1);
313     }
314 }
```

စာရင်း ၉.၃: ce\_wx\_tcp\_client.cpp

ပရိုဂရမ် အစမှာ wxSocketClient တစ်ခု ကို ဖန်တီးပြီး၊ သူအတွက် event handler တွေကို သတ်မှတ် ပါတယ်။

```

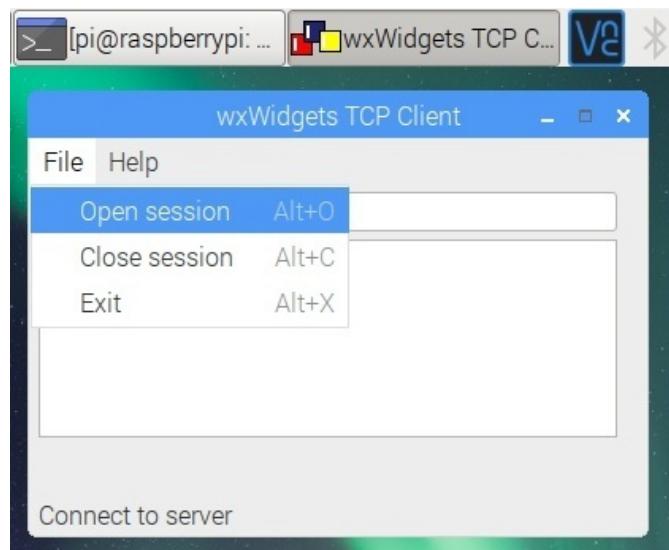
// Create the socket
sock = new wxSocketClient();

// Setup the event handler and subscribe to most events
sock->SetEventHandler( *this, SOCKET_ID );
sock->SetNotify(wxSOCKET_CONNECTION_FLAG |
wxSOCKET_INPUT_FLAG |
```

```
wxSOCKET_LOST_FLAG);
sock->Notify(true);
```

ပရိုဂရမ် ကို အောက်က command တွေ သုံးပြီး build နဲ့ run လုပ်ပြီးတဲ့ အခါ ပုံ ၉.၄ မှာ ပြထားသလို File→Open session ကို နှိပ်ပြီး အရင် အပိုင်း က run ထားတဲ့ TCP server ကို ဆက်သွယ်မှု စတင် ပြုလုပ် နိုင် ပါတယ်။ ဆက်သွယ်မှု ကို ပြန်ပိတ် ချင်ရင် တော့ Close session ကို နှိပ်နိုင် ပါတယ်။

```
g++ ce_wx_tcp_client.cpp `wx-config --cxxflags --libs` -o ce_wx_tcp_client
gksudo ./ce_wx_tcp_client
```



ပုံ ၉.၄: TCP client မှ open session ပြလုပ်ပုံ။

ဆက်သွယ်မှု ပြလုပ်ဖို့ အတွက် လိပ်စာ နဲ့ port number ထွေကို သတ်မှတ်ပြီး Connect ဆိုတဲ့ method ကို သုံးပြီး ဆက်သွယ် နိုင် ပါတယ်။

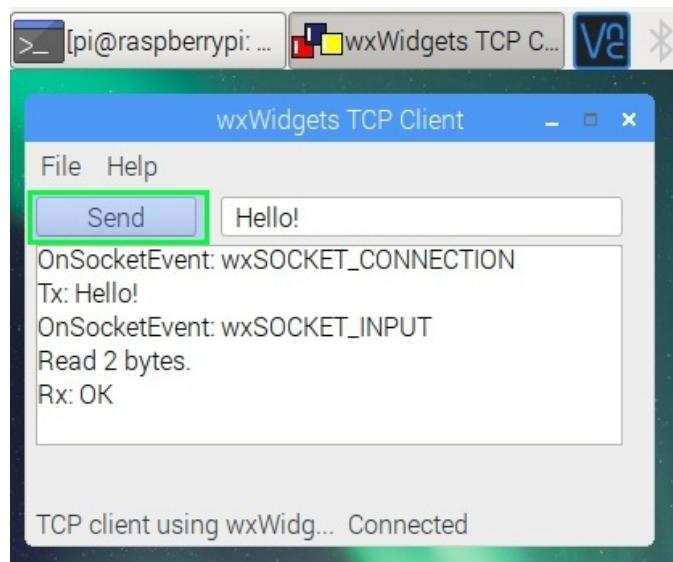
```
IPaddress addr;
addr.Hostname("localhost");
addr.Service(3000);
sock->Connect(addr, false);
```

ဆက်သွယ် မှု ပြုလုပ်ပြီး တဲ့ အခါ ဒေတာ တွေကို Send လဲထဲ နိုပ်ပြီး ပို့နိုင် ပါတယ်။ အောက်မှာ ဖော်ပြ ထားတဲ့ အတိုင်း ပို့မယ့် ဒေတာ တွေရဲ့ ပထမ byte မှာ အရော အတွက် ကို အရင်ထားပြီး buffer ထဲက ဒေတာ တွေကို နောက်က နေ Write ကိုသုံးပြီး ပို့နိုင် ပါတယ်။

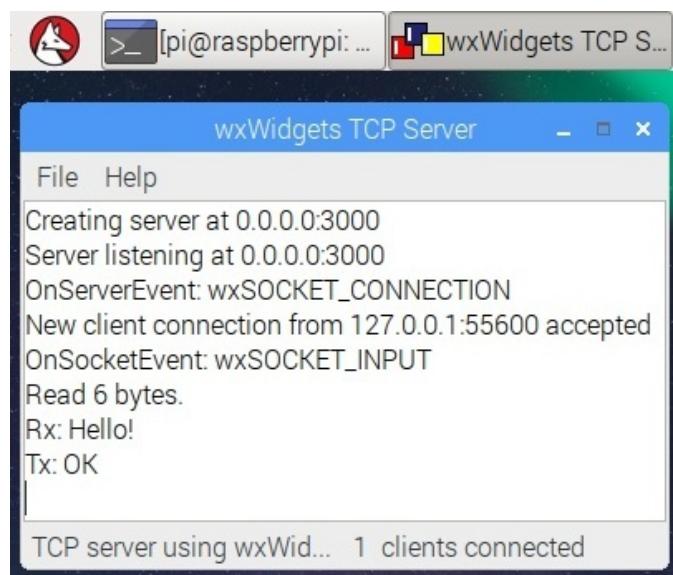
```
void MyFrame::OnSend(wxCommandEvent& WXUNUSED(event))
{
wxString str = txtSend->GetValue();
wxCharBuffer buffer = str.ToUTF8();
size_t txn = str.length();
unsigned char len;
len = txn;
sock->Write(&len, 1); //send the length of the message first
if (sock->Write(buffer.data(), txn).LastCount() != txn)
{
txtRx->AppendText(wxt("Write error.\n"));
return;
}
else {
txtRx->AppendText("Tx: " + str + "\n");
}
}
```

OnSocketEvent မှာ wxSOCKET\_INPUT ဆိုပြီး ဒေတာ တွေ လက်ခံ ရရှိတဲ့ အခါ အရင် အပိုင်းက TCP server မှာလိုပဲ Read method ကိုသုံးပြီး ဖတ်နိုင် ပါတယ်။ ဆက်သွယ်မှု ကို အဆုံးသတ်ဖို့ အတွက် File→Close session ကို နိုပ်နိုင်ပါတယ်။ ဒေတာ တွေကို ပို့ပြီး တဲ့ အခါ server က ပြန်ပို့ တာကို ဖော်ပြ ထားတာကို Client ပုံ ၉.၅ နဲ့ server ပုံ ၉.၆ မှာ တွေ့နိုင် ပါတယ်။

```
wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
```



ပုံ ၉.၅: TCP client ဖြင့် အသေးစိတ် ပို့ခြင်း နှင့် လက်ခံခြင်း။



ပုံ ၉.၆: TCP server တွင် အသေးစိတ် လက်ခံရသူ့ပုံး။

## ၉.၃ FTP Server တပ်ဆင်ခြင်း

Raspberry Pi မှာ SSH ဖွင့်ထား ရင် အပိုင်း ၂.၅ မှာ ပြောခဲ့ သလို SFTP နဲ့ တန်းပြီး ဆက်သွယ် အသုံးပြု လို ရပါတယ်။ အဲလို မဟုတ်ပဲ အကြောင်း အမျိုးမျိုး ကြောင့် ရှိုးရှိုး FTP server ကို တပ်ဆင် မယ် ဆိုရင်လည်း တပ်ဆင် နိုင် ပါတယ်။ FTP server အတွက် vsftpd လို ခေါ်တဲ့ Very Secure FTP Daemon ကို အသုံးပြု တဲ့ အကြောင်း ဈွေးနွေး ပါမယ် [Deb15]။ vsftpd ကို တပ်ဆင် ဖို့ အတွက် အောက်က command ကို သုံးနိုင် ပါတယ်။

```
$ sudo apt install vsftpd
```

တပ်ဆင် ပြီးတဲ့ အခါ ftp server က TCP port 21 ကို အလို အလျောက် စပြီး နားထောင် နေ ပါလိမ့် မယ်။ အဲဒါ ကို netstat နဲ့ အောက်ပါ အတိုင်း စစ်ကြည့် နိုင် ပါတယ်။

```
$ sudo netstat -npl
```

ပုံ ၉.၇ မှာ ပြထား သလို vsftpd က TCP port 21 မှာ နားထောင် နေတာ ကို တွေ့ရ မှာ ဖြစ် ပါတယ်။ tcp6 က IP version 6 ကို ဆိုလို ပြီး IP version 4 အတွက် လည်း ဘာမှ ထပ်လုပ် စရာ မလိုပဲ ထောက်ပံ့ ပါတယ်။

Proto Recv-Q Send-Q Local Address	Foreign Address	State
tcp 0 0 0.0.0.0:5900	0.0.0.0:*	LISTEN
tcp 0 0 0.0.0.0:22	0.0.0.0:*	LISTEN
tcp6 0 0 :::5900	:::*	LISTEN
tcp6 0 0 :::21	:::*	LISTEN
tcp6 0 0 :::22	:::*	LISTEN

ပုံ ၉.၇: netstat ကို သုံး၍ စစ်ခြင်း။

FTP server အတွက် home လုပ်ဖို့ ftp ဆိုတဲ့ directory ကို ဖန်တီးပြီး၊ သူကို configure လုပ်ဖို့ အတွက် /etc/vsftpd.conf ကို ပြုပြင် ပါမယ်။

```
$ mkdir ftp
```

```
$ sudo nano /etc/vsftpd.conf
```

ပုံမှန် အားဖြင့် Local user တွေ အတွက် access ပေးထား ပြီး၊ အဲဒီ အတွက် local\_enable က YES ဖြစ် နေရ ပါမယ်။ ရေးခွင့် ပေးဖို့ အတွက် write\_enable=YES ကို ထည့်ဖို့ # ကို ဖျက်ပြီး uncomment လုပ်နိုင် ပါတယ်။ Home directory ကို သတ်မှတ်ဖို့ အတွက် local\_root အတွက် path ကို သတ်မှတ် နိုင် ပါတယ်။

```
local_enable=YES
write_enable=YES
local_root=/home/pi/ftp
```

User တွေ က system တစ်ခု လုံးကို browse လုပ်နိုင် ပြီး၊ သူတို့ အတွက် access ကို home directory မှာပဲ ကန်သတ် ချင်ရင် လည်း ရပါတယ်။

```
allow_writeable_chroot=YES
chroot_local_user=YES
```

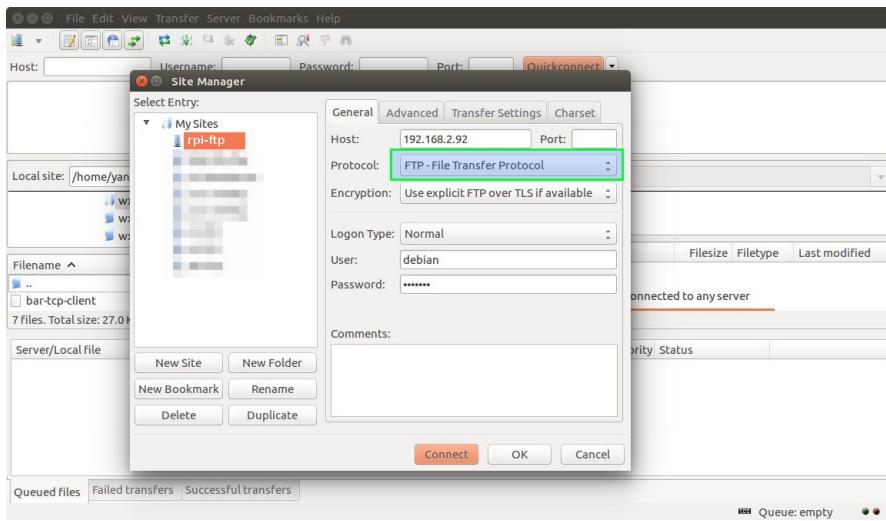
vsftpd.conf ကို စိတ်ကြိုက် ပြင်ပြီး တဲ့ အခါ service ကို restart လုပ်နိုင် ပါတယ်။

```
$ sudo service vsftpd restart
```

အဲဒီ နောက် မှာ vsftpd server ကို FileZilla စတဲ့ ftp client တွေ သုံးပြီး ဆက်သွယ် အသုံးပြု နိုင် ပါပြီ (ပုံ ၉.၈)။

## ၉.၄. WEB SERVER တပ်ဆင်ခြင်း

J77



ပုံ ၉.၈: FTP server ကို FileZilla client ဖြင့် ဆက်သွယ်ခြင်း။

## ၉.၅ Web Server တပ်ဆင်ခြင်း

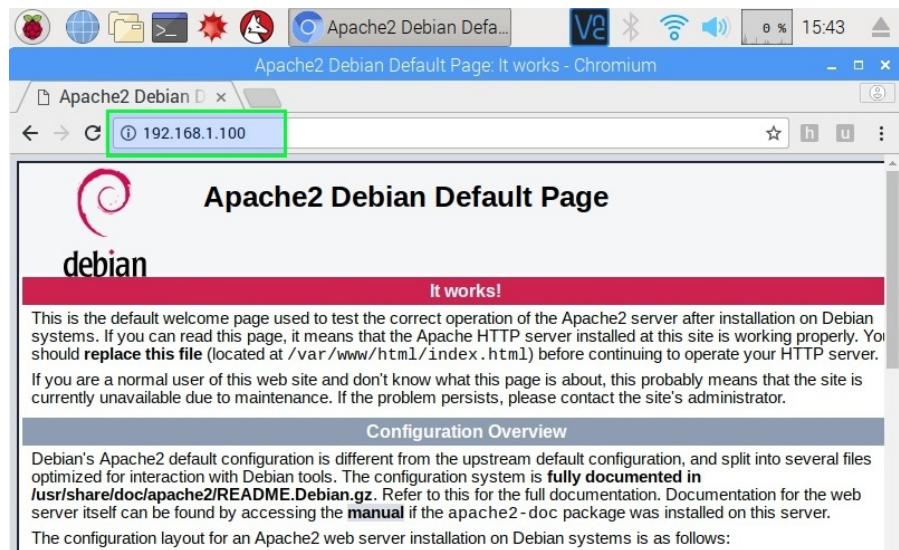
Raspberry Pi ကို တော်းပစ္စည်း၊ ကိရိယာ တွေ ကနေ ဆက်သွယ် ဖို့ အဆင်ပြေပြီး ကောင်းမွန် တဲ့ နည်းလမ်း တစ်ခု ကတော့ http ကို သုံးပြီး web browser စတာတွေ ကနေ ဆက်သွယ်တဲ့ နည်းပါ။ အဲဒီ အတွက် Apache web server ကို တပ်ဆင် အသုံးပြု တဲ့ အကြောင်း ဆွေးနွေး ချင် ပါတယ် [ras17c]။

```
$ sudo apt update  
$ sudo apt install apache2 apache2-utils  
$ sudo apt install php libapache2-mod-php
```

တပ်ဆင် ပြီးတဲ့ အခါ ထုံးစံ အတိုင်း netstat နဲ့ စစ်ကြည့် အခါ tcp port 80 မှာ apache2 နားထောင် နေတာ ကို တွေ့နိုင် ပါတယ်။

```
$ sudo netstat -npl
```

ဒါမူ မဟုတ် web browser မှာ Raspberry Pi ရဲ့ ip address ကို ရိုက်ထည့် ပြီးလည်း ကြည့်နိုင် ပါတယ် (ပုံ ၉.၉)။



ပုံ ၉.၉: Web server ကိစစ်ကြည့်ခြင်း။

### ၉.၄.၁ PHP ကို အသုံးပြုခြင်း

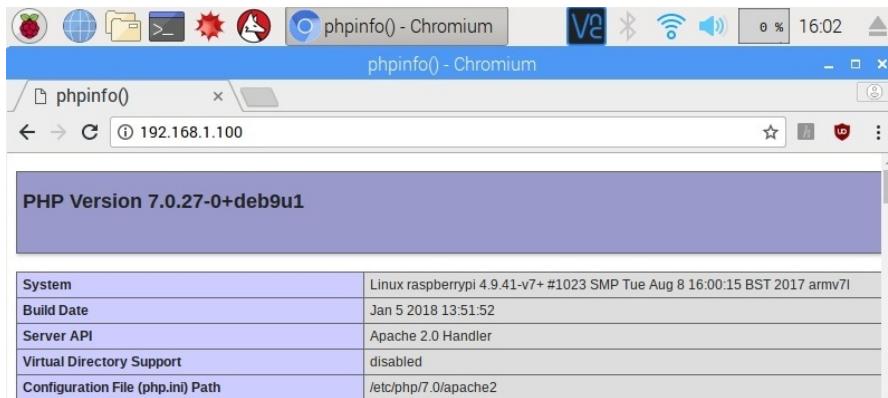
HTML အတွက် root folder ဖြစ်တဲ့ `/var/www/html` ရဲ့ ownership ကို pi ဖြစ်အောင် ပြောင်းပေးပါမယ်။ ပြီးရင် php ကို စမ်းသပ် ကြည့်ဖို့ အတွက် မူရင်း `index.html` ကို ပျက်ပြီး၊ php ဖိုင် တစ်ခု ကို ဖန်တီး ပါမယ်။

```
$ sudo chown -R pi /var/www/html
$ cd /var/www/html
$ rm index.html
$ nano index.php
```

အဲဒီ ထဲမှာ php information တွေကို ပြပေးတဲ့ ကုဒ်ကို အောက်ပါ အတိုင်း ထည့်နိုင် ပါတယ်။

```
<?php phpinfo(); ?>
```

ဖိုင်ကို သိမ်းပြီး တဲ့ အခါ web browser ကို ပြန်ဖွင့် ကြည့်လိုက် ရင် ပုံ ၉.၁၀ အတိုင်း PHP က အလုပ်လုပ်ပြီး information တွေ ပြပေး တာ ကို တွေ့ရ ပါမယ်။



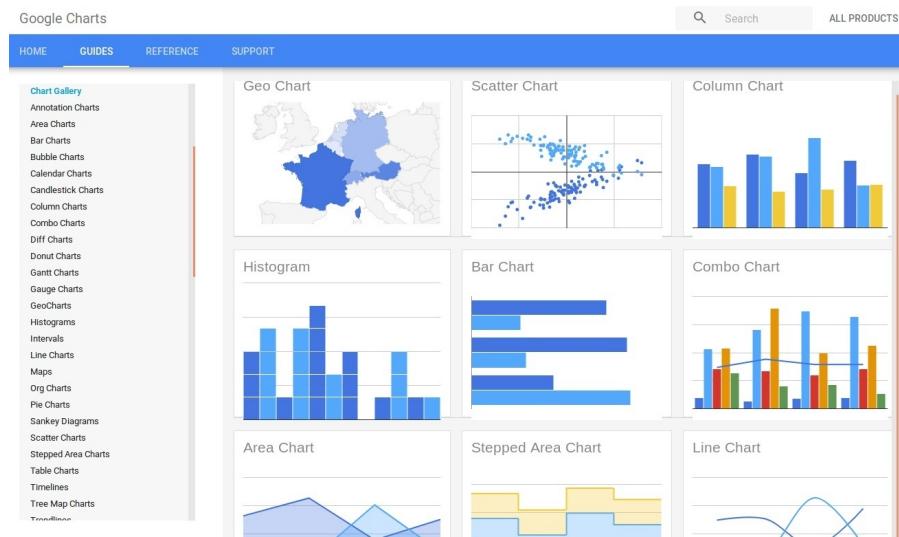
ပုံ ၉.၁၀: PHP ကိစစ်သပ်ခြင်း။

အဲဒီလို မဟုတ်ပဲ စက်ထဲ မှာ ရှိတဲ့ အခန်း တစ်ခု ခဲ့ကို root folder အနေဖြင့် သုံးချင်ရင် /var/www/ ထဲက html အခန်း တစ်ခု လုံးကို ဖျက်လိုက်ပြီး symbolic link လုပ်လို လည်း ရပါတယ်။

```
$ sudo mv /var/www/html /home/pi/html
$ sudo ln -s /home/pi/html /var/www/html
```

## ၉.၅ Google Charts

IoT application တွေမှာ သုံးထား တဲ့ sensor တွေ၊ controller တွေ၊ live data တွေကို web page ပေါ်မှာ ဖော်ပြ ဖို့ ကောင်းမွန် သင့်တော် တဲ့ ကိရိယာ တစ်ခု က Google charts ([developers.google.com/chart/](https://developers.google.com/chart/)) ပါ။ Google chart tools တွေက အစွမ်း ထက်မြက်၊ ရှိုးရှင်းရုံး မကပဲ free လည်း ဖြစ်ပါတယ်။ Data visualization အတွက် chart အမျိုး အစား အများကြီး ပါပြီး၊ HTML5/SVG သန္တသန် ပဲ သုံးထား တာမို့ ပလက်ဖောင်း အမျိုးမျိုး၊ browser အမျိုးမျိုး မှာ plugins တွေ မလိုပဲ သုံးနိုင် ပါတယ်။ နမူနာ Google chart တချို့ကို ပုံ ၉.၁၁ မှာ ပြထား ပါတယ်။



ပုံ ၉.၁၁: Google chart gallery

Google chart တစ်ခု ကို စမ်းသပ် ကြည့်ဖို့ အတွက် အောက်က စာရင်း ၉.၄ မှာ ဖော်ပြ ထားတဲ့ HTML ကုဒ်တွေ ကို gchart.htm စတဲ့ ပိုင် တစ်ခု အနေဖြင့် သိမ်းပြီး browser တစ်ခု ချုံးဖို့ကြည့် လိုက်ရင် stepped area chart တစ်ခု ကို ဖော်ပြ ပေးဘာ ကို တွေ့နှင့် ပါတယ် (ပုံ ၉.၁၂)။ Stepped area chart အစား bar chart နဲ့ ဖော်ပြ ချင်ရင် အဲဒီ ကုဒ် နမူနာ ထဲက google.visualization.SteppedAreaChart ဆိုတဲ့ နေရာမှာ google.visualization.BarChart လို ပြောင်းသုံး လိုက်ရုံး ပါဝဲ။

```

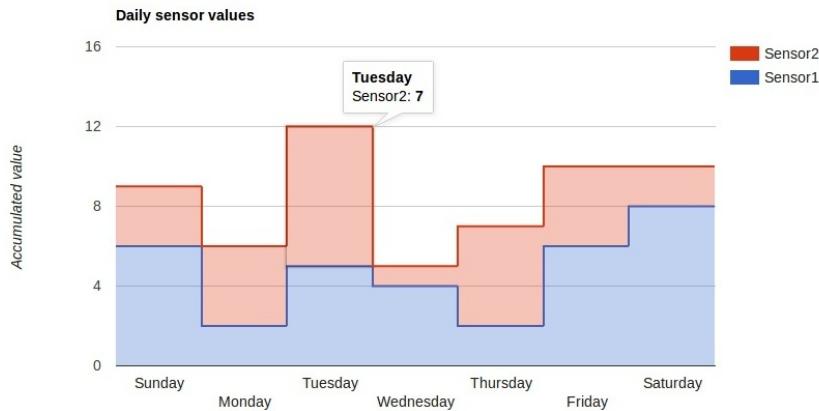
1 <html>
2   <head>
3     <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
4     <script type="text/javascript">
5       google.charts.load('current', {'packages':['corechart']});
6       google.charts.setOnLoadCallback(drawChart);
7
8       function drawChart() {
9         var data = google.visualization.arrayToDataTable([
10           ['Weekday', 'Sensor1', 'Sensor2'],
11           ['Sunday', 6, 3],
12           ['Monday', 2, 4],
13           ['Tuesday', 4, 5],
14           ['Wednesday', 3, 4],
15           ['Thursday', 5, 6],
16           ['Friday', 7, 8],
17           ['Saturday', 9, 7]
18         ]);
19
20         var options = {
21           title: 'Sensor Readings',
22           subtitle: 'Comparison of Sensor 1 and Sensor 2 over a Week',
23           x-axis: 'Weekday',
24           y-axis: 'Value',
25           steppedArea: true
26         };
27
28         var chart = new google.visualization.SteppedAreaChart(document.getElementById('chart_div'));
29         chart.draw(data, options);
30       }
31     </script>
32   </head>
33   <body>
34     <div id="chart_div" style="width: 100%; height: 400px;">
```

## 6.6. GOOGLE CHARTS

జో

```
13 ['Tuesday',5,7],
14 ['Wednesday',4,1],
15 ['Thursday',2,5],
16 ['Friday',6,4],
17 ['Saturday',8,2]
18 ]);
19
20     var options = {
21         title: 'Daily sensor values',
22         vAxis: {title: 'Accumulated value'},
23         isStacked: true
24     };
25
26     var chart = new google.visualization.SteppedAreaChart(document.
27 getElementById('chart_div'));
28     //var chart = new google.visualization.BarChart(document.
29     getElementById('chart_div'));
30
31     chart.draw(data, options);
32 }
33 </script>
34 </head>
35 <body>
36     <div id="chart_div" style="width: 900px; height: 500px;"></div>
37 </body>
38 </html>
```

ఓగ్గులు : 6.6: Stepped area chart ఫూట్ : gchart.htm



ပုံ ၉.၁: Stepped area chart တစ်ခု ဆွဲခြင်း။

### ၉.၁ Chart လိုင်ဘရိယည့်ခြင်း

Google chart ကို သုံးမယ် ဆိုရင် ဝက်တ် စာမျက်နှာ ရဲ့ head ဆိုတဲ့ အပိုင်း မှာ အောက်က ကုဒ် စာကြောင်း တွေကို ထည့်ပေး ဖို့ လိုပါတယ်။

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"
      ></script>
<script type="text/javascript">
google.charts.load('current', {packages: ['corechart']});
google.charts.setOnLoadCallback(drawChart);
...
</script>
```

ပထမ စာကြောင်း က loader ကို ထည့်ပေး တာပါ။ Chart ဘယ်နှစ်ခု ပဲ ဆွဲဆွဲ တစ်ခါပဲ ထည့်ပေး ဖို့ လိုပါတယ်။ google.charts.load ဆိုတဲ့ function က သတ်မှတ် တဲ့ chart အမျိုးအစား အတွက် packages တွေကို load လုပ်ပေး ပါတယ်။ ဒီ နူးနာ မှာတော့ bar, column, line, area, stepped area, bubble, pie, donut, combo, candlestick, histogram, scatter စားစာ တွေပါတဲ့ corechart ကို သုံးထား ပါတယ်။ ပထမ argument မှာ ထည့်လိုက်တဲ့ current ၏ latest release ကို သုံးမယ် လို ဆိုလို တာပါ။ ပြီးတဲ့ အခါ web page မှာ drawChart ဆိုတဲ့ Javascript function ရှိမယ် လို ယူဆ ထားပါတယ်။

Google chart ရဲ့ အားနည်းချက် က off line သုံးလို မရ ပါဘူး။ အကယ်၍ off line သုံးချင် ရင်တော့

Chart.js, D3.js စသာ တွေကို သုံးနှင့် ပါတယ်။

### ၉.၅.၂ ဒေတာများပြင်ဆင်ခြင်း

ဆွဲပြ ချင်တဲ့ ဒေတာ တွေကို ပေါ်လော် အနေနဲ့ အောက်က အတိုင်း ထည့်နှင့် ပါတယ်။

```
var data = google.visualization.arrayToDataTable([
  ['Weekday', 'Sensor1', 'Sensor2'],
  ['Sunday', 6, 3],
  ['Monday', 2, 4],
  ['Tuesday', 5, 7],
  ['Wednesday', 4, 1],
  ['Thursday', 2, 5],
  ['Friday', 6, 4],
  ['Saturday', 8, 2]
]);
```

အောက်က စာကြောင်း တွေကို သုံးပြီး Chart ရဲ ပုံစံ ကို စိတ်ကြိုက် ပြင်ဆင် လိုလည်း ရပါတယ်။

```
var options = {
  title: 'Daily sensor values',
  vAxis: {title: 'Accumulated value'},
  isStacked: true
};
```

### ၉.၅.၃ Chart ကို ဆွဲခြင်း

နောက်ဆုံး မှာ အောက်က စာကြောင်းတွေ အတိုင်း သုံးပြီး chart ကို ဆွဲနိုင် ပါတယ်။

```
var chart = new google.visualization.SteppedAreaChart(document.getElementById('chart_div'));
chart.draw(data, options);
```

## ၉.၆ D3.js

နောက်ထပ် ခေတ်စား တဲ့ data visualization tool တစ်ခု က D3.js ပါ။ D3.js ကို သုံးပြီး ရိုးရှင်းတဲ့ bar chart တစ်ခု ဆွဲတဲ့ နမူနာ d3bar.php ကို စာရင်း ၉.၅ မှာ ပြထား ပါတယ်။

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <!--script src="d3.min.js"></script-->
6     <script src="https://d3js.org/d3.v3.min.js"></script>
7
8     <style>
9         .chart div {
10             font: 10px sans-serif;
11             background-color: steelblue;
12             text-align: right;
13             padding: 3px;
14             margin: 1px;
15             color: white;
16         }
17
18         #dashboard {
19             width: 320px;
20             border: 1px solid black;
21         }
22
23         p {
24             text-align: center;
25         }
26     </style>
27 </head>
28 <body>
29
30 <div id="dashboard">
31     <p>Sensor values</p>
```

```

32 <div class="chart">
33
34 </div>
35 <br/>
36 </div>
37
38 <script>
39 //var data = [210, 860, 468, 681, 303, 565];
40
41 var data = [<?php
42
43 for ($x=0;$x<6;$x++) {
44 if($x!=0) echo ",";
45 $sv=shell_exec("/var/www/html/d3js/readsensor ".$x);
46 echo $sv;
47 }
48
49 //include 'myarr.php';
50
51 ?>
52 ];
53
54
55 d3.select(".chart")
56 .selectAll("div")
57 .data(data)
58 .enter()
59 .append("div")
60 .style("width", function(d) { return (d*300/860) + 'px' })
61 .text(function(d) { return d+' mA'; });
62 </script>
63
64 </body>
65 </html>

```

በዚህ የD3.js bar chart ቁጥር ነው : d3bar.php

d3js.org ကင် download လုပ်ပြီးရရှိတဲ့ဖိုင်ကို d3bar.php နဲ့ folder တစ်ခု ထဲမှာ အတူတူ ထားဖို့လိုပါတယ်။ နောက်ပြီး ဝက်ဘ် စာမျက်နှာ ဖိုင်ရဲ့ head အပိုင်း မှာ အောက်က စာကြောင်း ကို ထည့်နိုင် ပါတယ်။

```
<script src="d3.min.js"></script>
```

ဒေတာ တွေကို သတ်မှတ် ပြီး bar chart ဆွဲပြုဖို့ အတွက် script tag ထဲမှာ အောက်က ပြထား သလို ကုဒ် တွေကို သုံးနိုင် ပါတယ်။

```
var data = [210, 860, 468, 681, 303, 565];
d3.select(".chart")
.selectAll("div")
.data(data)
.enter()
.append("div")
.style("width", function(d) { return (d*300/860) + 'px' })
.text(function(d) { return d+' mA'; });
```

ဒါ နမူနာ မှာ တော့ data ဆိုတဲ့ array ကို တစ်ခါ ထည့်း အသေ ထည့် မထား ပဲ readsensor.cpp ဆိုတဲ့ စာရင်း ၉.၆ မှာ ပြထားတဲ့ C++ ပရိုဂရမ် လေး ရေးပြီး ချိတ်ဆက် အသုံးပြု လိုက်ပါမယ်။

```
1 #include <iostream>
2 #include <string>
3 #include <sstream>
4 using namespace std;
5 template <typename T>
6     T FromString(const string &Text)
7     {
8         istringstream ss(Text);
9         T result;
10        return ss >> result ? result : 0;
11    };
12
13 int main(int argc, char* argv[])
14 {
```

```

15     int data[] = {240,860,468,681,303,565};
16     string str=argv[1];
17     int v=FromString<int>(str);
18     int r=0;
19     if(v<6 && v>=0){
20         r=data[v];
21     }
22     cout<<r;
23     return 0;
24 }
```

စာရင်း ၉.၆: readsensor.cpp

အဲဒီ ပရိုဂရမ် ကို build လုပ်ဖို့ လိုချင် တဲ့ data တန်ဖိုး ရဲ့ index ကို argument အနေနဲ့ ထည့်ပြီး run ဖို့အောက်က နှမူနာ command တွေကို သုံးနိုင် ပါတယ်။ အဲဒီ မှာ argument အနေနဲ့ 1 ကို သုံးလိုက် တဲ့ အတွက် index 1 က တန်ဖိုး 860 ကို ရိုက်ပြ မှာပါ။

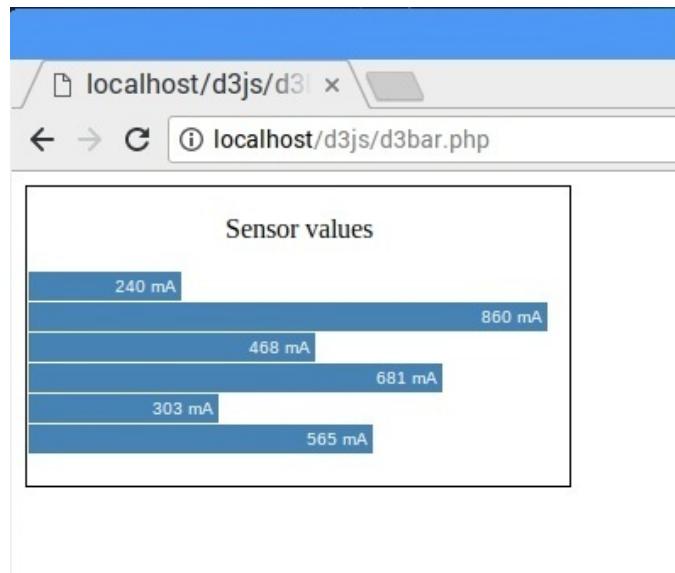
```
$ g++ readsensor.cpp -o readsensor
$ ./readsensor 1
```

Apache web server ရဲ့ home ဖြစ်တဲ့ /var/www/html ထဲမှာ ထည့်ထား တဲ့ အဲဒီ ပရိုဂရမ် ကို PHP ကနေ လုမ်းပြီး ချိတ်ဆက် အသုံးပြု တဲ့ ကုဒ် အပိုင်းအစ တရာ့ ကို အောက်က စာရင်း မှာ ပြထား ပါတယ်။

```

var data = [<?php
for ($x=0;$x<6;$x++) {
if($x!=0) echo ",";
$sv=shell_exec("/var/www/html/d3js/readsensor ".$x);
echo $sv;
}
?>
];
```

ခုနက နှမူနာ မှာ D3.js နဲ့ ဆွဲလိုက်တဲ့ bar chart ကို ပုံ ၉.၃၃ မှာ ပြထား ပါတယ်။



ပုံ ၉.၁၃: D3.js ဖြင့် bar chart တစ်ခု ဆွဲခြင်း။

ရှေ့မှာ ဖော်ပြု ခဲ့တဲ့ နည်းတွေ ကို မသုံးပဲ comma separated values တွေကို ဖိုင် တစ်ခု ထဲမှာ စုထားပြီး PHP ရဲ့ include နဲ့ တိုက်ရိုက် ထည့်သုံး မယ်ဆိုရင်လည်း အောက်က ကုဒ် လိုမျိုး အသုံးပြု နိုင် ပါတယ်။

```
var data = [<?php include 'myarr.php';?>];
```

## ၉.၇ Email ပို့ခြင်း

Raspberry Pi ကို သုံးပြီး sensor တွေ၊ ဒါမ္မ၊ မဟုတ် controller တွေ ပြေလှုပ် ပြီးတဲ့ အခါ သတ်မှတ် ထားတဲ့ အဖြစ် အပျက် တစ်ခုခု ဖြစ်ရင် administrator ဆိုကို email ပို့တဲ့ လုပ်ငန်း ကို လုပ်ချင် ရင်လည်း လုပ်နိုင် ပါတယ်။ အဲဒီ အတွက် ssmtp နဲ့ gmail ကို သုံးနိုင် ပါတယ် [Mol14; Ada15]။

```
$ sudo apt install ssmtp mailutils
```

အဲဒီလို တပ်ဆင် ပြီးတဲ့ အခါ nano ကို သုံးပြီး ssmtp ကို configure လုပ် ပါမယ်။

```
$ sudo nano /etc/ssmtp/ssmtp.conf
```

## ၉.၇. EMAIL ပို့ခြင်း

၂၈၉

အဲဒီ ssmtp.conf ဖိုင် ထဲမှာ အောက်ပါ အတိုင်း တန်ဖိုး တွေ သတ်မှတ် နိုင် ပါတယ်။

```
root=postmaster
mailhub=smtp.gmail.com:587
AuthUser=name@gmail.com
AuthPass=password
hostname=raspberrypi
rewriteDomain=gmail.com
FromLineOverride=YES
UseSTARTTLS=YES
UseTLS=YES
```

ဖိုင်ကို သိမ်းပြီး ထွက်ပြီး တဲ့ အခါ အောက်က command နဲ့ mail ပို့တာ ကို စမ်းကြည့် နိုင် ပါတယ်။

```
$ echo "Hello world email body" | mail -s "Test Subject" recipientname@domain
.com
```

အကယ်၍ email ပို့တာ မအောင်မြင်ပဲ၊ ပို့တဲ့ account ထဲကို Review blocked sign-in attempt ဆိုတဲ့ email ရောက်လာရင်၊ အဲဒီ ထဲက allowing access to less secure apps ဆိုတဲ့ link သို့ <https://myaccount.google.com/security> ကို သွားပြီး၊ Allow less secure apps: ON ဖြစ်အောင် ပြောင်းပေး ဖို့လို ပါတယ်။

## ၉.၇.၁ ဖိုင်ကိုပို့ခြင်း

ဖိုင်ကို email body အနေနဲ့ ပို့ချင် ရင်တော့ mpack ကို သုံးလို ရပါတယ်။

```
$ sudo apt install mpack
```

ဥပမာ emailbody.txt ဆိုတဲ့ ဖိုင်တစ်ခု ဖန်တီးပြီး စာတွေ ဖြည့်ပြီးရင် အောက်ပါ အတိုင်း ပို့နိုင် ပါတယ်။

```
$ mpack -s "Test file subject" /home/pi/rpi/iot/emailbody.txt
recipientname@domain.com
```

### ၉.၇.၂ C++ ဖြင့်ပို့ခြင်း

ဖော်ပြုခဲ့တဲ့ command တွေကို C++ ပရိုဂရမ် ထဲမှာ system() ကို သုံးပြီး ခေါ်သုံးလို ရပါတယ်။ နမူနာ အနေနဲ့ sendemail.cpp ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၉.၇ မှာ ပြထား ပါတယ်။

```

1 #include <iostream>
2 #include <stdlib.h>
3 #include <string>
4 using namespace std;
5 int main()
6 {
7     string cmdstr="mpack -s \"Test c++ pi email\" /home/pi/rpi/iot/emailbody.
8     txt yan9aye@gmail.com";
9     int r=system(cmdstr.c_str());
10    cout<<cmdstr<<endl;
11    cout<<"Return: "<<r<<endl;
12    return r;
13 }
```

စာရင်း ၉.၇: Email ကို C++ ပရိုဂရမ်ဖြင့် ပို့ခြင်း။

သူကို အောက်က အတိုင်း build လုပ်ပြီး run လိုက်တဲ့ အခါ email ပိုပေး တာကို တွေ့ရ ပါလိမ့်မယ်။

```
$ g++ sendemail.cpp -o sendemail
$ ./sendemail
```

### အကိုးအကားများ

- [Ada15] Adam. ssmtplib to send emails. 2015. url: [http://www.raspberry-projects.com/pi/software\\_utilities/email/ssmtp-to-send-emails](http://www.raspberry-projects.com/pi/software_utilities/email/ssmtp-to-send-emails).
- [Gar99] Guillermo Rodriguez Garcia. Client for wxSocket demo. 1999. url: <https://github.com/wxWidgets/wxWidgets/blob/master/samples/sockets/client.cpp>.

- [GZ09] Guillermo Rodriguez Garcia and Vadim Zeitlin. Server for wxSocket demo. 2009. url: <https://github.com/wxWidgets/wxWidgets/blob/master/samples/sockets/server.cpp>.
- [Mol14] Derek Molloy. Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. Wiley, 2014. isbn: 1118935128. url: <http://www.exploringbeaglebone.com/>.
- [ras17c] raspberrypi.org. Setting up an Apache web server on a Raspberry Pi. 2017. url: <https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>.
- [SH06] Julian Smart and Kevin Hock. Cross-Platform GUI Programming with wxWidgets. 1st. Pearson Education, Inc., 2006. isbn: 0-13-147381-6.
- [Deb15] Debian Wiki. Installing and configuring FTP server vsftpd. 2015. url: <https://wiki.debian.org/vsftpd>.

## အခန်း ၁၀

### အချိန်

Raspberry Pi က ဈေးသက်သက် သာသာ နဲ့ ထုတ်လုပ် ဖို့ ရည်ရွယ် ထားတာမြို့ ကွန်ပျူးတာ ကြီးတွေမှာလို ပါဝါ ပိတ်ထား တဲ့ အချိန် တွေမှာ ပြားစွဲ ဘက်ထရီ လေးကို သုံးဖြီး ဆက် အလုပ်လုပ် နိုင်တဲ့ Real Time Clock (RTC) ပါ မလာ ပါဘူး။ ဒါကြောင့် ပုံမှန် ဆိုရင် Raspberry Pi က အင်တာနက် ပေါ်က time server တွေရဲ့ အချိန်ကို ယူဖြီး ညိုယူ အသုံးပြု ပါတယ် [Ada16a]။

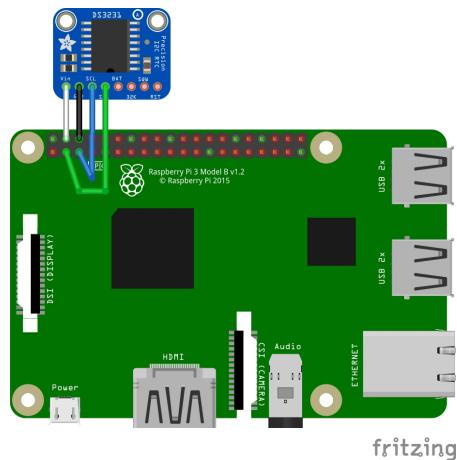
ပြဿနာ က Raspberry Pi က ပါဝါ ပိတ်ဖြီး ပြန်ပွင့် လာတဲ့ အခါ အင်တာနက် ချိတ်ဆက် မထားရင် မှန်ကန်တဲ့ အချိန်ကို မသိတော့ ပဲ နောက်ဆုံး မှတ်မိတဲ့ အချိန် ကိုပဲ ဆက်သုံး ပါတယ်။ အဲဒါကို ဖြေရှင်းဖို့ ဈေးသက်သာတဲ့ RTC လေးတွေ ကို Raspberry Pi နဲ့ ချိတ်ဆက် အသုံးပြု တဲ့ အကြောင်း ဆွေးနွေးပါမယ်။ ပြီးတဲ့ အခါ စက်တွေ အများကြီး ကို Network Time Protocol နဲ့ နာရီ အားလုံး ပြုတဲ့ ဖြစ်အောင် ချိန်ညီ တဲ့ အကြောင်း ဆက် ဆွေးနွေး ပါမယ်။

#### ၁၀.၁ Real Time Clock အသုံးပြုခြင်း

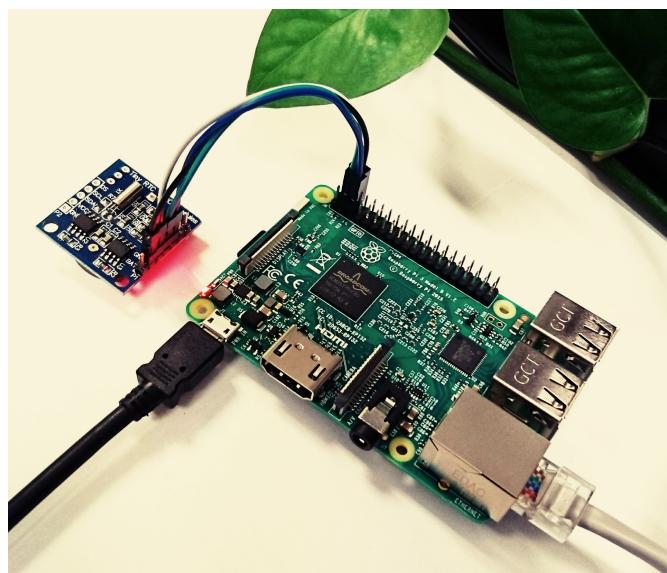
ခေတ်စား၊ အသုံးများ တဲ့ RTC တစ်ခု က DS1307 ပါ။ DS1307 RTC chip အပြင် AT24C32 ဆိုတဲ့ 32k EEPROM လေးပါ အဆစ် ပါတဲ့ RTC module လေးတွေက AliExpress မှာ ငါးမူး လောက်ပဲ ပေးရ ပါတယ်။ တခြား ဈေးပေါ်တဲ့ PCF8523 တို့၊ ပိုတိကျ ကောင်းမွန်တဲ့ DS3231 Precision RTC တို့ ကို လည်း သုံးလို ရပါတယ်။

### ၁၀.၁.၁ ဂိယာဆက်သွယ်မှု

RTC module ရဲ့ GND , SDA, SCL pin တွေကို Pi ရဲ့ GND, SDA, SCL pin တွေနဲ့ အသီးသီး ဆက်နိုင်ပါတယ်။ VCC ကို တော့ သုံးတဲ့ module အလိုက် 5V ဒါမူ မဟုတ် 3.3V နဲ့ ဆက်ဖို့ လိုပါတယ်။ DS1307 ရဲ့ နမူနာ ဆက်သွယ်မှု ကို ပုံ ၁၀.၁ နဲ့ ပုံ ၁၀.၃ မှာ ပြထား ပါတယ်။



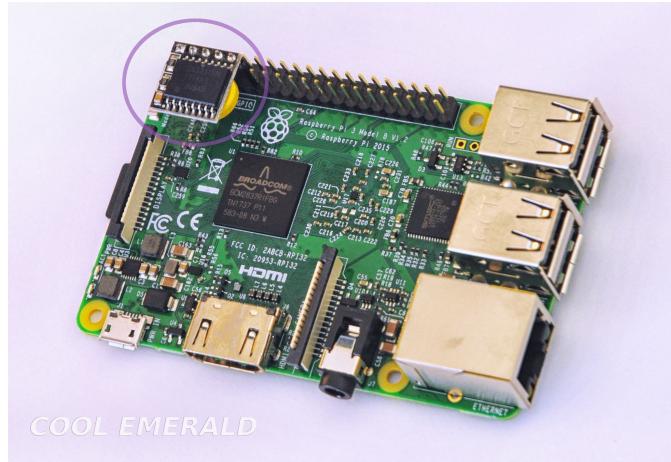
ပုံ ၁၀.၁: DS1307 RTC ကိုဆက်သွယ်ခြင်း။



ပုံ ၁၀.၂: DS1307 RTC ကို Pi နှင့် ချိတ်ဆက်ပုံ။

ပုံ မှာ ပြထား တဲ့ DS3231 Precision RTC module တို့လို Raspberry Pi အတွက် ရည်ရွယ်

ထုတ်လုပ် ထားတဲ့ module တွေ ဆိုရင် တော့ တပ်ဆင် ရတာ ပို့ပြီး လွယ်ကူ အဆင်ပြေ ပါတယ်။



ပုံ ၁၀.၃: DS3231 RTC ကို Pi တွင် တပ်ဆင်ထားပုံ။

## ၁၀.၂.၂ I2C ဆက်သွယ်မှု

RTC ကို I2C interface နဲ့ ဆက်သွယ်ဖို့ terminal မှာ

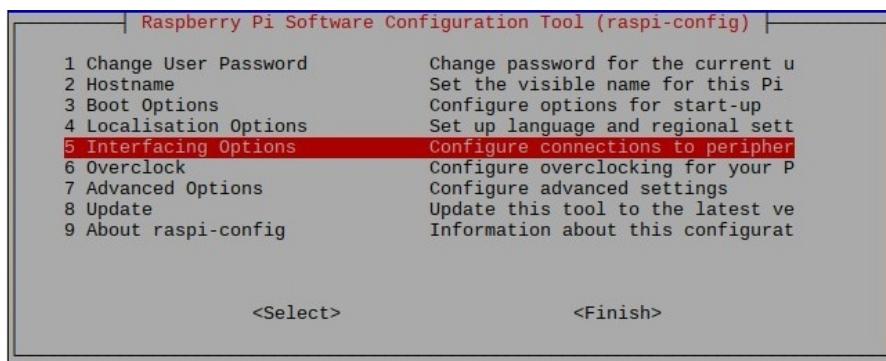
```
$ sudo raspi-config
```

ကို ရိုက်ထည့် ပါမယ်။ အဲဒီ အခါ ပေါ်လာတဲ့ configuration tool မှာ Interfacing Options ကို သွားပြီး I2C ကို Enable လုပ်ဖို့ လိုပါတယ်။ ပြီးရင် Pi ကို reboot လုပ်လိုက် ပါမယ်။ အဲဒီ အတွက် အောက်က command ကို သုံးလို့ ရပါတယ်။

```
$ sudo reboot
```

## ၁၀.၁. REAL TIME CLOCK အသုံးပြုခြင်း

၂၉၅

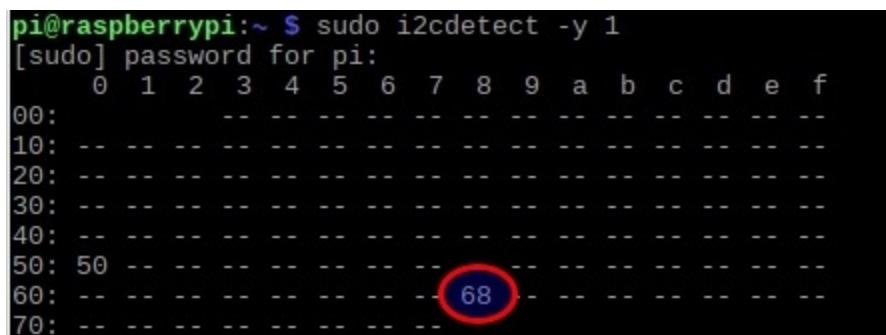


ပုံ ၁၀.၁: Raspi-config tool တွင် interfacing options ရှိ I2C ကို enable လုပ်ခြင်း။

အဲဒီနောက် terminal မှာ

```
$ sudo i2cdetect -y 1
```

ကို ရိုက်ထည့် ပြီး i2c ကို scan လုပ်ကြည့် တဲ့ အခါ ပုံ ၁၀.၃ မှာ တွေ့ရ သလို RTC ရဲ့ address 68  
ပေါ်လာတာကို တွေ့ရပါလိမ့် မယ်။ အဲဒါ ဆို RTC Module က ဝါယာ ချိတ်ဆက် မှု မှန်ကန် ပြီး I2C  
ဆက်သွယ်မှု က အဆင်ပြေ တယ်လို့ သိနိုင် ပါတယ်။



ပုံ ၁၀.၃: I2C ကို scan လုပ်ခြင်း။

DS1307 ရဲ့ register address 0 ကနေ 7 အထိက second, minute, hour, weekday, day, month, year အသီးသီး ဖြစ်တာမူး address 0 က second တန်ဖိုး 0x00 ကနေ 0x59 ထိ ပြောင်းလဲ  
နေတာကို အောက်က အတိုင်း ဖတ်ကြည့် နိုင် ပါတယ်။

```
$ i2cget -y 1 0x68 0x00
```

## ၁၀.၁.၃ RTC ကို setup လုပ်ခြင်း

နောက် အဆင့် အနေဖြင့် RTC အထောက် အပံ့ ကို စက်မှာ device tree overlay သုံးပြီး ထည့်ဖို့ /boot/config.txt ကို အောက်က command သုံးပြီး ဖွင့်လိုက် ပါမယ်။

```
$ sudo nano /boot/config.txt
```

အဲဒီ ဖိုင်ရဲ နောက်ဆုံးမှာ သုံးတဲ့ RTC module အလိုက်

```
dtoverlay=i2c-rtc,ds1307
```

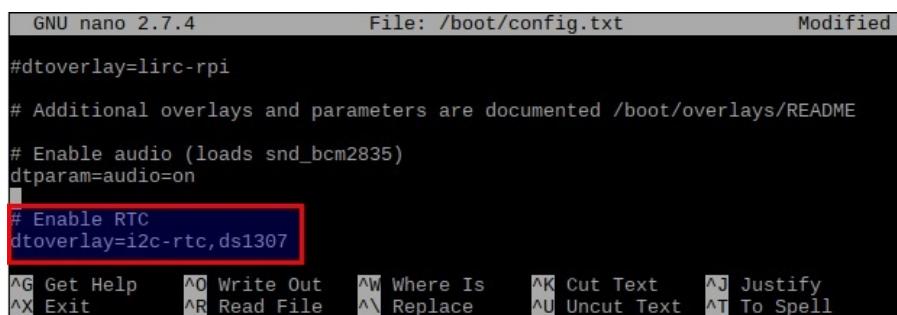
သို့မဟုတ်

```
dtoverlay=i2c-rtc,pcf8523
```

သို့မဟုတ်

```
dtoverlay=i2c-rtc,ds3231
```

ကို ထည့်ပြီး သိမ်းလိုက် ပါမယ် (ပုံ ၁၀.၆)။



```
GNU nano 2.7.4          File: /boot/config.txt          Modified
#dtoverlay=lirc-rpi
#
# Additional overlays and parameters are documented /boot/overlays/README
#
# Enable audio (loads snd_bcm2835)
dtparam=audio=on
#
# Enable RTC
#dtoverlay=i2c-rtc,ds1307
^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text    ^J Justify
^X Exit         ^R Read File    ^\ Replace      ^U Uncut Text  ^T To Spell
```

ပုံ ၁၀.၆: RTC အတွက် device tree overlay သုံးခြင်း။

အဲဒီနောက် စက်ကို reboot ပြန်လုပ်ပြီး

```
$ sudo i2cdetect -y 1
```

ကို ပြန်သုံးကြည့်တဲ့ အခါ RTC ကို device driver က အသုံးပြု ထောက်ပံ့ နေတဲ့ အတွက် 68 အစား ပူး  
လို ပြမာ ဖြစ် ပါတယ် (ပုံ ၁၀.၇)။

```
pi@raspberrypi:~ $ sudo i2cdetect -y 1
[sudo] password for pi:
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: --
10: --
20: --
30: --
40: --
50: 50 --
60: -- UU --
70: --
```

ပုံ ၁၀.၇: I2C ကို scan ပြန်လုပ်ခြင်း။

RTC အစစ် ရှိသွားပြီ ဖြစ်တဲ့ အတွက် သူကို အနောက် အယုက် မဖြစ်အောင် စက်ထဲ မှာ အရင်ရှိ နေတဲ့ RTC အတူ ကို အောက်က အတိုင်း ဖယ်ရှိနိုင် ပါတယ်(ပုံ ၁၀.၈)။

```
$ sudo apt-get -y remove fake-hwclock
$ sudo update-rc.d -f fake-hwclock remove
```

```
pi@raspberrypi:~ $ sudo apt-get -y remove fake-hwclock
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  fake-hwclock
0 upgraded, 0 newly installed, 1 to remove and 160 not upgraded.
After this operation, 32.8 kB disk space will be freed.
(Reading database ... 122753 files and directories currently installed.)
Removing fake-hwclock (0.11) ...
Processing triggers for man-db (2.7.6.1-2) ...
pi@raspberrypi:~ $ sudo update-rc.d -f fake-hwclock remove
```

ပုံ ၁၀.၈: Fake-RTC ကိုဖယ်ခြင်း။

Hardware clock အတူ အစား အစစ် ကို သုံးဖို အတွက် /lib/udev/hwclock-set ကို ပြုပြင် ပါမယ်။  
အဲဒီ အတွက်

```
$ sudo nano /lib/udev/hwclock-set
```

ကိုသုံးပြီး၊ အောက်မှာဖော်ပြထားတဲ့စာကြောင်းသုံးကြောင်းကိုcommentလုပ်ပြီးဖယ်လိုက်ပါမယ်(ပုံ၁၀.၉)။

```
#if [ -e /run/systemd/system ] ; then
#    exit 0
#endif
```

```
GNU nano 2.7.4          File: /lib/udev/hwclock-set          Modified

#!/bin/sh
# Reset the System Clock to UTC if the hardware clock from which it
# was copied by the kernel was in localtime.

dev=$1
#if [ -e /run/systemd/system ] ; then
#    exit 0
#endif
if [ -e /run/udev/hwclock-set ]; then
    exit 0
fi
```

ပုံ၁၀.၉: RTC အစစ်ကိုသတ်မှတ်ခြင်း။

Hardware clock က အချိန် ကို

```
$ sudo hwclock -D -r
```

ဆိုတဲ့ command သုံးပြီး ဖတ်လို ရပါတယ်။ ပထမ အကြိမ် စတပ် တဲ့ အချိန် မှာ hardware clock ရဲ့ အချိန်က မှားနေ့နှင့်ပါတယ်။ လက်ရှိ စက်ရဲ့ အချိန်က အင်တာနက် ချိတ်ဆက် ထားမယ်ဆုံးရင် မှန်ကန် မူရိမှာ ဖြစ်ပြီး။

```
$ date
```

ဆိုတဲ့ command သုံးပြီး စစ်ကြည့်နိုင် ပါတယ်။ ကိုယ့် ဟာကိုယ် manually အချိန်တိုက် မယ် ဆုံးရင်လည်း အောက်က အတိုင်း တိုက်နိုင် ပါတယ်။

```
$ timedatectl
$ sudo timedatectl set-ntp no
$ sudo timedatectl set-time "2017-12-27 17:15:43"
```

အချိန် တိုက်ပြီး တဲ့ အခါ လက်ရှိ စက်ထဲ က အချိန်ကို hardware clock ထဲကို ထည့်မယ် ဆိုရင် အောက်က command နဲ့ အလွယ် တကူ ရေး ထည့်နိုင် ပါတယ်(ပုံ ၁၀.၁၀)။

```
$ sudo hwclock -w
```

```
pi@raspberrypi:~ $ sudo hwclock -D -r
hwclock from util-linux 2.29.2
Using the /dev interface to the clock.
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
/dev/rtc does not have interrupt functions. Waiting in loop for time from /dev
/rtc to change
...got clock tick
Time read from Hardware Clock: 2017/12/14 09:51:13
Hw clock time : 2017/12/14 09:51:13 = 1513245073 seconds since 1969
Time since last adjustment is 1513245073 seconds
Calculated Hardware Clock drift is 0.000000 seconds
2017-12-14 17:51:12.865620+0800
pi@raspberrypi:~ $ date
Thu 14 Dec 17:53:33 +08 2017
pi@raspberrypi:~ $ sudo hwclock -w
pi@raspberrypi:~ $
```

ပုံ ၁၀.၁၀: Hardware clock ကို ရေးဖတ်ခြင်း။

အဲဒီ နောက် စက်ကို ပါဝါ ပိတ်ပြီး ပြန်ဖွင့် တဲ့ အခါ မှာလည်း RTC ကို ဖတ်ပြီး တိုက်ဆိုင် ချိန်ညို တာမို့ စက်ရဲ့ အချိန်က မှန်ကန် နေတာ ကို တွေ့ရမှာ ဖြစ်ပါတယ်။

## ၁၀.J NTP Server

Network Time Protocol (NTP) က ကွန်ပျူးတာ စနစ် တွေ ကို နာရီ တွေ ကိုက်ညီအောင် ချိန်ညို တိုက်ဖို့ အသုံးပြု တဲ့ protocol တစ်ခု ဖြစ်ပါတယ် [Wik17a]။ နောင့်နေးမှု အမျိုးမျိုး ရှိနိုင်တဲ့ packet-switched ဒေတာ network ဆက်သွယ်မှု ကို အသုံးပြု နိုင်ပါတယ်။ NTP က ကွန်ပျူးတာ တွေကို Coordinated Universal Time (UTC) ကနေ မီလီ စကြန် အနည်းငယ် လောက်ပဲ အမှား အယွင်း ရှိအောင် ချိန်ညို ပေးနိုင် ဖို့ ရည်ရွယ် ပါတယ်။

ဒါ protocol ကို client-server ပုံစံ နဲ့ ဖော်ပြလေ့ ရှိပေမယ့် တစ်ခု ကို တစ်ခု အချိန် ကိုးကား လို့ရတဲ့ peer-to-peer ပုံစံ မျိုးနဲ့ လည်း အလွယ် တကူ သုံးလို့ ရပါတယ်။ User Datagram Protocol (UDP) ကို port နံပါတ် 123 သုံးပြီး timestamp တွေ ပို့တာတို့ လက်ခံ တာတို့ ကို လုပ်ပါတယ်။ Client တွေက နားထောင် တာပဲ လုပ်တဲ့ broadcasting ဒါမှ မဟုတ် multicasting ကိုလည်းပဲ သုံးလို့ ရပါတယ်။ NTP က leap second ချိန်ညိုမှု စတာ တွေအတွက် အသိပေးချက် တွေ ပို့ပေးနိုင် ပေမယ့် local time zones

ထို့ daylight saving time တို့နဲ့ ပတ်သက် တဲ့ အချက်အလက် တွေကို တော့ မသယ်ပို့ ပါဘူး။

လက်ရှိ protocol က version 4 (NTPv4) ဖြစ်ပြီး RFC 5905 [Mil10] အနေနဲ့ မှတ်တမ်း တင် စံသတ်မှတ်ချက် အနေနဲ့ အဆို တင်သွင်း ထားပါတယ်။ သူက RFC 1305 [Mil92] မှာ သတ်မှတ် ဖော်ပြ ထားတဲ့ version 3 နဲ့လည်း backward compatible ဖြစ် ပါတယ်။

### ၁၀.၂ Basic Time Commands

#### date

ကိုယ့် စက်ရဲ အချိန်ကို ကြည့်ဖို့ အခြေခံ command တစ်ခု ဖြစ်တဲ့ [Bou17b].

```
$ date
```

ကို သုံးနိုင် ပါတယ်။ ပုံမှန် အားဖြင့် စက်တွေ က UTC time zone ဆိုပြီး ပုံ ၁၀.၁၁ မှာ ပြထား သလို ဖြစ် နေ့နှင့် ပါတယ်။

```
Terminal File Edit View Search Terminal Help
pi@raspberrypi:~ $ date
Fri Nov 17 07:28:38 UTC 2017
pi@raspberrypi:~ $
```

ပုံ ၁၀.၁၁: Output of date command

#### timedatectl

နောက်ပိုင်း ထွက်တဲ့ Debian releases တွေမှာ timedatectl ကို ntpdate [Ubu17] အစား အသုံးပြု လာကြ ပါတယ်။ ပုံမှန် အားဖြင့် timedatectl က boot လုပ်တဲ့ အချိန်ရယ်၊ socket activation ဖြစ်တဲ့ အချိန် တွေမှာ နာရီ ပြန်တိုက် လေ့ရှိပြီး၊ network ဆက်သွယ်မှု ရှိနေတဲ့ အချိန် တွေမှာ လည်း ပုံမှန် ပြန်တိုက် ပါတယ်။

အကယ်၍ စက်ထဲမှာ ntpdate ထို့ ntp တို့ တပ်ဆင် ထားတယ် ဆိုရင်တော့ timedatectl က နောက်ဆုတ် ပေးပြီး အရင် ရှိနေတဲ့ အသုံးပြုမှု ကိုပဲ ဆက်သုံး မှာပါ။ ဒါက စက်ကို upgrade လုပ်ပြီး ရင်လည်း နာရီ တိုက်တဲ့ service အချင်းချင်း ပြသေနာ မဖြစ် ဖို့ပါ။

### Time Zone

ရှိတဲ့ time zone တွေကို ကြည့်ချင်ရင် အောက်က command ကို သုံးလို ရပါတယ်။

```
$ timedatectl list-timezones
```

Time zone ကို Asia/Singapore ကို set လုပ်ပြီး confirm ပြန်လုပ်ဖို့ အောက်က command တွေ အတိုင်း သုံးနိုင် ပါတယ်။

```
$ sudo timedatectl set-timezone Asia/Singapore
$ date
```

### timesyncd

အခု နောက်ပိုင်း စက်တွေမှာ အရောင်လို ntpd client ကို မသုံးပဲ timesyncd ကပဲ အချိန် ကို ပုံမှန် စစ်ပြီး နာရီ တိုက်ပေး ပါတယ်။ လက်ရှု အချိန်နဲ့ ပတ်သက် တဲ့ timedatectl နဲ့ timesyncd အကြောင်း အခြေ အချက် အလက် တွေကို အောက်ပါ အတိုင်း စစ်ကြည့် နိုင် ပါတယ်။

```
$ timedatectl status
```

```
pi@raspberrypi:~ $ timedatectl status
    Local time: Fri 2017-11-17 15:52:14 +08
    Universal time: Fri 2017-11-17 07:52:14 UTC
        RTC time: n/a
          Time zone: Asia/Singapore (+08, +0800)
      Network time on: yes
    NTP synchronized: yes
      RTC in local TZ: no
```

ပုံ ၁၀.၁၂: Checking current status of time.

အကယ်၍ NTP ရှိနေပြီး လုပ်ဆောင် နောက် ဆိုရင် "NTP synchronized" မှာ yes လို ပြပါမယ်။ Network time on: yes ဆိုတာက တော့ timesyncd က enabled ဖြစ်နေ တယ်လို ဆိုလို တာပါ။ အကယ်၍ timesyncd က enabled ဖြစ် မနေရင် အောက်က အတိုင်း ဖွင့် ပေးနိုင် ပါတယ်။

```
$ sudo timedatectl set-ntp on
```

## ၁၀.၂.၂ ntpd ပြောင်းသုံးခြင်း

အကယ်၍ timeserver လုပ်ချင်တာ ဖြစ်ဖြစ်၊ ပိုတိကျ တဲ့ နာရီ တိုက်နည်း ကို အလို ရှိ တာပဲ ဖြစ်ဖြစ် ဆုရင် timesyncd အစား ntpd ကို ပြောင်းသုံး နှင့် ပါတယ်။ ntpd ကို မတပ်ဆင် ခင်မှာ timesyncd ကို အောက်က အတိုင်း ပိတ်နှင့် ပါတယ်။

```
$ sudo timedatectl set-ntp no
```

ပိတ် မပိတ် အောက်က အတိုင်း ပြန်စစ် နှင့် ပါတယ်။

```
$ timedatectl
```

Network time on: no လို့ပြရင် timesyncd ပိတ်သွားပြီလို့ သိနိုင် ပါတယ်။ အဲဒီ နောက် ntp package ကို apt-get သုံးပြီး အောက်က အတိုင်း တပ်ဆင် နှင့် ပါတယ်။

```
$ sudo apt install ntp
```

တပ်ဆင် ပြီးသွားရင် ntpd က အလို အလျောက် စတင် ပါတယ်။ သူ့ရဲ့ လုပ်ဆောင်မှု အဆင်ပြု မပြု status ကို အောက် က command နဲ့ စစ်နိုင် ပါတယ်။

```
$ sudo ntpq -p
```

remote	refid	st	t	when	poll	reach	delay	offset	jitter
<hr/>									
0.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.001
1.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.001
2.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.001
3.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.001
-sin.wrtpoona.in	203.123.48.219	2	u	1	64	1	2.420	1.031	2.047
-188.166.215.214	103.31.135.255	4	u	1	64	1	4.570	-4.239	2.656
+pontoon.latt.ne	199.101.100.221	3	u	3	64	1	3.593	-1.527	1.861
*139.59.240.152	242.44.235.40	2	u	2	64	1	3.274	-1.053	1.894
+ntp.sg.eria.one	202.156.0.34	2	u	1	64	1	3.245	1.943	2.736
-gandhari.thirde	193.190.230.66	2	u	2	64	1	2.937	24.230	1.790
-118.189.177.157	.PPS.	1	u	2	64	1	6.100	1.203	1.536
a.sin.pobot.net	71.80.83.115	2	u	1	64	1	2.763	9.737	2.160
128.199.91.191	140.217.111.255	5	u	1	64	1	3.602	-4.153	1.980
li1635-51.membe	211.22.103.157	3	u	1	64	1	2.633	10.003	1.493

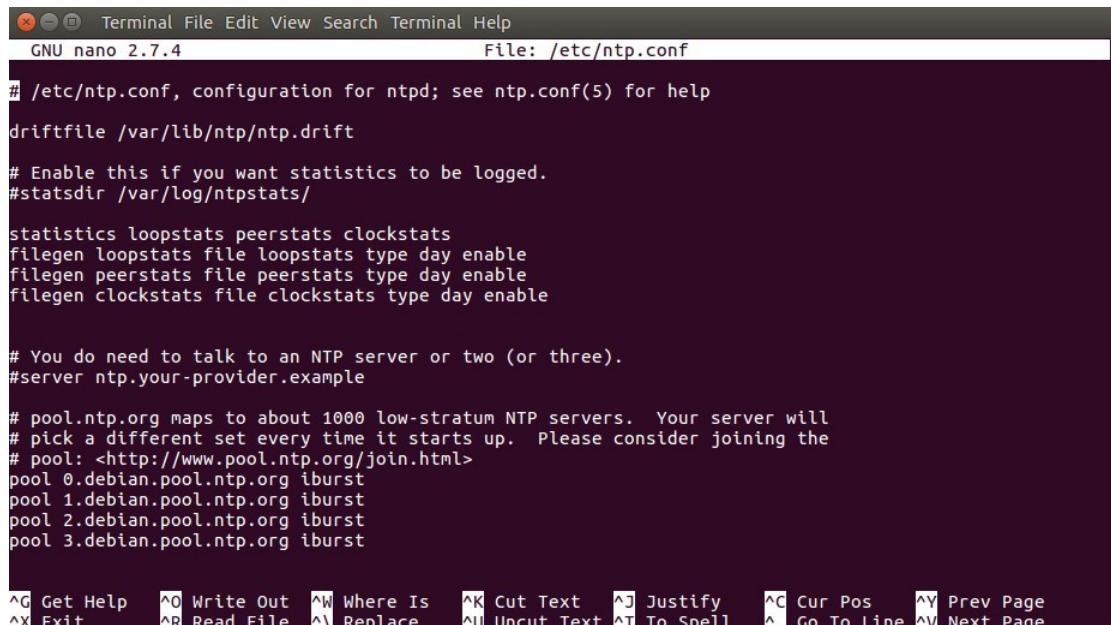
ံ ၁၀.၁၃: NTP status information.

ntpq ဆိုတာ က ntpd အတွက် query လုပ်ပေးတဲ့ ကိရိယာ တစ်ခု ဖြစ်ပါတယ်။ အဲဒီမှာ -p ဆိုတဲ့ flag က ntpd ဆက်သွယ် နေတဲ့ NTP servers ဒါမှ မဟုတ် peer တွေကို ဆိုလို ပါတယ်။ စက် တရာ့ တစ်ခု အနည်းငယ် ကွာခြား နိုင် ပါတယ်။ ပုံမှန် အားဖြင့် pool server တရာ့နဲ့ တခြား server တရာ့ ပါပါ လိမ့်မယ်။ မိနစ် အနည်းငယ် ထိ ကြာနိုင် တာကို သတိပြုဖို့ လို ပါတယ်။

### ၁၀.၂.၃ ntpd ကို ပုံစံသွင်းခြင်း

ntpd လက် ပုံစံ အနေအထား ကို သတ်မှတ်တဲ့ configuration ဖိုင်ကို /etc/ntp.conf မှာ တွေ့နိုင် ပါတယ် [mbs08]။ သူကို ပြင်ဖို့ အောက်က command ကို သုံးနိုင် ပါတယ်။

```
$ sudo nano /etc/ntp.conf
```



```
# /etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help
driftfile /var/lib/ntp/ntp.drift

# Enable this if you want statistics to be logged.
#statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

# You do need to talk to an NTP server or two (or three).
#server ntp.your-provider.example

# pool.ntp.org maps to about 1000 low-stratum NTP servers. Your server will
# pick a different set every time it starts up. Please consider joining the
# pool: <http://www.pool.ntp.org/join.html>
pool 0.debian.pool.ntp.org iburst
pool 1.debian.pool.ntp.org iburst
pool 2.debian.pool.ntp.org iburst
pool 3.debian.pool.ntp.org iburst
```

ပုံ ၁၀.၃၄: Editing /etc/ntp.conf

### Time servers

သင့် စက်ရဲ နာရီ ကို တိုက် စေချင် တဲ့ server တွေရဲ စာရင်း ကို ပြင်ဆင်၊ ထပ်ဖြည့် နိုင် ပါတယ်။

```
1 pool 0.debian.pool.ntp.org iburst
2 pool 1.debian.pool.ntp.org iburst
```

```
3 pool 2.debian.pool.ntp.org iburst
```

စာရင်း ၁၀.၁: နမူနာ time server စာရင်း

Server ရဲ့ နောက်မှာ 'iburst' ကို ထည့်ပေး ထားရင် ntpd က စက်ဖွင့် တာနဲ့ အဲဒီ server ကို အမြန် တိုက်ပါ လိမ့်မယ်။

### time server အနေဖြင့်လုပ်ဆောင်ခြင်း

ntp က အလုပ်လုပ် နေပြီး၊ အချိန် လည်း တိုက်ထား ပြီးတာနဲ့ အဲဒီ စက်ကို တွေ့ကြား စက်တွေ အတွက် server အနေနဲ့ လည်း ပြင်ဆင် နိုင် ပါတယ်။ အဲဒီ အတွက် အောက် စာကြောင်း တွေကို configuration ဖိုင် မှာ ထပ်ဖြည့် နိုင် ပါတယ်။ IP address တွေက နမူနာ သာ ဖြစ်ပြီး၊ ကိုယ့် network နဲ့ ကိုက်ညီတဲ့ တန်ဖိုးတွေကို ထည့်နိုင် ပါတယ်။

```
1 # Allow LAN machines to synchronize with this ntp server
2 restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
3 restrict 192.168.2.0 mask 255.255.255.0 nomodify notrap
```

စာရင်း ၁၀.၂: စက်ကို time server အနေဖြင့် လုပ်ဆောင်ရန် ပုံစံ သွင်းခြင်း။

Broadcast စာကြောင်း ကို ထည့်ဖို့ အောက်ကလို uncomment လုပ်နိုင် ပါတယ်။

```
broadcast 192.168.2.255
```

အင်တာနက် မရှိတဲ့ ပြတ်သွားတဲ့ အချိန်တွေ မှာ ကိုယ့်စက်ရဲ့ လက်ရှိ local time ကို သုံးပြီး တွေ့ကြား စက်တွေကို ပေးနိုင်ဖို့ အောက်က စာကြောင်း တွေကို ဖြည့်နိုင် ပါတယ်။

```
server 127.127.1.0
fudge 127.127.1.0 stratum 10
```

config ဖိုင်ကို ပြောင်း ပြီး တဲ့ အခါ ntpd ကို အောက်က အတိုင်း ပြန်စ နိုင်ပါတယ်။

```
$ sudo /etc/init.d/ntp restart
```

စက်ရဲ့ system log မှာ time server နဲ့ synchronize လုပ် မလုပ်ကို အောက်က command သုံးပြီး  
ဖြန်ကြည့် လို့ ရပါတယ်။

```
$ tail -f /var/log/syslog
```

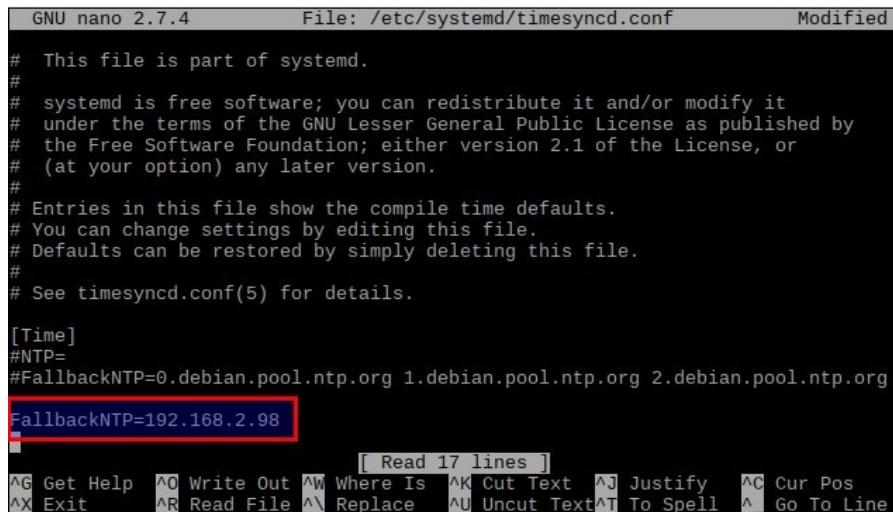
## ၁၀.၃ NTP Client

### ၁၀.၃.၁ Linux

#### timesyncd

timedatectl ကို ခုနက အသစ် တပ်ဆင် ပြင်ဆင် ထားတဲ့ time server ကို သုံးပြီး နာရီ တိုက်ဖို့ အတွက်  
`/etc/systemd/timesyncd.conf` မှာ ပြင်ဆင် အသိပေး နိုင် ပါတယ်။

```
$ sudo nano /etc/systemd/timesyncd.conf
```



```
GNU nano 2.7.4          File: /etc/systemd/timesyncd.conf          Modified
#
# This file is part of systemd.
#
# systemd is free software; you can redistribute it and/or modify it
# under the terms of the GNU Lesser General Public License as published by
# the Free Software Foundation; either version 2.1 of the License, or
# (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
#
# See timesyncd.conf(5) for details.

[Time]
#NTP=
#FallbackNTP=0.debian.pool.ntp.org 1.debian.pool.ntp.org 2.debian.pool.ntp.org
FallbackNTP=192.168.2.98

[ Read 17 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File  ^N Replace   ^U Uncut Text  ^T To Spell  ^L Go To Line
```

ပုံ ၁၀.၁၅: Configuring time server for timedatectl.

ဥပမာ time server ရဲ့ IP address ၏ 192.168.2.98 ဆိုရင် ပုံ ၁၀.၁၅ မှာ ပြထား သလို ပြင်ဆင် ပါတယ်။

```
FallbackNTP=192.168.2.98
```

ပြီးတဲ့ အခါ အဲဒီ client စက်ကို restart လုပ်ပြီး၊ သူရဲ့ system log ကို ကြည့်လိုက် ရင် သတ်မှတ်ထားတဲ့ server နဲ့ နာရီ တိုက်လိုက် တာကို တွေ့နှင့် ပါတယ်။

### ntpd

NTP client စက်ထဲမှာ ntpd တပ်ဆင် ထားတယ် ဆိုရင် time server ကို /etc/ntp.conf မှာ ပုံ ၁၀.၁၆ မှာ ပြထား သလို ထည့်ပေးနိုင် ပါတယ်။

```
server 192.168.2.98
```

```
# You do need to talk to an NTP server or two (or three).
#server ntp.your-provider.example
server 192.168.2.53
```

ပုံ ၁၀.၁၆: Configuring time server.

ပြီးရင် ntp service ကို ပြန်စပြီး status ကို ပုံ ၁၀.၁၇ ပြထား သလို ကြည့်နိုင် ပါတယ်။

```
$ sudo service ntp restart
$ sudo ntpq -p
```

remote	refid	st	t	when	poll	reach	delay	offset	jitter
192.168.2.53	210.23.25.77	2	u	3	64	1	0.606	-1.107	0.002
0.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.002
1.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.002
2.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.002
3.debian.pool.n	.POOL.	16	p	-	64	0	0.000	0.000	0.002
time3.maxonline	.GPS.	1	u	1	64	1	3.431	-3.065	0.002

ပုံ ၁၀.၁၇: Configuring time server.

ဒါမှ မဟုတ် တွေ့နိုင် ပဲ နာရီ တိုက်ဖို့ -q option နဲ့ အောက်ပါ အတိုင်း သုံးနိုင် ပါတယ်။

```
$ sudo service ntp stop
```

```
$ sudo ntpd -q 192.168.2.98
$ sudo service ntp start
```

### ntpdate

အကယ်၍ client စက်ထဲမှာ အရင် အဟောင်း ntpdate ရှိတယ် ဆိုရင်လည်း server နဲ့ နာရီ တိုက်ဖို့ အောက်က အတိုင်း ရိုက်ထည့် နိုင်ပါတယ်။

```
$ sudo service ntp stop
$ sudo ntpdate -u 192.168.2.98
$ sudo service ntp start
```

### အချိန်ကို manual တိုက်ခြင်း

လက်ရှိ စက်ရဲ အချိန်ကို ကိုယ်တိုင် ကိုယ့်ဟာကိုယ် ပြင် မယ် ဆိုရင် လည်း အောက်က အတိုင်း date ကို သုံးပြီး ပြင်နိုင် ပါတယ်။ ပြီးတဲ့ အခါ hardware clock ကို ပါ ရေးနိုင်၊ ဖတ်နိုင် ပါတယ်။

```
$ date +%Y%m%d -s "20180614"
$ date +%T -s "11:39:40"
$ sudo hwclock -w
$ sudo hwclock -r
```

နောက် တစ်နည်း အနေနဲ့ hardware clork ကို အရင် ပြင်ပြီး အဲဒီ တန်ဖိုး ကို system အတွက် ပြန် သတ်မှတ် ရင်လည်း ရ ပါတယ်။

```
$ sudo hwclock --set --date "2018-06-14 11:47:45"
$ sudo hwclock -s
```

## ၁၀.၂.J Windows

Windows စက်တွေ ကတေသ့ NTP ကို ရှိရှင်း အောင် လျော့ချ ထားတဲ့ Simple Network Time Protocol (SNTP) ဆိုတာကို သုံးပြီး NTP server တွေနဲ့ နာရီ တိုက်နိုင် ပါတယ်။ အဲဒီ အတွက် အချိန် ပြနေ တဲ့ အပေါ်မှာ ကလစ် နှစ်ချက် နိုဝင်း "Internet Time" ဆိုတဲ့ tab ကိုဖွင့် ပါမယ်။ ပြီးရင် Server ဆိုတဲ့ နေရာ မှာ ခုန် က လုပ်ထားတဲ့ server ရဲ့ IP address ကို ဖြည့်နိုင် ပါတယ်။

## ၁၀.၄ Watchdog

ဂွန်ပျူးတာ စံနစ် တွေ မှားယွင်း၊ ရပ်တန်းမှူး မရှိ အောင် Watchdog timer တွေကို အသုံးပြုကြ ပါတယ်။ Watchdog timer ကို အချိန် မလွန်ခင် အချိန်မှန် သွားသွား reset လုပ်ပေး နေဖို့ လိုပါ တယ်။ အဲဒီ ကို watchdog ကို feed ဒါမှ မဟုတ် pat လုပ်တယ် လိုလည်း ခေါ်ပါတယ်။ သတ်မှတ် ထားတဲ့ အချိန် အတွင်း feed လုပ်ဖို့ ပျက်ကွက် ခဲ့ရင် တစ်ခုခဲ့ မှားယွင်း နေဖြီ လို ယူဆပြီး၊ အဲဒီ စံနစ်ကို watchdog ကနေ reboot ပြန်လုပ် ပေးမှာ ဖြစ်ပါတယ်။ ဥပမာ ဂွန်ပျူးတာ hang သွားတယ် ဆိုရင် reset ခလုတ် ကို ဘယ်သူမှ သွားနိုပ် စရာ မလိုပဲ သူ အလိုလို reboot ပြန်ဖြစ် သွားမှာ ဖြစ် ပါတယ်။

Watchdog ကို စတင် ဖို့ အတွက် /dev/watchdog ကို ဖွင့် နိုင် ပါတယ် [Kun10; Mol16]။ သူကို စတင်တဲ့ နမူနာ C ကုဒ် အပိုင်းအစ တစ်ခုခဲ့ ကို အောက်မှာ ပြထား ပါတယ်။

```
#define WATCHDOG "/dev/watchdog"
fd = open(WATCHDOG, O_RDWR);
```

Watchdog timer အချိန် ပြည့် သွားမယ့် timeout စကြော် တန်ဖိုး ကို အောက်က အတိုင်း သတ်မှတ်လို ရပါ တယ်။

```
ioctl(fd, WDIOC_SETTIMEOUT, &interval);
```

အဲဒီ ဖိုင်ကို ဖွင့်ပြီး တာနဲ့ watchdog ကို ပုံမှန် feed လုပ်နေ ဖို့ IOCTL ကိုသုံးပြီး WDIOC\_KEEPALIVE ဆိုတဲ့ တန်ဖိုး ကို ထည့်ပေး နေလို ရပါတယ်။ မဟုတ်ရင် system က reboot ဖြစ်သွား မှာပါ။ အဲလို မဟုတ်ပဲ နောက်ထပ် ပုံမှန် feed လုပ်လို ရတဲ့ နည်းကတေသ့ /dev/watchdog ဆိုတဲ့ ဖိုင်ထဲမှာ 'V' မဟုတ် တဲ့ character တစ်ခုခဲ့ ကို ရေးပေး တာပါ။

```
r=ioctl(fd, WDIOC_KEEPALIVE, NULL);
```

Watchdog ကို ရပ်ချင် ရင် 'V' ဆိုတဲ့ character ကို /dev/watchdog မှာ ရေးပြီး၊ ဖိုင်ကို ပိတ်နှင့် ပါတယ်။ Kernel configuration မှာ CONFIG\_WATCHDOG\_NOWAYOUT ကို enabled လုပ်ထား ရင် ဘေး watchdog ကို စတင် ပြီးရင် ပြန်ရပ် လို့ မရ ပါဘူး။

```
write(fd, "V", 1);
r=close(fd);
```

Watchdog ကို သုံးတဲ့ နမူနာ class တစ်ခု အနေနဲ့ ce\_watchdog.h ကို စာရင်း ၁၀.၃ မှာ ပြထား ပါတယ်။ အဲဒီ class ကို အသုံးပြု ပုံ နမူနာ watchdog.cpp ကို စာရင်း ၁၀.၄ မှာ တွေ့နိုင် ပါတယ်။

```
1 #ifndef CE_WATCHDOG_H
2 #define CE_WATCHDOG_H
3
4 //References
5 //https://github.com/derekmolloy/exploringrpি/tree/master/chp12/watchdog
6 //https://embeddedfreak.wordpress.com/2010/08/23/howto-use-linux-watchdog/
7
8 #include<stdio.h>
9 #include<fcntl.h>
10 #include<stdlib.h>
11 #include<sys/ioctl.h>
12 #include<unistd.h>
13 #include<linux/watchdog.h>
14 #define WATCHDOG "/dev/watchdog"
15
16 #define CE_WD_PRINT 0
17
18 class CE_Watchdog {
19     int fd;
20     int interval;
21 public:
22     CE_Watchdog();
23     ~CE_Watchdog();
```

```

24     int Begin();
25     int Pat();
26     int Close();
27     int GetInterval();
28     bool IsLastBootByWatchdog();
29 }
30
31 CE_Watchdog::CE_Watchdog()
32 {
33     fd=-1; //invalid handle
34     interval=15; //in seconds
35     //Begin();
36 }
37
38 int CE_Watchdog::Begin()
39 {
40     int r=-1;
41     if(fd==(-1)){
42         //if(getuid()!=0){
43             //printf("You must run this program as root.\n");
44         //}
45         if ((fd = open(WATCHDOG, O_RDWR))<0){
46             perror("Error in opening watchdog.\n");
47             return r;
48         }
49         // set the timing interval
50         if (ioctl(fd, WDIOC_SETTIMEOUT, &interval)!=0){
51             perror("Error in setting watchdog interval.\n");
52             return r;
53         }
54         r=0;
55         #if CE_WD_PRINT ==1
56         printf("Watchdog begun. \n");
57         #endif // CE_WD_PRINT
58     }
59     return r;

```

```
60 }
61
62 CE_Watchdog::~CE_Watchdog()
63 {
64     this->Close();
65 }
66
67 int CE_Watchdog::Pat()
68 {
69     int r=-1;
70     if(fd!=(-1)){
71         r=ioctl(fd, WDIOD_KEEPALIVE, NULL);
72         #if CE_WD_PRINT ==1
73             printf("Patting Watchdog. \n");
74         #endif // CE_WD_PRINT
75     }
76     return r;
77 }
78
79 int CE_Watchdog::Close()
80 {
81     int r=-1;
82     if(fd!=(-1)){
83         r=write(fd, "V", 1);
84         r=close(fd);
85         fd=-1;
86         #if CE_WD_PRINT ==1
87             printf("Closing Watchdog. \n");
88         #endif // CE_WD_PRINT
89     }
90     return r;
91 }
92
93 int CE_Watchdog::GetInterval()
94 {
95     int v=0;
```

```

96     if(fd != (-1)){
97         if (ioctl(fd, WDIOC_GETTIMEOUT, &v) != 0) {
98             perror("Error in reading watchdog interval.\n");
99             v=0;
100        }
101    }
102    return v;
103 }
104
105 /*
106 bool CE_Watchdog::IsLastBootByWatchdog()
107 {
108     int v=0;
109     if(fd != (-1)){
110         if (ioctl(fd, WDIOC_GETBOOTSTATUS, &v) != 0) {
111             perror("Error in reading boot status.\n");
112             v=0;
113        }
114    }
115    return (v!=0)?true:false;
116 }
117 */
118 #endif

```

စာရင်း ၁၀.၃: Watchdog နမူနာ class : ce\_watchdog.h

```

1 #include "ce_watchdog.h"
2 int main(){
3     int state;
4
5     if(getuid() !=0){
6         printf("This program needs elevated privileges.\n");
7         return 1;
8     }
9     else {

```

```

10     printf("This program is running on elevated privileges.\n");
11 }
12 CE_Watchdog wd;
13 wd.Begin();
14 printf("Watchdog interval: %d s \n", wd.GetInterval());
15 /*
16 if(wd.IsLastBootByWatchdog()){
17     printf("Last boot is by watchdog \n");
18 }
19 else {
20     printf("Last boot is by power-on-reset \n");
21 }
22 */
23 printf("Enter p to pat the watchdog \n");
24 printf("Enter q to quit \n");
25 do{
26     state = getchar();
27     if(state=='p'){
28         printf("pat... \n");
29         wd.Pat();
30     }
31 } while (state!='q');
32 printf("Closing the application\n");
33 wd.Close();
34 return 0;
35 }
```

စာရင်း ၁၀.၅: ce\_watchdog.h အသုံးပြု ပုံမှန် watchdog.cpp

ပရိုဂရမ် ကို build လုပ်ပြီး၊ run ဖို့ အတွက် အောက်က command တွေကို သုံးနိုင် ပါတယ်။ သူကို သုံးဖို့ elevated privileges လိုတာ မို့ ရှေ့မှာ sudo ခံပြီး run ဖို့လို ပါတယ်။

```
$ g++ watchdog.cpp -o watchdog
$ sudo ./watchdog
```

```
pi@raspberrypi:~/watchdog $ g++ watchdog.cpp -o watchdog
pi@raspberrypi:~/watchdog $ sudo ./watchdog
Watchdog interval: 15 s
Enter p to pat the watchdog
Enter q to quit
p
pat...
p
pat...
q
Closing the application
pi@raspberrypi:~/watchdog $
```

ပုံ ၁၀.၁၈: Watchdog အသုံးပြုခြင်း နမူနာ။

ပရိုဂရမ် ကို run ပြီးတဲ့ အခါ သူရဲ့ ကုဒ် ထဲမှာ သတ်မှတ် ထားတဲ့ ၁၅ စက္ကန် မကုန်ခင် ပုံမှန် p ခလုတ် ကို နှိပ် enter နှိပ် ပြီး pat လုပ်ပေး ရင် ပရိုဂရမ် က အလုပ်လုပ် နေမှာ ဖြစ်ပြီး၊ ထွက်ချင် ရင် q ခလုတ် ကို နှိပ်ပြီး ထွက်လို့ ရပါတယ်။ အဲဒါဆို ပရိုဂရမ် က watchdog ကို ရပ်ပြီး အဆုံးသတ် သွားမှာ ပါ (ပုံ ၁၀.၁၈)။ အဲလို့ q ကို နှိပ်ပြီး မထွက်ပဲ အချိန်လွှန် သွားတဲ့ အထိ p နဲ့လည်း pat မလုပ်ရင် စက်က အလိုအလျောက် reboot ဖြစ်သွားတာ ကို တွေ့နိုင် ပါတယ်။

## ၁၀.၅ Crontab ကိုသုံးခြင်း

Linux ရဲ့ cron daemon က သတ်မှတ် ပေးလိုက်တဲ့ အလုပ် တွေ ကို သတ်မှတ်ထားတဲ့ အချိန် ပေးနိုင် ပါတယ်။ Cron က အချိန်လို့ အမိပှာယ် ရတဲ့ ကို စာလုံး chronos ကို ဆိုလို တဲ့ utility ပရိုဂရမ် တစ်ခု ဖြစ်ပြီး၊ လုပ်ဆောင် ချင်တဲ့ လုပ်ငန်း တွေနဲ့ အချိန် စာရင်း ကို crontab ဆိုတဲ့ ဖိုင်မှာ သတ်မှတ် နှင့် ပါတယ်။ သူကို ခိုင်းလို့ ရတဲ့ အမြန်ဆုံး ကြိမ်နှုန်းက တစ် မိနစ် တစ်ခါ ဖြစ်ပြီး၊ အနေးဆုံး ကြိမ်နှုန်းက တစ်နှစ် တစ်ခါ ဖြစ်ပါတယ်။

Cron က အလိုအလျောက် ပုံမှန် အချိန် ပေးနိုင် backup လုပ်တာ၊ maintenance လုပ်တာတွေ၊ တခြား အချိန် ပေးနိုင် ဝါယာ၊ အချိန် ထိန်းကျောင်း စတာတွေ အတွက် အလွယ်တကူ အသုံးပြု နိုင် ပါတယ်။ ဥပမာ RPi နဲ့ ဆက်ထား တဲ့ sensor တွေကို အချိန်မှန် ဖတ်၊ controller တွေကို အချိန်မှန် ထိန်းကျောင်း စတာတွေ အတွက် အလွယ်တကူ အသုံးပြု နိုင် ပါတယ်။ ဥပမာ ဗြိဒ္ဓဟန် surveillance ကင်မရာ အတွက် ဆိုရင် ဗြိဒ္ဓဟန် ဖိုင်တွေ များများ လာတဲ့ အခါ စက်မှာ နေရာ မလောက် တော့မယ့် ပြဿနာ ရှိလာ နိုင်ပါတယ်။ အဲဒီ အတွက် ဖိုင် အဟောင်းတွေ ကို သတ်မှတ် ထားတဲ့ ဆာဟာ မှာ အလိုအလျောက် backup လုပ်တာ၊ ပြီးတဲ့ အခါ ဖျက်ပစ်တာ စတာတွေကို script ဖိုင် တစ်ခု ရေးပြီး cron ကို ပုံမှန် လုပ်ခိုင်း လို့ ရပါတယ်။

Crontab ဖိုင်ကို ဖွင့်ဖို့ အတွက် terminal မှာ အောက်က command ကို ရိုက်နိုင် ပါတယ် [Hof11]။

```
$ crontab -e
```

ပထာမ ဆုံး အကြိမ် ဆိုရင် သုံးမယ့် editor ကို သတ်မှတ် နိုင်ပြီး၊ သုံးနေကျ nano ကို ပုံ ၁၀.၁၉ မှာ ပြထား သလို ရွေးလိုက် ပါမယ်။

```
pi@raspberrypi:~ $ crontab -e
no crontab for pi - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      <---- easiest
 3. /usr/bin/vim.tiny

Choose 1-3 [2]: 2
```

ပုံ ၁၀.၁၉: Crontab ကို ပြုပြင်ခြင်း။

### ၁၀.၅.၁ Crontab တွင်လုပ်ငန်းသတ်မှတ်ခြင်း

Crontab မှာ လုပ်ငန်း တစ်ခု ကို သတ်မှတ် မယ် ဆိုရင် မိနစ် (0-59)၊ နာရီ (0-23)၊ ရက် (1-31)၊ လ (1-12)၊ နေ့ (0-6)၊ နဲ့ လုပ်ရမယ့် command တွေကို ကြေားထဲမှာ space ခံပြီး ရေးနိုင် ပါတယ်။ ဘာ တန်ဖိုး ပဲ ဖြစ်ဖြစ် လုပ်ချင်တယ် ဆိုရင် \* ကို ဖြည့်နိုင် ပါတယ်။ ဥပမာ လ နေရာမှာ \* ထည့်ထား ရင် လစဉ် ပုံမှန် လုပ် မှာပါ။ တန်ဖိုး တစ်ခု မက ထည့်ချင်ရင် comma (,) ခံပြီး ထည့်လို ရပါတယ်။ ဥပမာ နာရီ နေရာ မှာ 6,18 လိုပေး ရင် မနက် ၆ နာရီ တစ်ခါ၊ ၂၄နာရီ ၆ နာရီ တစ်ခါ လုပ် ပါမယ်။ အကယ်၍ range ကို ထည့်ချင်ရင် dash (-) ကို သုံးနိုင် ပါတယ်။ ဥပမာ နေရာ မှာ 1-5 ဆိုရင် တန်လှာ ကနေ သောကြာ ထိ နေတိုင်း လုပ် ပါမယ်။ လုပ်ငန်း တစ်ခု ကို တစ်ကြောင်း နဲ့ လိုသလောက် ထည့်နိုင် ပြီး၊ comment လုပ် ချင်ရင် စာကြောင်း အစ မှာ hash (#) သက်တဲ့ ကို ထည့်နိုင် ပါတယ်။

ဥပမာ အနေနဲ့ vctask.sh ဆိုတဲ့ script ကို နာရီတိုင်း run ချင်ရင် အောက်က အတိုင်း crontab မှာ ဖြည့်ပြီး၊ ctrl+o နဲ့ သိမ်း၊ ctrl+x နဲ့ ထွက်နိုင် ပါတယ် (ပုံ ၁၀.၂၀)။ အဲဒီ အခါ crontab: installing new crontab ဆိုပြီး သတ်မှတ် လိုက်တဲ့ လုပ်ငန်း အောင်အောင် မြင်မြင် တပ်ဆင် ပြီးတဲ့ အကြောင်း တွေ ရမှာ ပါ။

```
0 * * * * /home/pi/vctask.sh
```

```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 2.7.4          File: /tmp/crontab.TKNjSu/crontab          Modified
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 * * * * /home/pi/vctask.sh_
```

**Key Bindings:**

- ^G Get Help
- ^O Write Out
- ^W Where Is
- ^K Cut Text
- ^J Justify
- ^C Cur Pos
- ^X Exit
- ^R Read File
- ^V Replace
- ^U Uncut Text
- ^T To Spell
- ^L Go To Line

ပုံ ၁၀.၂၁: Crontab တွင် လုပ်ငန်းများသတ်မှတ်ခြင်း။

Cron နဲ့ သတ်မှတ် ထားတဲ့ အလုပ် တွေက ကွန်ပျူးတာ ပိတ်ထား တဲ့ အချိန်မှာ မလုပ်လိုက် ရဘူး ဆိုရင်၊ ကွန်ပျူးတာ ပြန်ပွင့် လာတဲ့ အခါ လုပ်ဆောင် ပေးမှာ မဟုတ် ပါဘူး။ အဲဒီ လို မလုပ်လိုက်ရတဲ့ အလုပ် တွေကို စက်စ ပြန်ပွင့် လာတဲ့ အချိန်မှာ လုပ်ဆောင် ပေးစေခဲင် ရင်တော့ Anacron (anachronistic cron) ကို သုံးနိုင် ပါတယ်။ သူကို အောက်က command နဲ့ တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt install anacron
```

## အကိုးအကားများ

[Ada16a] Lady Ada. Adding a Real Time Clock to Raspberry Pi. 2016. url: <https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi>.

[Bou17b] Brian Boucheron. How To Set Up Time Synchronization on Ubuntu 16.04. 2017. url: <https://www.digitalocean.com/community/tutorials/how-to-set-up-time-synchronization-on-ubuntu-16-04>.

- [Hof11] Chris Hoffman. How to Schedule Tasks on Linux: An Introduction to Crontab Files. 2011. url: <https://www.howtogeek.com/101288/how-to-schedule-tasks-on-linux-an-introduction-to-crontab-files/>.
- [Kun10] Kunilkuda. Howto Use Linux Watchdog. 2010. url: <https://embeddedfreak.wordpress.com/2010/08/23/howto-use-linux-watchdog/>.
- [mbs08] mbsullivan. HOWTO: Set Up an NTP Server. 2008. url: <https://ubuntuforums.org/showthread.php?t=862620>.
- [Mol16] Derek Molloy. Exploring Raspberry Pi: Interfacing to the Real World with Embedded Linux. Wiley, 2016. isbn: 9781119188687. url: <http://www.exploringrpi.com/>.
- [Wik17a] Wiki. Network Time Protocol. 2017. url: [https://en.wikipedia.org/wiki/Network\\_Time\\_Protocol](https://en.wikipedia.org/wiki/Network_Time_Protocol).
- [Mil10] Mills, et al. Network Time Protocol Version 4: Protocol and Algorithms Specification. 2010. url: <https://tools.ietf.org/html/rfc5905>.
- [Mil92] Mills, et al. Network Time Protocol (Version 3) Specification, Implementation and Analysis. 1992. url: <https://tools.ietf.org/html/rfc1305>.
- [Ubu17] Ubuntu Server Guide. Time Synchronisation with NTP. 2017. url: <https://help.ubuntu.com/lts/serverguide/NTP.html>.

## အခန်း ၁၁

# Sensors

### ၁၁.၁ Temperature sensor ဖြင့်အပူချိန်ကိုအာရုံခံခြင်း

Analog Devices ကထုတ်တဲ့ TMP36 low voltage temperature sensor လေးက ပါဝါ အဝင် 2.7 V ကနေ 5.5 V ထိ အလုပ်လုပ် ပါတယ်။ ထုတ် ပေးတဲ့ analog output ဗြိုက လည်း -55°C ကနေ +125°C ကို 0 V ကနေ 1.8 V အတွင်း ပဲရှိ ပါတယ်။ သူ့ရဲ့ datasheet [Ana15] မှာ ဖော်ပြ ထားတာ က 50°C မှာ 1 V ထွက်ပြီး၊ ပုံ ၁၁.၁၁ မှာ ဖော်ပြ ထားတဲ့ အတိုင်း 10 mV/°C နဲ့ linear ပြောင်းလဲ တယ် လို ဆိုတဲ့ အတွက်၊ အထွက် ပို့ (V<sub>o</sub>) နဲ့ အပူချိန် (T) ရဲ့ ဆက်သွယ်ချက် ကို အောက်ပါ အတိုင်း ဖော်ပြ နိုင်ပါတယ်။

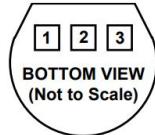
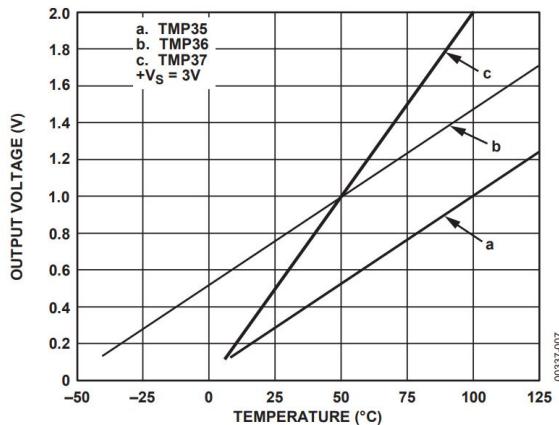
$$V_o = 1 + 0.01(T - 50) \quad (၁၁.၁)$$

ဒီ sensor ရဲ့ analog ဗြို့အား အထွက် ကို ဖတ်ဖို့ အတွက် Raspberry Pi ကို အပိုင်း ၄.၃ မှာ ဆွေးနွေး ခဲ့တဲ့ MCP3008 နဲ့ တွဲသုံး လိုက် ပါမယ်။ MCP3008 ရဲ့ ADC က 10 bits ဖြစ်တာ ကြောင့် ဖတ်လို ရမယ့် တန်ဖိုး (N) နဲ့ အပူချိန် (T) အတွက် ညီမျှခြင်း ၁၁.၂ အတိုင်း ဖော်ပြ နိုင် ပါတယ်။

$$N = \frac{1023}{V_{aref}}(0.01T + 0.5) \quad (၁၁.၂)$$

$$T = \frac{N \times V_{aref} - 511.5}{10.23} \quad (၁၁.၃)$$

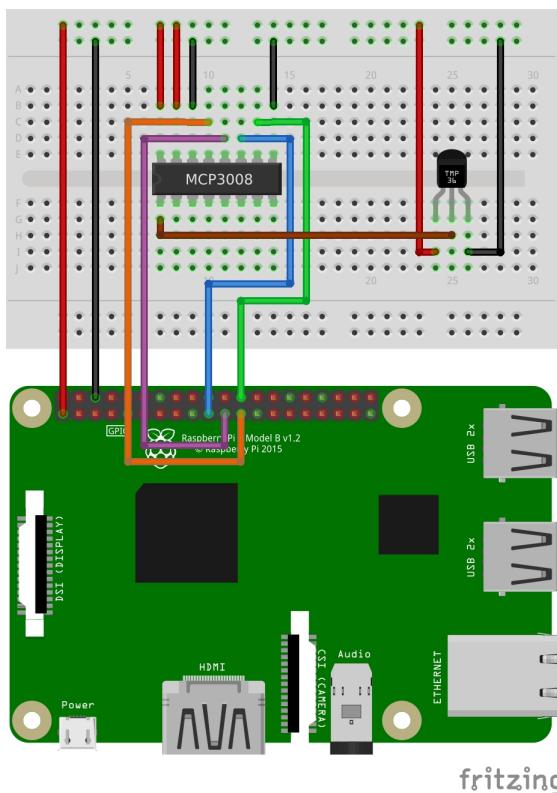
ပုံ ၁၁.၁၃ မှာ TO-92 package အတွက် TMP36 ရဲ့ pin တွေကို ဖော်ပြထား ပြီး၊ သူကို ပုံ ၁၁.၂ မှာ ပြထား သလို Raspberry Pi နဲ့ ဆက်သွယ် လိုက် ပါမယ်။



(b) TO-92 package အတွက် pin configuration။

(a) အထွက်မြို့နှင့် အပူချိန် ဆက်သွယ်မှု။

ပုံ ၁၁၁: TMP36 ၏ အထွက်မြို့နှင့် pin များ။ (From TMP36 datasheet)



ပုံ ၁၁၂: TMP36GT9Z နှင့် Raspberry Pi ကို ဆက်သွယ်ပုံ။

အပူချိန်ကို အာရုံခံ တဲ့ ကိရိယာ ကို အသုံးပြု တဲ့ နမူနာ C++ ပရိုဂရမ် temperature.cpp ကို စာရင်း ၁၁.၁ မှာ ပြထား ပါတယ်။ အပိုင်း ၄.၃ မှာ ဖော်ပြ ခဲ့တဲ့ ce\_spi.h ကို ပြန်သုံး ထားပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include "ce_spi.h"
4 using namespace std;
5 int main()
6 {
7     CE_SPI tsensor;
8     tsensor.Begin();
9
10    int n;
11    float v, t;
12    for(int i=0;i<10;i++){
13        tsensor.tx[0]=0x01;//Send start bit - 0000 0001
14        tsensor.tx[1]=0x80;//read channel 0 - | s/~d | d2 d1 d0 | x x x x |
15        // b7= single/~differential = 1 for single ended
16        // d2 d1 d0 = 0 for channel 0
17        tsensor.tx[2]=0x00; // don't care = x x x x x x x x
18        tsensor.Transfer(3);
19        //printf("%02X %02X \n", (unsigned char)tsensor.rx[1], (unsigned char)
20        tsensor.rx[2]);
21        n = tsensor.rx[1]<<8 | tsensor.rx[2];
22        n&=0x03FF;//mask out invalid bits
23        v = 3.3*float(n)/ 1023;//using 3.3 Vref
24        t = (float(n)*3.3 - 511.5) / 10.23;
25        printf("n= %d v=%f t=%f\n",n,v,t);
26        usleep(1000000);
27    }
28    return 0;
}

```

စာရင်း ၁၁.၁: temperature.cpp

သူကို အောက်ပါ အတိုင်း build လုပ်၊ run လုပ်နိုင်ပြီး၊ ပရိုဂရမ် ရဲ့ output ကို ပုံ ၁၁.၃ နဲ့ TMP36GT9Z

## ၁၁.၁. TEMPERATURE SENSOR ဖြင့်အပူချိန်ကိုအာရုံခံခြင်း

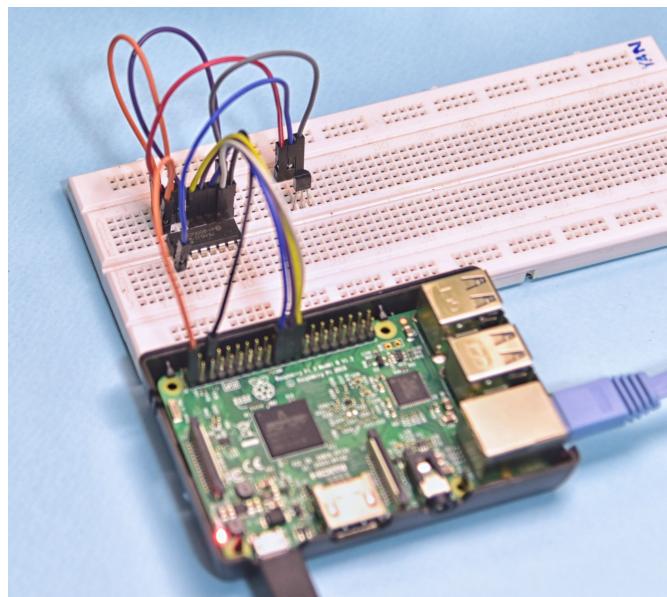
၃၂။

ကို ပုံ ၁၁.၄ မှာ ပြထား ပါတယ်။

```
$ g++ temperature.cpp -o temperature  
$ ./temperature
```

```
pi@raspberrypi:~/sensor $ ./temperature-bar.sh  
Building  
Running  
n= 238 v=0.767742 t=26.774193  
n= 237 v=0.764516 t=26.451612  
n= 237 v=0.764516 t=26.451612  
pi@raspberrypi:~/sensor $
```

ပုံ ၁၁.၃: temperature.cpp ၏ရလဒ်။



ပုံ ၁၁.၄: TMP36GT9Z နှင့် Raspberry Pi ချိတ်ဆက်ထားပုံ။

## ၁၁.၂ Accelerometer သုံးခြင်း

LIS3DH က STMicroelectronics ထုတ်ပဲ 3-axis MEMS accelerometer တစ် ဖြစ်ပါ တယ်။ Full scale ကို  $\pm 2$  g ကနေ  $\pm 16$  g ထိ အမျိုးမျိုး ရွှေးလို ရ ပါတယ်။ အရွယ် အစား က လည်း 3 mm x 3 mm ပဲမို့ သေးသေးလေး ပါ။ Digital output ကို SPI ဒါမ္မမဟုတ် I2C ကြိုက်တာ သုံးလို ရ ပါတယ်။ Supply voltage က 1.71 V ကနေ 3.6 V ထိ ပေးလို ရ ပါတယ် [STM16]။ Sparkfun LIS3DH Breakout လေး နဲ့ အဲဒီ accelerometer ကို အသုံး ပြုတဲ့ အကြောင်း ဆွေးနွေး ပါမယ်။

အဲဒီ မှာ Acceleration တန်ဖိုး တွေကို 16 bit digital output နဲ့ output data rate (ODR) 1.344 kHz ထိ ထုတ်ပေး နိုင်တာ ကလည်း သဘော ကျမိုး တဲ့ အချက် တရ ပါ။ Digital output အတွက် resolution ကို Singapore မှာရှိတဲ့ gravity 9781 mm/s<sup>2</sup> နဲ့ ရှာလိုက် ရင် အောက်ပါ အတိုင်း ရ ပါမယ်။

$$\text{mg/digit} = \frac{2g - (-2g)}{2^{16} - 1} = 61 \mu\text{g} = 0.5966 \text{ mm s}^{-2} \quad (၁၁.၄)$$

ဒါကြောင် 61  $\mu\text{g}/\text{digit}$  ထိ ခွဲခြား ဖော်ပြ နိုင် ပါတယ်။ အဲဒီက digital output ရဲ့ ခွဲခြား ဖော်ပြ နိုင်တဲ့ sensitivity (resolution) ဖြစ်ပြီး accelerometer ရဲ့ တကယ် sense လုပ်နိုင် တဲ့ sensitivity (resolution) ကတော့ သူရဲ့ noise floor ကနေ တွက်ယူ ရ ပါမယ်။ Datasheet မှာ ဖော်ပြ ထားတဲ့ သူရဲ့ noise density က 220  $\mu\text{g}/\text{sqrt(Hz)}$  ဖြစ်တာ ကြောင့် bandwidth 100 Hz သုံးမယ် ဆိုရင် သူရဲ့ resolution ကို 2200  $\mu\text{g}$  လို ရ ပါမယ်။

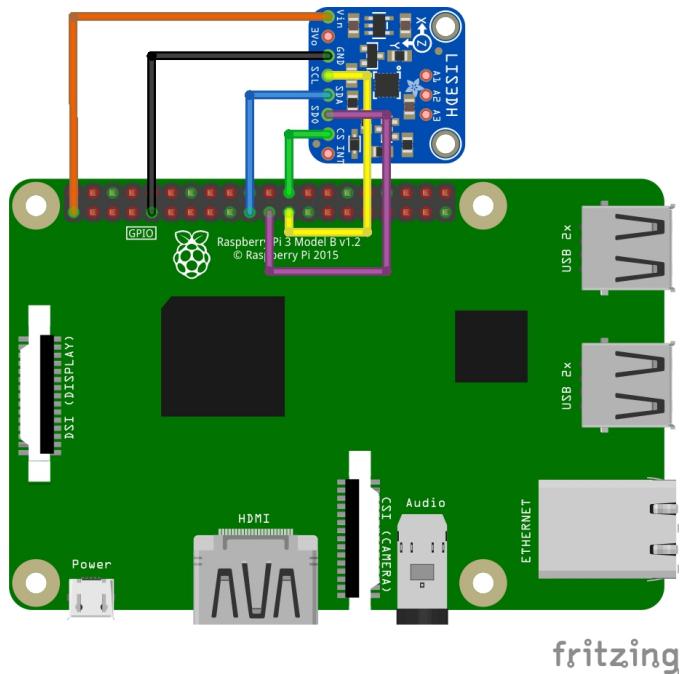
$$An = 220 \times \sqrt{BW} = 2200 \mu\text{g} = 21.52 \text{ mm s}^{-2} \quad (၁၁.၅)$$

## ၁၁.၂.၁ SPI ဖြင့် သုံးခြင်း

SPI interface ကို သုံးဖို့ အတွက် LIS3DH ကို ပုံ ၁၁.၅ အတိုင်း ဆက်နိုင် ပါတယ်။ ဒီနူးမူးနာ မှာတော့ 4-wires ပဲ သုံးပြီး interrupt တွေကို မသုံး ပါဘူး။

## ၁၁. J. ACCELEROMETER သုံးခွင့်

RJR



fritzing

ပုံ ၁၁.၂: LIS3DH ကို SPI သုံးချိ Raspberry Pi နှင့်ဆက်သွယ်ပဲ။

နှမူနာ C++ ပရိုဂရမ် accelerometer-spi.cpp ကို စာရင်း ၁၁.၂ မှာ ပြထား ပါတယ်။ အပိုင်း ၄.၃ မှာ ဖော်ပြ ခဲ့တဲ့ ce\_spi.h ကို ပြန်သုံး ထား ပါတယ်။

```
1 // File: accelerometer-spi.cpp
2 // Description: SPI communication with LIS3DH accelerometer
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 #include <stdio.h>
8 #include <stdlib.h>
9 #include "ce_spi.h"
10 using namespace std;
11
12 int main()
13 {
14     CE_SPI a;
15     a.Begin();
```

```

16
17     int x,y,z;
18
19     float K=0.061036; // (4000/65535) milli-g per digit for +/-2g full scale
20     //using 16 bit digital output
21
22     //read the "who am i" register at address 0x0F
23     //Set R/~W bit to read.
24     //its value should be 0x33
25
26     a.tx[0] = 0x8F;
27     a.tx[1] = 0x00;
28     a.Transfer(2);
29
30     printf("I am 0x%02x\n", a.rx[1]);
31
32
33     a.tx[0]=0x20;//Send address of 'Control register 1' to write configuration
34     a.tx[1]=0x97;//Write a value that enables x,y,z accelerometers, ODR 1.344
35     kHz,
36     // High resolution mode so that BW is ODR/9 ~ 150Hz
37     a.Transfer(2);
38
39
40     a.tx[0]=0x23;//Send address of 'Control register 4' to write configuration
41     a.tx[1]=0x88;//set BDU - block data update, set HR - High resolution mode
42     a.Transfer(2);
43
44
45     for(int j=0;j<10;j++)
46     {
47
48         a.tx[0]=0xE8;//Send address of LSB of x.
49         //Set R/~W bit to read. Set M/~S to auto-increase address for multiple rw
50         .
51
52         a.Transfer(7);// 6 bytes to read after the address
53
54         //printf("0x%02x%02x 0x%02x%02x 0x%02x%02x \n", a.rx[1],a.rx[0],a.rx[3],a
55         .rx[2],a.rx[5],a.rx[4]);
56
57         x=((int(a.rx[2])<<8)+a.rx[1]) | (a.rx[2] & 0x80 ? 0xFFFF0000 : 0x00000000
58         );
59
60         y=((int(a.rx[4])<<8)+a.rx[3]) | (a.rx[4] & 0x80 ? 0xFFFF0000 : 0x00000000
61         );
62
63         z=((int(a.rx[6])<<8)+a.rx[5]) | (a.rx[6] & 0x80 ? 0xFFFF0000 : 0x00000000
64         );
65
66     }

```

```

    );
    printf("x = %d > mg = %f , \t ",x,(K*x));
    printf("y = %d > mg = %f , \t ",y,(K*y));
    printf("z = %d > mg = %f ",z,(K*z));
    printf("\n");
    usleep(1000000);
}
return 0;
}

```

စာရင်း ၁၁.၂: accelerometer-spi.cpp

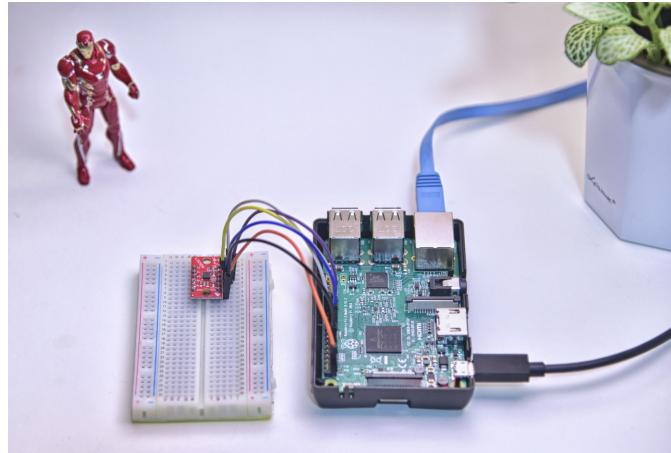
သူကို အောက်ပါ အတိုင်း build လုပ်၊ run လုပ်နိုင် ပါတယ်။

```
$ g++ accelerometer-spi.cpp -o accelerometer-spi
$ ./accelerometer-spi
```

ဆက်သွယ် ထားတဲ့ Accelerometer (ပုံ ၁၁.၇) ကို အနေ အထား အမျိုးမျိုး ပြောင်းကြည့် တဲ့ အခါ gravity ရဲ့ သက်ရောက်မှု အလိုက်သင့် ပြောင်းသွား တာကို ပရိုဂရမ် ရဲ့ အထွက် မှာ ပုံ ၁၁.၆ အတိုင်း ထွေးရ မှာ ဖြစ်ပါတယ်။

```
pi@raspberrypi:~/sensor $ ./accelerometer-spi
I am 0x33
x = 416 > mg = 25.390976 ,      y = -960 > mg = -58.594559 ,      z = 15824 > mg = 965.833618
x = 448 > mg = 27.344128 ,      y = -960 > mg = -58.594559 ,      z = 16048 > mg = 979.505676
x = 432 > mg = 26.367552 ,      y = -944 > mg = -57.617981 ,      z = 15936 > mg = 972.669678
x = 3952 > mg = 241.214264 ,     y = -288 > mg = -17.578367 ,      z = 15840 > mg = 966.810181
x = 13760 > mg = 839.855347 ,     y = 1024 > mg = 62.500862 ,      z = 9888 > mg = 603.523926
x = 13664 > mg = 833.995850 ,     y = 832 > mg = 50.781952 ,      z = 9280 > mg = 566.414062
x = 13776 > mg = 840.831909 ,     y = 112 > mg = 6.836032 ,      z = 9648 > mg = 588.875305
x = 5040 > mg = 307.621429 ,     y = -13952 > mg = -851.574219 ,     z = 9328 > mg = 569.343811
x = 4192 > mg = 255.862900 ,     y = -13424 > mg = -819.347229 ,     z = 7968 > mg = 486.334839
x = 4304 > mg = 262.698944 ,     y = -13760 > mg = -839.855347 ,     z = 7664 > mg = 467.779877
pi@raspberrypi:~/sensor $
```

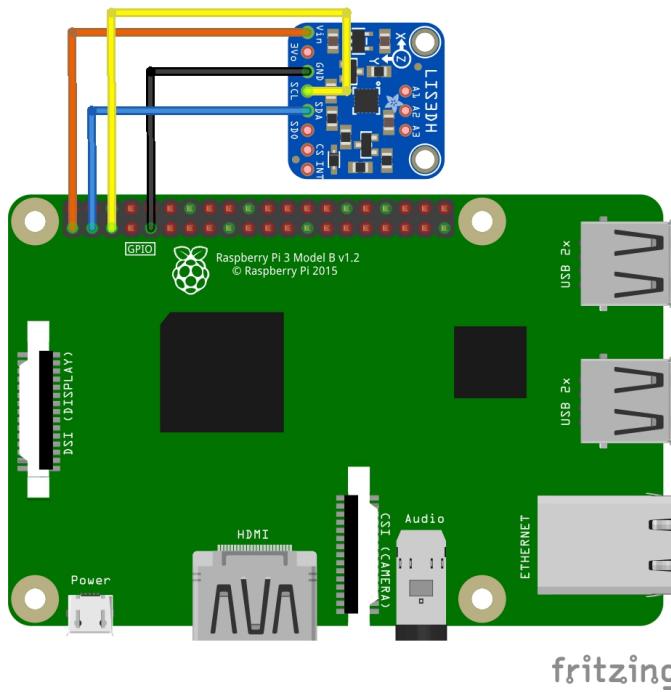
ပုံ ၁၁.၆: LIS3DH ကိုဖတ်၍ ရသည့် ရလဒ်။



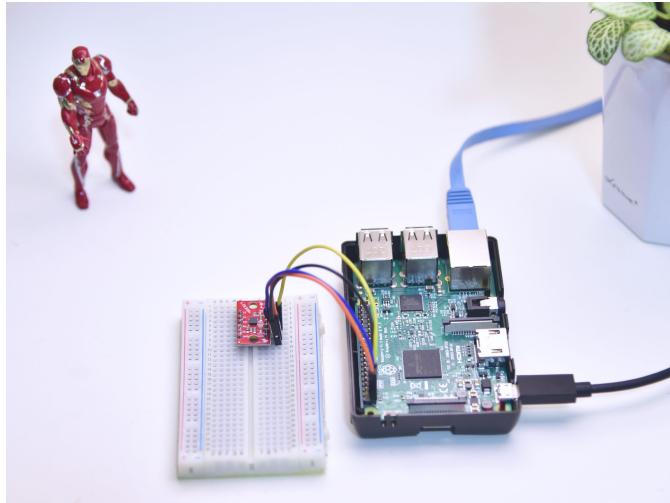
ပုံ ၁၁.၇: LIS3DH ဆက်သွယ် ရုံး အတွက် SPI အသုံးပြု ထားပုံ။

### ၁၁.၈ I2C ဖြင့် သုံးခြင်း

LIS3DH ကို I2C interface က တဆင့် သုံးဖို့ အတွက် တော့ ပုံ ၁၁.၈ နဲ့ ပုံ ၁၁.၉ မှာ ပြထား တဲ့ အတိုင်း Raspberry Pi နဲ့ ဆက်နိုင် ပါတယ်။



ပုံ ၁၁.၈: LIS3DH ကို I2C သုံး၍ Raspberry Pi နှင့် ဆက်သွယ်ပုံ။



ပုံ ၁၁.၉: LIS3DH ဆက်သွယ် ရန် အတွက် I2C အသုံးပြု ထားပို။

အောက်က command တွေသုံးပြီး i2c bus ကို probe လုပ်ကြည့်လိုက်ရင် device address 0x19 ကို ပုံ ၁၁.၁၀ အတိုင်း တွေ့ရပါမယ်။ အဲဒါ က သူရဲ့ device address pin A0 ကို ဘာမှ ဆက်မထား တဲ့ အတွက် pull up လုပ်ထား တဲ့ တန်ဖိုး 1 ဖြစ်နေ တဲ့ အချိန် ပါ။ LIS3DH နှစ်ခု ပြိုင်တူ ဆက်မယ် ဆိုရင် နောက်တစ်ခု ရဲ့ A0 ကို GND နဲ့ jumper ဆက်ပြီး device address မတူ အောင် (0x18 ဖြစ်အောင်) လုပ်လို ရပါတယ်။ i2cdump နဲ့ ကြည့်လိုက် တဲ့ အခါ accelerometer ရဲ့ register တန်ဖိုး တွေကို လည်း တွေ့နိုင် ပါတယ်။

```
$ i2cdetect -l
$ i2cdetect -y -r 1
$ i2cdump -y 1 0x19
```

```

File Edit Tabs Help
pi@raspberrypi:~ $ i2cdetect -l
i2c-1    i2c          bcm2835 I2C adapter
pi@raspberrypi:~ $ i2cdetect -y 1
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -----
10: -- - - - - - - - - - - - - - - - - - - - - - -
20: -----
30: -----
40: -----
50: -----
60: -----
70: -----
pi@raspberrypi:~ $ i2cdump -y 1 0x19
No size specified (using byte-data access)
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 33
10: 9d 0b 03 21 a0 31 20 23 1e a0 70 75 c0 00 10 00
20: 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 20
30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 33
90: 9d 0b 03 21 a0 31 20 23 1e a0 70 75 c0 00 10 00
a0: 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 20
b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
pi@raspberrypi:~ $

```

ပုံ ၁၁.၁၀: LIS3DH အတွက် I2C bus ကို probe လုပ်ခြင်း။

နမူနာ C++ ပရိုဂရမ် accelerometer-i2c.cpp ကို စာရင်း ၁၁.၃ မှာ ပြထားပါတယ်။ အပိုင်း ၄.၁ မှာ ဖော်ပြ ခဲ့တဲ့ ce\_i2c.h ကို ပြန်သုံးထားပါတယ်။

```

1 // File: accelerometer-i2c.cpp
2 // Description: I2C communication with LIS3DH accelerometer
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye
6
7 #include<stdio.h>
8 #include"ce_i2c.h"
9 using namespace std;
10 int main()
11 {
12     int x,y,z;
13     float K = 0.061036; // (4000/65535) milli-g per digit for +/-2g full scale
                        using 16 bit digital output

```

```

14 char d[8]={0,0,0,0,0,0,0,0};
15 CE_I2C a(1,0x19); //i2c-1 , device address 0x19
16
17 //read the "who am i" register at address 0x0F
18 //its value should be 0x33
19 d[0]=0x0F;
20 a.Write(d,1);
21 a.Read(d,1);
22 printf("I am 0x%02x\n",d[0]);
23
24 d[0]=0x20; //Send address of 'Control register 1' to write configuration
25 d[1]=0x97; //Write a value that enables x,y,z accelerometers, ODR 1.344kHz,
26 // High resolution mode so that BW is ODR/9 ~ 150Hz
27 a.Write(d,2);
28
29 d[0] = 0x23; //Send address of 'Control register 4' to write configuration
30 d[1] = 0x88; //set BDU - block data update, set HR - High resolution mode
31 a.Write(d, 2);
32
33 for(int i=0;i<10;i++)
34 {
35     //read address 0x28, OUT_X_L register
36     d[0]=0xA8; //set msb for address auto-increase
37     a.Write(d,1);
38     a.Read(d,6);
39     //printf("0x%02x%02x 0x%02x%02x 0x%02x%02x \n", d[1],d[0],d[3],d[2],d[5],d
40     [4]);
41     x=((int(d[1])<<8)+d[0]) | (d[1] & 0x80 ? 0xFFFF0000 : 0x00000000);
42     y=((int(d[3])<<8)+d[2]) | (d[3] & 0x80 ? 0xFFFF0000 : 0x00000000);
43     z=((int(d[5])<<8)+d[4]) | (d[5] & 0x80 ? 0xFFFF0000 : 0x00000000);
44     printf("x = %d > mg = %f , \t ,x,(K*x));
45     printf("y = %d > mg = %f , \t ,y,(K*y));
46     printf("z = %d > mg = %f ",z,(K*z));
47     printf("\n");
48     usleep(1000000);
49 }
```

```

49     return 0;
50 }
```

### စာရင်း ၁၁.၃: accelerometer-i2c.cpp

သူကို အောက်ပါ အတိုင်း build လုပ်၊ run လုပ်နိုင် ပါတယ်။

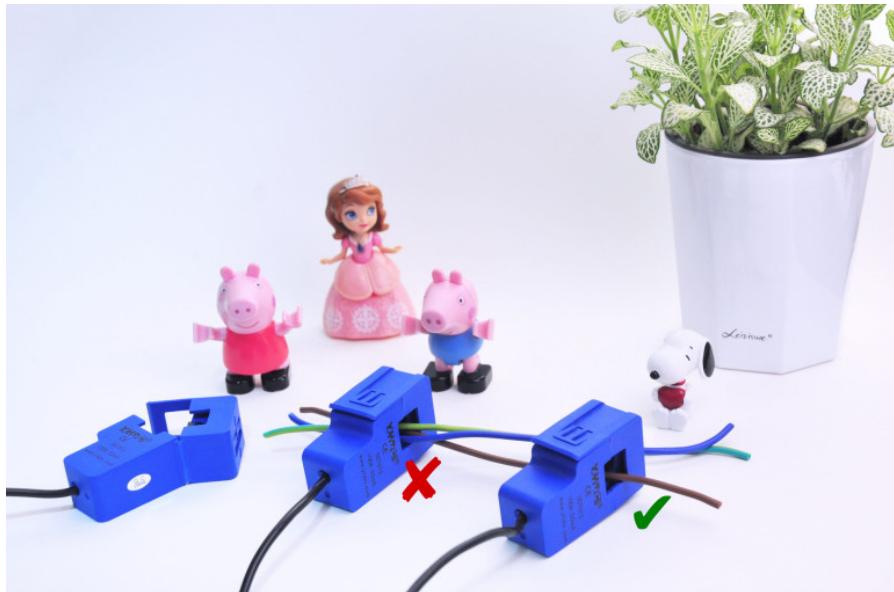
```

$ g++ accelerometer-i2c.cpp -o accelerometer-i2c
$ ./accelerometer-i2c
```

## ၁၁.၃ Energy Monitor ပြည်ခြင်း

အိမ်မှာ ရှိတဲ့ လျှပ်စစ် အသုံး အဆောင် တွေရဲ့ စွမ်းအင် အသုံး ပြုမှု ကို စီစစ် တိုင်းတာ ဖို့ အတွက် ရှိုးရှင်း ပြီး၊ အခြေခံ ကျတဲ့ energy monitor တစ်ခု ကို လုပ်ကြည့် ပါမယ်။ တိုင်းတာ အသုံး ပြုတဲ့ နည်း အများကြီး ရှို့ပေ မယ့် သင်ယူ လေ့လာမှု အတွက် အထောက် အကူးဖြစ်ဖို့ ရော၊ အန္တရာယ် ကင်းဖို့ အတွက်ပါ သင့်တော်မှု ရှို့မယ် ထင်တဲ့ YHDC Current Transformer SCT-013-000 [YHD18] ကို သုံးဖို့ ရွှေးချယ် လိုက်ပါ တယ်။

သူက split-core ဖြစ်ပြီး တိုင်းတာ ချင်တဲ့ ဖို့အား မြင့် ဝါယာ လိုင်း တွေကို ဘာမှ ဖြုတ် စရာ မလိုပဲ ဝါယာ ပေါ်ကို ပတ်ပြီး ကလစ် တပ်လိုက် ဖို့ပဲ လိုတာမို့ non-invasive ဖြစ်ပြီး၊ အန္တရာယ် လည်းကင်း၊ အသုံးပြုဖို့လည်း လွယ်ကူ ပါတယ်။ အဲဒီ လို ပတ်တဲ့ အခါ ပုံ ၁၁.၁၁ ရဲ့ ညာဖက် ဆုံးမှာ ပြထား သလို ဝါယာ တစ်စ ကိုပဲ ပတ်ရ မှာဖြစ် ပါတယ်။



ပုံ ၁၁.၁၁: (ဘယ်ဘက်) SCT-013 က split core ဖြစ်တဲ့ အတွက် ကလစ်ကို ဖွင့်ပြီး ဝါယာ ပေါ်ကို အလွယ် တကူ တပ်ဆင်နိုင် ပါတယ်။ (အလယ်) ဝါယာ အကုန်လုံးကို ပတ်လိုက်ရင် အင် နဲ့ အတွက် ဖြန့်ကြေပြီး core ထဲမှာ ဖြတ်စီး တဲ့ စုစုပေါင်း current က သူည့် ဖြစ်သွား တဲ့ အတွက် တိုင်းလို့ မရ ပါဘူး။ (ညာဖက်) ဝါယာ တစ်စ ကို ပဲ core ထဲမှာ ပတ်ဖို့ လိုပါတယ်။

### ၁၁.၃.၁ Current တိုင်းတာရန်ပြင်ဆင်ခြင်း

Current transformer တွေက primary winding မှာ စီးဆင်း နေတဲ့ AC current ကို ဝါယာ ပတ်ထား တဲ့ အပတ် အရေး အတွက် အချိုး အလိုက် secondary winding မှာ induced လုပ်ပေး ပါတယ်။ သူ့ရဲ့ datasheet မှာ ဖော်ပြ ထား တာက turn ratio 2000:1 ဖြစ်ပြီး၊ အင် current 100 A (rms) အတွက် အထွက် current 50 mA (rms) ထူတ်ပေး ပါမယ်။ Rated current က 120 A ဖြစ်ပြီး၊ အထဲ မှာ transient voltage suppressor (TVS) ပါ ပါတယ်။

အခု ဒီဇီုင်း လုပ်ချင်တဲ့ မိတ္တ မှာ အင် primary rms current  $I_{i_{rms}}$  ကို 30 A လောက်ထိ ပဲ အများဆုံး တိုင်းချင် ပြီး၊ interface လုပ်မဲ့ signal processor ဘက်မှာ 3.3 V လောက် ရုံး စဉ်းစား ပါမယ်။ CT (current transformer) ကထွက်လာ မယ့် secondary current  $I_s$  ကို ရုံး အတွက်၊ အင် current ကို turn ratio နဲ့ အချိုးချွဲ ပါမယ်။ Passive load တွေ အတွက် ရည်ရွယ် ပြီး ဒီဇီုင်း လုပ်မှာ ဖြစ်တဲ့ အတွက် current ကို လည်း sinusoidal လို့ ယူဆ ပြီး၊  $I_s$  ရဲ့ peak to peak တန်ဖိုး က rms တန်ဖိုး ရဲ့  $2\sqrt{2}$  ဆ လို့ သတ်မှတ် နိုင် ပါတယ်။ အဲဒီ အခါ အင် rms current နဲ့ အတွက် peak to

peak current ရဲ့ ဆက်သွယ်မှု ကို ၁၁.၇ အတိုင်း ရနိုင်ပါတယ်။

$$I_{s_{p-p}} = \frac{I_{i_{rms}}}{2000} \times 2\sqrt{2} \quad (၁၁.၆)$$

$$I_{s_{p-p}} = \frac{\sqrt{2}I_{i_{rms}}}{1000} \quad (၁၁.၇)$$

### Burden Resistor

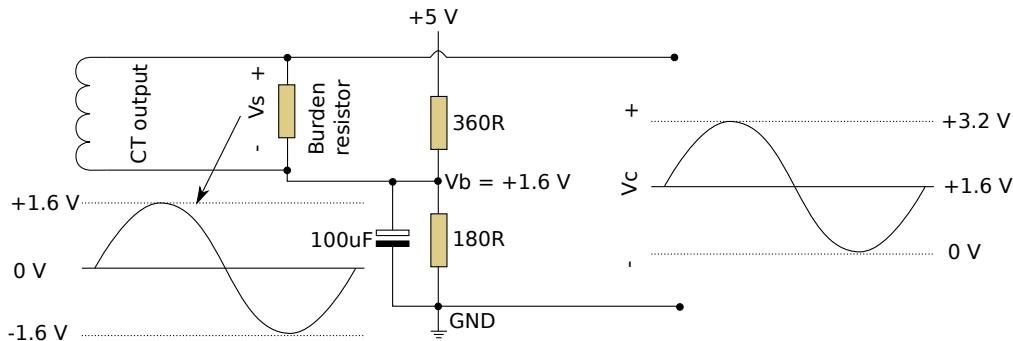
ဥပမာ အဝင် current 30 A အတွက် CT ရဲ့ အထွက် က 42.43 mA peak to peak ဖြစ်ပါတယ်။ အဲဒီ အချိန် မှာ ရချင်တဲ့ voltage က 3.3 V ဆိုပါတဲ့ Ohm's law  $R = \frac{V}{I}$  ကို သုံးပြီး ရှာတဲ့ အခါ သုံးရမယ့် burden resistance တန်ဖိုး ကို 78 Ω ဆိုပြီး ရပါမယ်။ အဲဒီ အတွက် အလွယ် တကူ ရရှိ နိုင်တဲ့ အနီး စပ်ဆုံး common resistor တန်ဖိုး 75 Ω ကို သုံးနိုင်ပါတယ်။ Burden resistance 75 Ω ကို သုံးတဲ့ အခါ ထွက်လာ မယ့် peak to peak voltage  $V_{s_{p-p}}$  ကို ၁၁.၉ အတိုင်း ရနိုင်ပါတယ်။

$$V_{s_{p-p}} = \frac{75\sqrt{2}I_{i_{rms}}}{1000} \quad (၁၁.၈)$$

$$V_{s_{p-p}} = 0.106 \times I_{i_{rms}} \quad (၁၁.၉)$$

### ၁၁.၂.J Bias ပေးခြင်း

ရလာတဲ့ ဗိုက အပေါင်း အနှစ် တလုညွှိ ဖြစ်နေ တာမို့ signal processor ဘက် အတွက် အဆင်ပြေ တဲ့ အပေါင်း ဗို့ အမြဲ ဖြစ်နေ အောင် bias voltage တစ်ခု ကို ထည့်ပေါင်း ပေးပါမယ် [LH18]။ ပုံ ၁၁.၁၂ မှာ ပြထား သလို ခန်းမှန်းခြေ -1.6 V နဲ့ 1.6 V ကြားမှာ ပြောင်း နေတဲ့ အဝင် ဗို့ ကို 1.6 V လောက် ရှိတဲ့ bias voltage  $V_b$  ပေါင်း ပေးလိုက် တဲ့ အခါ 0 V နဲ့ 3.2 V ကြားမှာ ပြောင်း နေတဲ့ အထွက် ဗို့  $V_c$  ကို ရလာ ပါမယ်။ Bias voltage ရဖို့ အတွက် တော့ ရှိုးရှင်း တဲ့ voltage divider လေးကို 180 Ω resistor တွေ သုံးပြီး တည်ဆောက် နိုင်ပါတယ်။ 360 Ω resistor ရဖို့ 180 Ω resistor နှစ်လုံး ကို series ဆက်ပြီး သုံးနိုင်ပါတယ်။ ရလာ တဲ့ အထွက် ဗို့  $V_c$  ကို Arduino တို့လို microcontroller တွေနဲ့ ဆက်ပြီး current တွေ ပါဝါ တွေကို တိုင်းတာ တွက်ချက် နိုင်ပါတယ်။



ပုံ ၁၁.၁၂: Bias ဖို့ ထည့်ပေးခြင်း။

### ၁၁.၃.၃ Power တိုင်းတာခြင်း

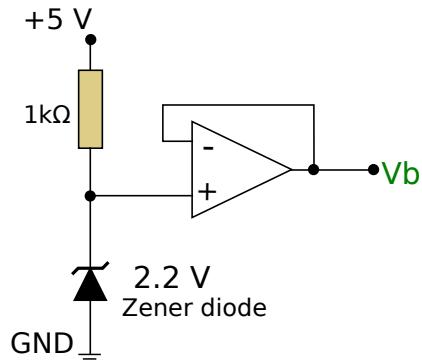
ရွှေမှာ ဆွေးနွေး ခဲ့တဲ့  $V_c$  ကို တိုင်းတာ ပြီး၊ CT တပ်ထား တဲ့ ဝါယာ မှာ စီးဆင်း နေတဲ့ current ကို သိနိုင် ပါတယ်။ Power ရဖို့ အတွက် voltage နဲ့ current ကို မြောက်ပြီး၊ တွက်နိုင် ပါတယ်။ အခု တည်ဆောက် မပုံး energy monitor မှာ အောက်ပါ အချက် တွေ အတိုင်း ယူဆ မှု တွေ လုပ်ထား ပါတယ်။

- ၁။ လျှပ်စစ် ဗို့အား ကို 230 V အသေလို ယူဆ ထား ပါမယ်။ မီးအား မြှုပ် ရင်တောင် မှ တိုင်းတာ မှု လုပ်ထား တဲ့ ဝါယာက အလို အလျောက် မီးအား မြှင့်ထား တဲ့ ဘက်မှာ ရှိရင် ယူဆ ချက် သိပ် မမှားဘူး ပြောနိုင် ပါမှာ။ အကယ်၍ အတိ အကျ လိုချင် ရင်တော့ voltage divider ဒါမှ မဟုတ် voltage transformer လေးပဲ ဖြစ်ဖြစ် သုံးပြီး ဗို့အား အတိ အကျ တိုင်းတာ ရရှိ နိုင် ပါတယ်။
- ၂။ သုံးမယ့် load ရဲ့ reactive power ကြောင့် ရလာ မယ့် apparent power  $P = VI$  က real power နဲ့ သိပ် မကွာ ဘူးလို ယူဆ ပါမယ်။ ဥပမာ စင်ကာပူ မှာ ဆိုရင် သတ်မှတ် ထားတဲ့ power factor 0.95 ထက် နည်းတဲ့ load ကို သုံးလို မရ ပါဘူး။ အတိ အကျ လိုချင် ရင်တော့ arduino တို့လို microcontroller တစ်ခုခု နဲ့ voltage, current တွေကို 1 kHz လောက်နဲ့ sample ကောက်ပြီး တွက်လို ရသလို ADE9153A တို့လို energy metering IC တွေကို သုံးလို ရ ပါတယ် [Ana18]။
- ၃။ သုံးမယ့် load တွေက passive load တွေ ပဲဖြစ် တယ် လို ယူဆ ပြီး၊ current က ဗို့အား နဲ့ linear ဖြစ်ပြီး sinusoidal ပုံစံ ပဲ တွက်မယ် လို ယူဆ ပါမယ်။ ဒါ နမူနာ မှာ တော့ rms တန်ဖိုး တွက်တဲ့ အခါ ပုံမှန် sinusoidal တွေရဲ့ ဆက်သွယ် ချက် နဲ့ပဲ ခန်းမှန်း တွက်ချက် ထားပါတယ်။ မဟုတ် ရင် တော့ AD8346 တို့လို RMS to DC Converters တွေကို သုံးလို လည်း ရပါတယ် [Ana17]။

နောက် တစ်ခါ အသုံးပြု တဲ့ component တွေရဲ့ တန်ဖိုး က လည်း အတိ အကျ မရ နိုင် တာကြောင့် အဲဒီ အတွက် error တွေ လည်း ရှိ ပါတယ်။ ဥပမာ 75 Ω burden resistor ဆိုရင် လည်း သူမှာ 1% ဒါမှ မဟုတ် 5% စသဖြင့် tolerance ရှိတာ မို့ ထွက်လာ တဲ့  $V_c$  မှာ လည်း မတိ ကျမှု တွေ ရှိ ပါတယ်။ သတင်း ကောင်း တစ်ခု ကတော့ energy monitor တွေဟာ အသေ တပ်ထား လေ့ ရှိတာမို့ အခါ ပြောခဲ့ တဲ့ ယူဆချက် တွေ ကြောင့် ရလာ တဲ့ အမှား တွေ အားလုံး နီးပါး ကို calibration လုပ်ပြီး ဖျောက်ပစ် နိုင်ပါ တယ်။ ဒါကြောင့် ရလာ မယ့် energy monitor က တော်တော် တိကျမှု ရှိနိုင် ပါတယ်။

### ၁၁.၃.၄ Signal Processing

ဂျွိန်တော် တို့ SBC ရဲ့ Linux က real-time ကြိမ်နှုန်း မြင့် လုပ်ငန်း တွေ မလုပ် နိုင်တာ မို့ အသုံးပြု တဲ့ rms current အလိုက် အချိုးကျ dc voltage ထွက်မယ့် ရှိုးရှင်း တဲ့ signal processor တစ်ခု ကို op-amp လေးတွေ သုံးပြီး တည်ဆောက် ပါမယ်။ အဲဒီ op-amp ရဲ့ အထွက် မို့ က အနည်းဆုံး 0.8 V ကနေ အများဆုံး  $V_{cc} - 1.5$  V အထိ ထုတ် ပေးနိုင် တာမို့ 5 V ပါဝါ သုံး လိုက်ရင်၊ အထွက် voltage range 2 V ကျော်ကျော် ရနိုင် ပါတယ် [Tex17]။ နှုံးနှုံး အနေနဲ့ ရွှေ့က ဆွေးနွေး ခဲ့တဲ့ အတိုင်း 75 Ω burden resistor ကိုပဲ ပြန်သုံး မယ ဆိုရင် 20 A လောက်ထိ တိုင်းနိုင် တဲ့ energy monitor တစ်ခု ကို ရနိုင် ပါတယ်။ တိုင်းတာ ချင်တဲ့ current ရဲ့ range ကို burden resistor တန်ဖိုး နဲ့ အလွယ် တကူ ပြောင်းလို့ ရ ပါတယ်။

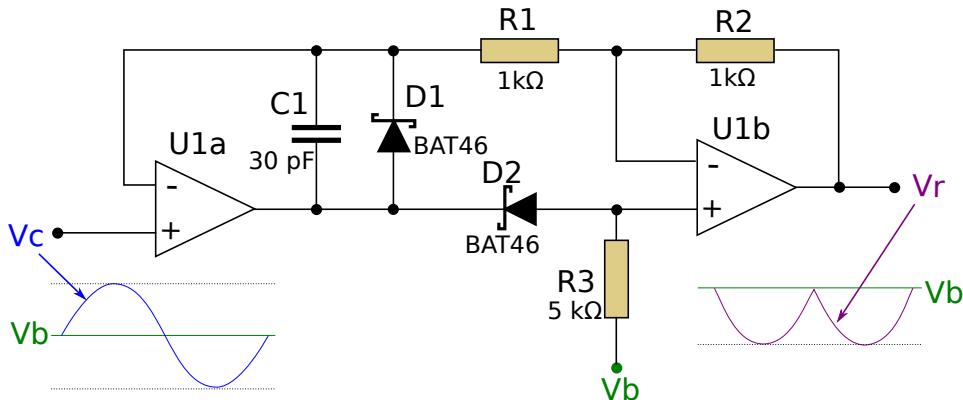


ပုံ ၁၁.၁၃: Voltage reference တစ်ခု တည်ဆောက်ခြင်း။

အခါ op-amp ကို 5 V ပါဝါ supply မြှို့အား နဲ့ သုံးမှာ မို့ သူရဲ့ အထွက် မို့ က 0.8 V ကနေ 3.5 V

ကြားထဲမှာ ပြောင်းလဲ နိုင်ပြီး၊ အလယ် မှတ်က 2.2 V ဝန်းကျင် မှာ ရှိပါတယ်။ အဲဒါ ကြောင့် ပေါင်းထည့်ပေးချင်တဲ့ bias voltage  $V_b$  အတွက် ပိုမို သင့်တော် တဲ့ 2.2 V ထူတ်ပေးမယ့် voltage reference တစ်ခုကို ပုံးပိုး သို့ပြီး တည်ဆောက် နိုင်ပါတယ်။

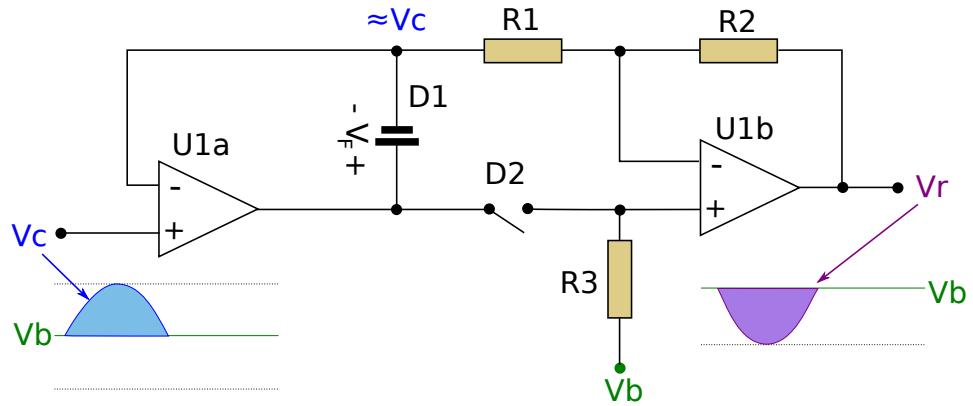
အဲဒီ နောက် precision rectifier တစ်ခုကို ပုံးပိုး သို့ပြီး diode ကို ရွေးချယ် တဲ့ အခါ switching speed မြန်ပြီး၊ forward voltage  $V_F$  နည်းတဲ့ small signal Schottky diode BAT46 ကို ရွေးလိုက် ပါတယ်။ Capacitor  $C_1$  က frequency response ကောင်းအောင် ကူညီ ပေးပါတယ်။



ပုံ ၁၁.၁၄: Fullwave precision rectifier တစ်ခုကို op-amp များဖြင့် တည်ဆောက်ပုံး။

အဝင် ပုံ  $V_c$  က အပေါင်း ဘက်ခြမ်း (ဆိုလို တာက  $V_c > V_b$ ) ဖြစ်နေတဲ့ အချိန်မှာ ပုံ ၁၁.၁၄ မှာ ပြထား သလို D1 က forward ဖြစ်နေပြီး၊ D2 က reverse ဖြစ်နေ ပါမယ်။  $V_b$  ကို ground အနေ နဲ့ သဘော ထားပြီး စဉ်းစား မယ်ဆို ပထမ op-amp က voltage follower ပုံစံ မျိုး လုပ်ဆောင် ပြီး သူရဲ့ အနုတ် အဝင် မှာ ရှိတဲ့ ပို့အား က  $\approx V_c$  ဖြစ်နေ ပါမယ်။ အဲဒီ အချိန် မှာ ဒုတိယ op-amp က inverting amplifier ပုံစံ ဖြစ်သွား တာမို့ rectified voltage  $V_r$  ကို အောက်ပါ အတိုင်း ဖော်ပြ နိုင် ပါတယ်။

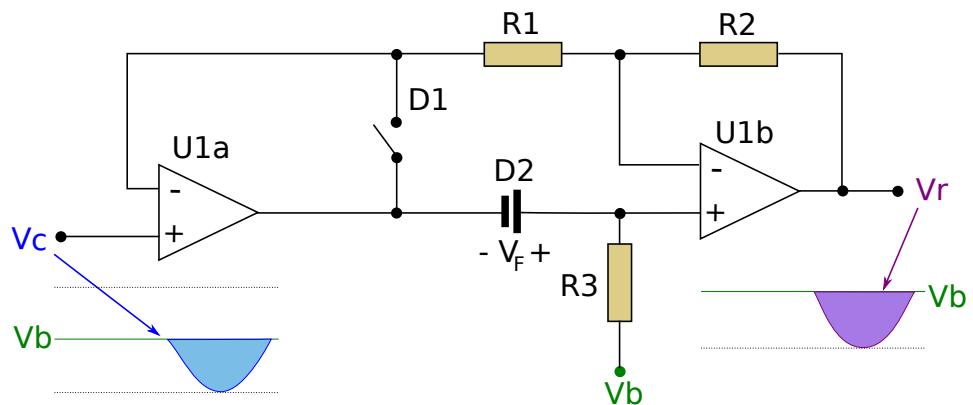
$$V_r = -\frac{R_2}{R_1} \times V_c \quad (၁၁.၁၀)$$



ပုံ ၁၁.၁၅: အဝင် အပေါင်း ဖြစ်ချိန် တွင် precision rectifier က inverting amplifier အနေဖြင့်  
လုပ်ဆောင်ပြီး အထွက် တွင် အနှစ် အနေဖြင့် ထုတ်ပေးသည်။

$R_1 \neq R_2$  ရဲ့ တန်ဖိုး က တူတာ မို့ အထွက် က အဝင် ရဲ့ ပြောင်းပြန် အနှစ် အခြမ်း ရ လာ ပါမယ်။

$$V_r = -V_c \quad (၁၁.၁၁)$$



ပုံ ၁၁.၁၆: အဝင် အနှစ် ဖြစ်ချိန် တွင်လည်း voltage follower အနေဖြင့် အနှစ် ထုတ်ပေးသည်။

အဝင် ဖို့  $V_c$  က အနှစ် ဘက်ခြမ်း (ဆိုလို တာက  $V_c < V_b$ ) ဖြစ်နေတဲ့ အချိန်မှာ ပုံ ၁၁.၁၆ မှာ ပြထားသလို D1 က reverse ဖြစ်နေပြီး၊ D2 က forward ဖြစ်နေပါမယ်။ အဲဒီ အချိန် မှာ high input impedance အဝင် တွေနဲ့ ဆက်ထားတဲ့  $R_1$  နဲ့  $R_2$  မှာ current မစီးတဲ့ အတွက် voltage drop က သူည့် ဖြစ်နေပါမယ်။ ဒါကြောင့် op-amp နှစ်ခု စလုံးက voltage follower ပုံစံ ဖြစ် နေပြီး အထွက်  $V_r$  က အဝင်  $V_c$  နဲ့ တန်ဖိုး အတူတူ အနှစ် ပဲ ထွက်လာ မှာ ဖြစ်ပါတယ်။

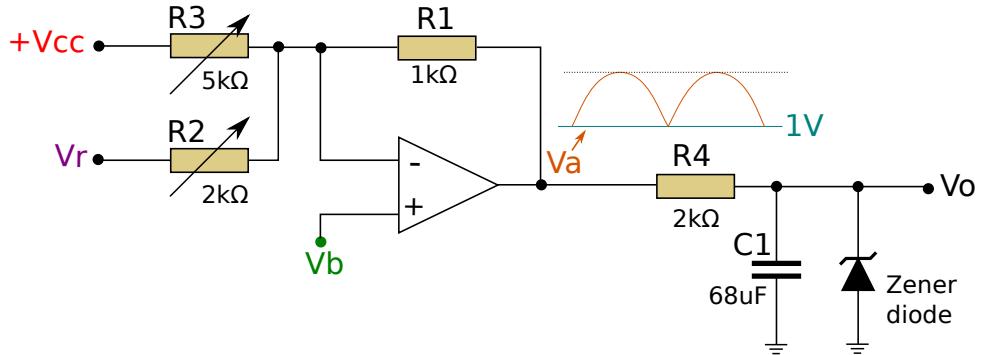
ဒီ configuration မှာ ပထမ op-amp ရဲ့ အထွက် peak to peak voltage က အဝင်  $V_c$  ထက်  $2V_F$  ပိုများ တာကို တွေ့နိုင် ပါတယ်။ ဒါကြောင့် အဝင်  $V_c$  အထွက် အများဆုံး လက်ခံ နိုင်တဲ့ peak to peak voltage တန်ဖိုးကို ၁၁.၃.၅ အတိုင်း တွက်နိုင် ပါတယ်။

$$V_{c_{p-p}} = V_{cc} - 1.5 - 0.8 - 2V_F \quad (၁၁.၁၂)$$

$V_{cc}$  တန်ဖိုး 5 V သုံးတဲ့ အချိန် Schottky diode ရဲ့  $V_F$  ကို 0.25 V လို့ ယူဆ လိုက်ရင်  $V_{c_{p-p}}$  ရဲ့ အများဆုံး တန်ဖိုးကို 2.2 V လို့ ရှိနိုင် ပါတယ်။ ညီမျှခြင်း ၁၁.၉ မှာ အဲဒီ တန်ဖိုး ထည့်လိုက် ရင် အများဆုံး တိုင်းတာ နှင့်တဲ့ အဝင် current  $I_{i_{rms}}$  ကို 20 A လို့ တွေ့ရ မှာ ဖြစ် ပါတယ်။

### ၁၁.၃.၅ Calibration

ဒီ energy monitor က ရဲ့ တိုင်းတာ မှုံကို တိကျ အောင် calibration လုပ်လို့ ရပါတယ်။ သူနဲ့ ဆက်သွယ် မယ့် microcontroller ဒါမှ မဟုတ် SBC ရဲ့ ပရိုဂရမ် မှာ ချိန်ညီ လိုရ ပေမယ့် hardware trimmer တွေ သုံး တာက ပိုမို လွယ်ကူ အဆင်ပြေပြီး လက်တွေ့ ပိုကျ ပါတယ်။ ဒါ ကြောင့် အဝင် rms current နဲ့ အထွက် ဖို့ ကြား ဆက်သွယ် မှုံ ရဲ့ offset နဲ့ scale factor ကို ချိန်ညီ ဖို့ အထွက် inverting op-amp adder တစ်ခု ကို ပုံ မှာ ပြထားသလို ထပ်ဖြည့် လိုက် ပါမယ်။ အဲဒီ မှာ တည်ပြုမဲ့ တဲ့ အထွက် ဖို့အား ရဖို့ low pass filter တစ်ခု ကို ပါ ထည့်ထား ပါတယ်။ အထွက် မှာ zener diode တစ်ခု ကို သုံးပြီး ဖို့အား ကို 1.8 V တို့ 3.3 V တို့ စတဲ့ တန်ဖိုး တွေထက် မကျော် အောင် limit လုပ်နိုင် ပါသေးတယ်။



ပုံ ၁၁.၁၇: Calibration အတွက် op-amp adder နှင့် filter, voltage clipper တို့ အတွက် schematic ပါ။

Op-amp ရဲ့ အထွက် ဖို့  $V_a$  ကို ညီမျှခြင်း ၁၁.၁၃ နဲ့ ဖော်ပြနိုင် ပါတယ်။

$$V_a = V_b - \frac{R_1}{R_2}(V_r - V_b) - \frac{R_1}{R_2}(V_{cc} - V_b) \quad (၁၁.၁၃)$$

အဝင် current တန်ဖိုး မရှိတဲ့ အချိန် ထွက်ချင် တဲ့ offset voltage က ၁ V ထားချင် တာမူး အဲဒီ ညီမျှခြင်း မှာ  $V_b$  တန်ဖိုး 2.2 V သုံးပြီး၊  $V_{cc}$  တန်ဖိုး +5 V နဲ့  $V_r$  ရဲ့ quiescent တန်ဖိုး 2.2 V သုံး လိုက် တဲ့ အခါ  $R_3$  တန်ဖိုး ကို 2.3 kΩ လို ရလာ ပါမယ်။ ဒါကြောင့် အဲဒီ နေရာ မှာ အနီး စပ်ဆုံး mid point ရှိတဲ့ 5 kΩ trimmer ကို သုံးလိုက် ပါမယ်။

ညီမျှခြင်း အရ  $V_{c_{p-p}}$  ရဲ့ peak to peak အများဆုံး တန်ဖိုးက 2.2 V ဖြစ်တာ မို့  $V_a$  ရဲ့ peak တန်ဖိုး  $V_p$  က အများဆုံး 1.1 V ထိ ရှိနိုင် ပါတယ်။ အဲဒီ အချိန်မှာ အဝင် တိုင်းချင် တဲ့ current  $I_{i_{rms}}$  က 20 A ဖြစ်ပြီး၊ နောက်ဆုံး ပျမ်းမျှ အထွက်  $V_o$  က Beaglebone Black တို့ Odroid-XU4 တို့လို SBC တွေရဲ့ analog input reference တွေနဲ့ ပါ အဆင်ပြု နိုင်တဲ့ 1.8 V ဖြစ်အောင် ဒီဇိုင်း လုပ် ပါမယ်။ Offset voltage တန်ဖိုး 1 V ကို ဖယ်လိုက် ရင် အဝင် ကြောင့် ဖြစ်လာ တဲ့ ပျမ်းမျှ ဖို့အား တန်ဖိုး အထွက်  $V_d$  က 0.8 V ပါ။  $V_d$  နဲ့  $V_p$  ရဲ့ တန်ဖိုး ဆက်သွယ် ချက် ကို ညီမျှခြင်း ၁၁.၁၅ နဲ့ ဖော်ပြနိုင် ပါတယ်။

$$V_d = \frac{V_p}{\pi} \int_{\omega t=0}^{\pi} \sin(\omega t) d\omega t \quad (၁၁.၁၄)$$

$$V_d = \frac{2V_p}{\pi} \quad (၁၁.၁၅)$$

$V_d$  တန်ဖိုး 0.8 V အစား သွင်းလိုက် တဲ့ အခါ  $V_p$  က 1.26 V လို့ ရလာ ပါမယ်။ Inverting op-amp အတွက်  $R_1$  နဲ့  $R_2$  အချိုးကို voltage အချိုးနဲ့ အောက်ပါ အတိုင်း ဖော်ပြု နိုင် ပါတယ်။ အဲဒီ အခါ ရှိရမယ့်  $R_2$  တန်ဖိုး ကို 870 Ω ဆိုပြီး ရလာ တာမို့၊ သူ နေရာ မှာ 2 kΩ trimmer ကို သုံးလိုက် ပါမယ်။

$$\frac{R_1}{R_2} = \frac{1.26}{1.1} \quad (၁၁.၁၆)$$

DC တန်ဖိုး ရ အောင် သုံးထား တဲ့ low pass filter အတွက် cutoff frequency  $f_c$  ကို အောက်ပါ ညီမျှခြင်း ၁၁.၁၇ နဲ့ ဖော်ပြု နိုင် ပါတယ်။

$$f_c = \frac{1}{2\pi RC} \quad (၁၁.၁၇)$$

အဲဒီ မှာ R တန်ဖိုး 2 kΩ နဲ့ C တန်ဖိုး 68 μF သုံးလိုက် တဲ့ အခါ cutoff frequency ကို  $\approx 1$  Hz ရ ပါတယ်။ ထိုအား ကို ကန်သတ် ဖို့ အတွက် voltage clipper မှာ 1.8 V Zener diode ကို သုံးနိုင် ပါတယ်။

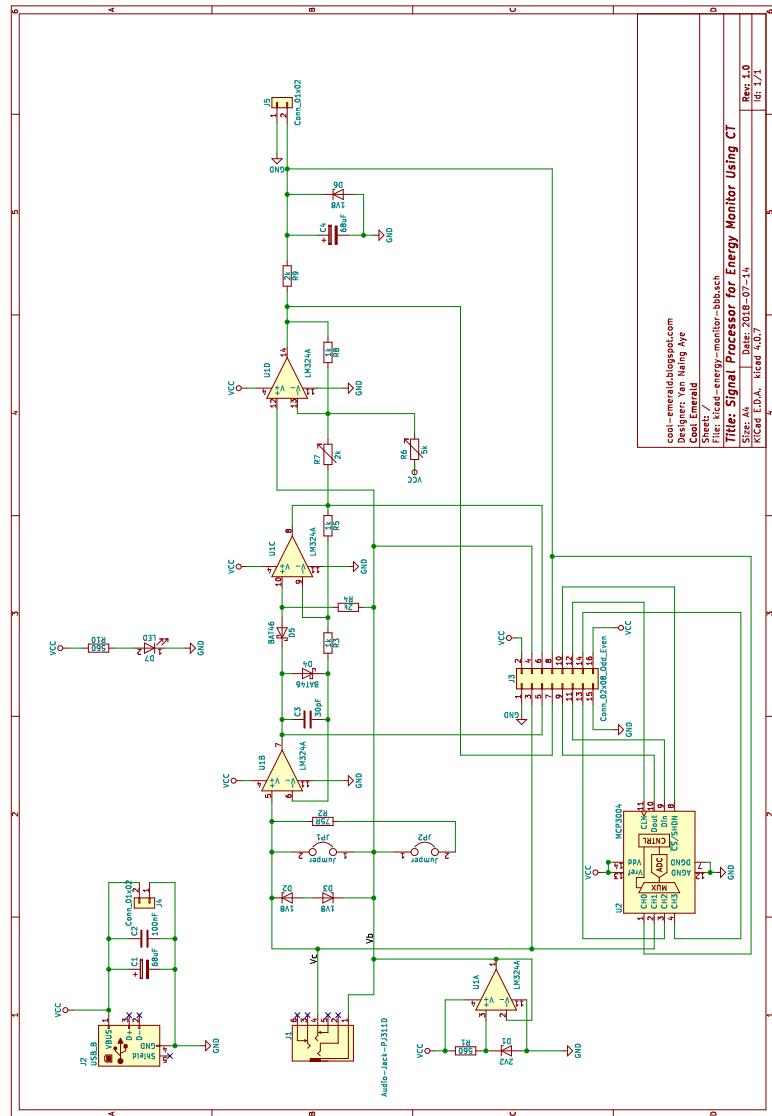
### Steps

Calibration လုပ်တဲ့ အခါ ပထမ အဆင့် အနေနဲ့ ဘာ load မှာ မရှိ တဲ့ အချိန်  $V_o$  ကို 1 V ရအောင်  $R_3$  trimmer ကို လှည့်ပြီး ချိန် ပါမယ်။ ပြီးတဲ့ အခါ ကိုယ့်မှာ ရှိတဲ့ တန်ဖိုး သိတဲ့ load ကို ဖွင့်ပြီး၊ အထွက်  $V_o$  ကိုက်ညီ အောင်  $R_3$  trimmer ကို လှည့်ပြီး ချိန်ရ ပါမယ်။ ဥပမာ 2000 W ရှိတဲ့ hair dryer ကို ဆိုရင် 230 V အတွက် current တန်ဖိုး ကို 8.7 A လို့ ရ ပါမယ်။ အဲဒီ အချိန်မှာ ရှိ ရမယ့် အထွက် ဖို့ ကို ညီမျှခြင်း ၁၁.၁၈ သုံးပြီး ရှာ လိုက်ရင် ချိန်ရ မယ့်  $V_o$  ကို 1.35 V လို့ ရ လာ ပါမယ်။

$$V_o = 1 + 0.8 \times \frac{I_{rms}}{20} \quad (၁၁.၁၈)$$

### ၁၁.၃.၆ Schematic circuit

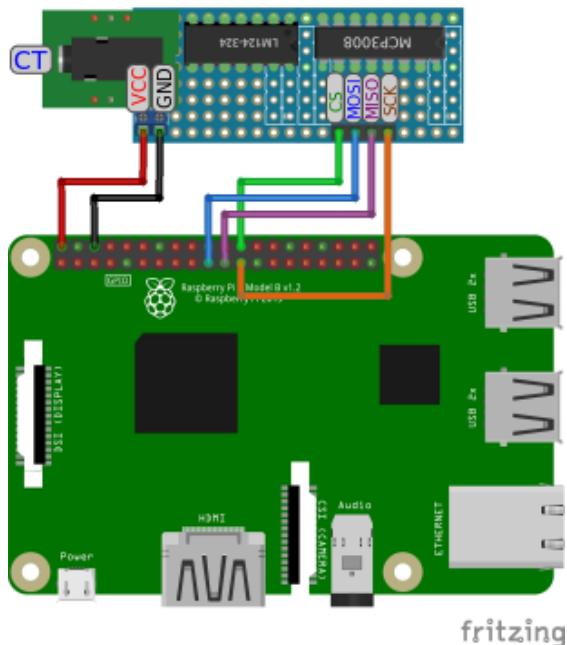
Signal processor က ထွက်လာ တဲ့ အထွက် ကို BeagleBone Black တို့လို SBC တွေရဲ့ analog input သုံးပြီး ဖတ်လို ရသလို၊ Arduino တို့လို microcontroller တွေ သုံးပြီး ဖတ်လို လည်း ရပါတယ်။ Analog input မရှိ တဲ့ Raspberry Pi အတွက် တော့ MCP3004 တို့လို ADC chip တစ်ခုခု ခံ သုံးဖို့ လို ပါလိမ့် မယ်။ ရှုံးက ပြောခဲ့ တဲ့ အစိတ် အပိုင်း တွေ အားလုံး ကို စုပေါင်း ပြီးတဲ့ အခါ ရလာ တဲ့ နမူနာ schematic circuit တစ်ခု ကို ပုံ ၁၁.၁၈ မှာ ပြထား ပါတယ်။



## ပုံ ၁၁.၁၈: Signal processor နှင့် schematic circuit ပုံ။

### ၁၁.၃.၇ C++ ဖြင့်သုံးခြင်း

Signal processor က 5 V power supply နဲ့ အသုံးပြုနိုင် တဲ့ အတွက်၊ သူ့ အတွက် Vcc ကို RPi ရဲ့ 5 V pin နဲ့ ချိတ်ပေး နိုင် ပါတယ်။ Signal processor ရဲ့ အတွက် ဖို့  $V_o$  ကို ဖတ်တဲ့ ADC ရဲ့ SPI interface pin တွေကို RPi နဲ့ ပဲ ပြထား သလို ဆက်သွယ် လိုက် ပါမယ်။



ပဲ ၁၁.၉၉: Energy Monitor ကို Raspberry Pi ၏ SPI interface ဖြင့် ဆက်သွယ်ခြင်း။

ADC ကို ဖတ်ဖို့ အတွက် အပိုင်း ၄.၃ မှာ ဖော်ပြ ခဲ့တဲ့ CE\_SPI class module ကို အလွယ် တကူ ပြန်သုံး နိုင် ပါတယ်။ အဲဒီ နောက် wxWidgets timer ကို သုံးပြီး တစ် စက္နာနဲ့ တစ်ခါ sample ကောက် ပေးတဲ့ ရှိုးရှင်းတဲ့ C++ နူးနာ em.cpp ကို စာရင်း ၁၁.၄ မှာ ဖော်ပြ ထားပါ တယ်။ သူက တစ်မီနှစ် စာ ဒေတာ တွေကို web interface မှာ ဖော်ပြ တဲ့ အခါ အသုံး ပြုနိုင် အောင် eseconds.php ဆိုတဲ့ ဖိုင် မှာလည်း သိမ်းပေး ပါတယ်။ မိနစ် အပြင် hourly တို့၊ daily ဒေတာ တို့ကို လည်း အလွယ် တကူ ထပ်ဖြည့် နိုင် ပါတယ်။

```

1 // File: em.cpp
2 // Description: Sampler for energy monitor
3 // WebSite: http://cool-emerald.blogspot.com
4 // MIT License (https://opensource.org/licenses/MIT)
5 // Copyright (c) 2018 Yan Naing Aye

```

```
6
7 #include <wx/wx.h>
8 #include <string>
9 #include <fstream>
10 #include "ce_spi.h"
11 class MyFrame : public wxFrame
12 {
13 public:
14     MyFrame(const wxString& title);
15     void OnTimer(wxTimerEvent& event);
16     int Write(string mes);
17 private:
18     int m_data[60];
19     wxStaticText *m_lbl;
20     wxTextCtrl *m_txt;
21     CE_SPI *m_ai;
22     wxTimer m_timer;
23     wxDECLARE_EVENT_TABLE();
24 };
25 const int ID_LABEL = 102;
26 const int ID_TXT = 103;
27 const int ID_TIMER = 104;
28 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
29 EVT_TIMER(ID_TIMER, MyFrame::OnTimer)
30 wxEND_EVENT_TABLE()
31 MyFrame::MyFrame(const wxString& title)
32     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(250, 150))
33     , m_timer(this, ID_TIMER)
34 {
35     m_lbl = new wxStaticText(this, ID_LABEL, wxT("Analog input:"), wxPoint
36         (20,15), wxSize(100,25));
37     m_txt = new wxTextCtrl(this, ID_TXT, wxT("0"), wxPoint(140,10), wxSize
38         (50,25));
39     m_ai = new CE_SPI();
40     m_ai->Begin();
41     Centre();
```

```

40     for (int i = 0; i < 60; i++) m_data[i] = 0;
41     m_timer.Start(1000);
42 }
43
44 class MyApp : public wxApp
45 {
46     public:
47         virtual bool OnInit();
48 };
49
50 IMPLEMENT_APP(MyApp)
51 bool MyApp::OnInit()
52 {
53     MyFrame *myFrame = new MyFrame(wxT("Energy Monitor"));
54     myFrame->Show(true);
55
56     return true;
57 }
58 void MyFrame::OnTimer(wxTimerEvent& WXUNUSED(event))
59 {
60     static int i = 0;
61     i=++i%60;
62
63     int x=0;
64     m_ai->tx[0]=0x01;//Send start bit - 0000 0001
65     m_ai->tx[1]=0x80;//read channel 0 - | s/~d | d2 d1 d0 | x x x x |
66     // b7= single/~differential = 1 for single ended
67     // d2 d1 d0 = 0 for channel 0
68     m_ai->tx[2]=0x00; // don't care = x x x x x x x
69     m_ai->Transfer(3);
70     x = m_ai->rx[1]<<8 | m_ai->rx[2]; //read channel 0
71     x&=0x03FF;//mask out invalid bits
72
73     float a = 205; // offset = 1023.0 / 5 = 205 N/V;
74     float k = 28; // scale factor = (20 * 230) / (0.8*205) => 28 VA/N
75     float val = x;

```

```

76     float w = k*(val - a);
77
78     m_data[i] = int(w);
79     string str = "";
80     for (int j = 0, k = i+1; j++,k++) {
81         k %= 60;
82         str += "[" + to_string(j + 1) + "," + to_string(m_data[k]) + "]";
83         if (j >= 59) break;
84         str += ",";
85     }
86     m_txt->Clear();
87     m_txt->AppendText(wxString::Format(wxT("%d"), m_data[i]));
88     this->Write(str);
89 }
90 int MyFrame::Write(string mes)
91 {
92     ofstream wfile;
93     int r = -1;
94     string fpath = "./eoseconds.php";
95     wfile.open(fpath.c_str(), std::fstream::out);
96     if (wfile.is_open()) {
97         wfile << mes << endl;
98         r = 0;
99     }
100    wfile.close();
101    return r;
102 }
```

စာရင်း ၁၁.၄: em.cpp

ပရိုဂရမ် ကို build and run လုပ်ဖို့ အတွက် စာရင်း ၁၁.၅ မှာ ဖော်ပြထားတဲ့ em-bar.sh ကို အောက်က အတိုင်း သုံးနှင့် ပါတယ်။

```
$ ./em-bar.sh
```

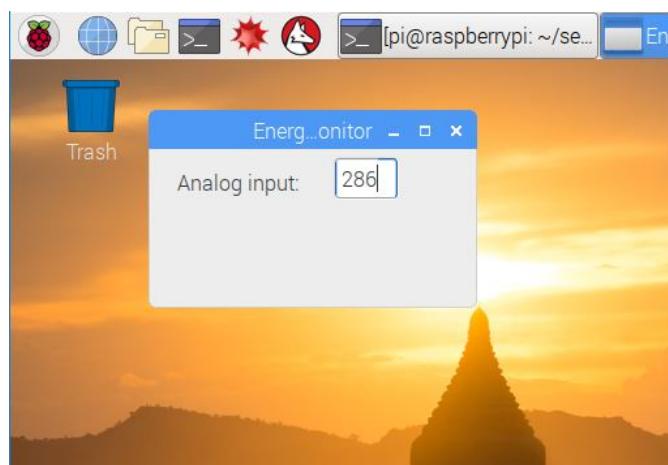
```

1 #!/bin/bash
2
3 echo "Compiling..."
4 unamestr=`uname -m`
5 if [[ "$unamestr" == 'x86_64' ]]; then
6     echo "on x86_64"
7     g++ em.cpp `wx-config --cxxflags --libs std` -o em -std=c++11
8     echo "CWD: $PWD"
9     gnome-terminal -x sh -c './em'
10 elif [[ "$unamestr" == 'armv7l' ]]; then
11     echo "on armv7l"
12     g++ em.cpp `wx-config --cxxflags --libs std` -o em -std=c++11
13     echo "CWD: $PWD"
14     gksudo ./em
15 fi
16 echo "Done."

```

စာရင်း ၁၁.၂: em-bar.sh

ပရိုဂရမ် ရဲ့ GUI ကို ပုံ ၁၁.၂၀ မှာ ပြထား ပါတယ်။



ပုံ ၁၁.၂၀: em.cpp အဲ GUI ။

## ၁၁.၃.၈ Web Interface

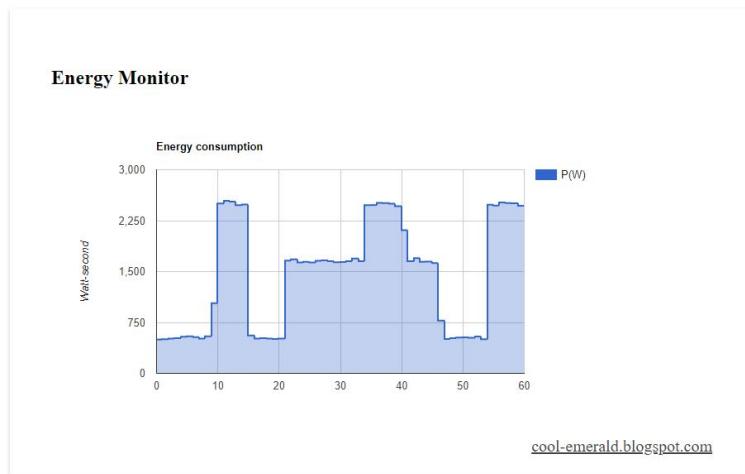
Energy monitor က ဖတ်လို ရတဲ့ ဒေတာ တွေကို RPi ရဲ့ web server မှာ PHP web page တစ်ခု အနေ နဲ့ တိုက်ရိုက် ဖော်ပြ နိုင် ပါတယ်။ အဲဒီ web interface က ကွန်ပျူတာ ဖြစ်ဖစ်၊ ဖုန်း နဲ့ပဲ ဖြစ်ဖစ် internet ဆက်သွယ်မှု ရှိတဲ့ ဘယ်နေ ရာက မဆို လှမ်းကြည့် လို ရနိုင် ပါတယ်။ အပိုင်း ၉.၅ မှာ ဆွေးနွေး ခဲ့တဲ့ gchart ကို အသုံးပြု ထားတဲ့ ဂိုးရှင်း တဲ့ web page နမူနာ em.php ကို စာရင်း ၁၁.၆ မှာ ဖော်ပြ ထားပြီး၊ သူရဲ့ web UI ကို ပုံ ၁၁.၂၁ မှာ တွေ့နိုင် ပါတယ်။

```

1 <html>
2   <head>
3     <meta http-equiv="refresh" content="1">
4   <title>Energy Monitor</title>
5   <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"
6         >
7   </script>
8   <script type="text/javascript">
9     google.charts.load('current', {'packages':['corechart']});
10    google.charts.setOnLoadCallback(drawChart);
11    function drawChart() {
12      var data = google.visualization.arrayToDataTable([
13        ['T(s)', 'P(W)'],
14        <?php include 'econds.php'; ?>]);
15      var options={title:'Energy consumption',vAxis:{title: 'Watt-second'}};
16      var chart = new google.visualization.SteppedAreaChart(document.
17        getElementById('chart_div'));
18      chart.draw(data, options);
19    }
20  </script>
21 </head>
22 <body>
23 <div style="margin-left: auto; margin-right: auto; position: relative;
24   width: 800px; padding: 50px; box-shadow: 0 2px 6px rgba(100,100,100,0.3);">
25   <h2>Energy Monitor</h2>
26   <div id="chart_div" style="width: 700px; height: 400px;"></div>
27   <a style="float: right; font-size: 20px; color: #444444;" href="http://cool-emerald.blogspot.com/">cool-emerald.blogspot.com</a>
28 </div>
```

27 </body>  
 28 </html>

စာရင်း ၁၁.၆: em.php



ပုံ ၁၁.၆: em.php နဲ့ UI ။

## အကိုးအကားများ

- [LH18] Trystan Lea and Glyn Hudson. CT sensors - An Introduction. 2018. url: <https://learn.openenergymonitor.org/electricity-monitoring/ct-sensors/introduction>.
- [Ye13] Ting Ye. Precision Full-Wave Rectifier, Dual-Supply. 2013. url: <http://www.ti.com/lit/ug/tidu030/tidu030.pdf>.
- [Ana15] Analog Devices. Low Voltage Temperature Sensors: TMP35/TMP36/TMP37. 2015. url: [http://www.analog.com/media/en/technical-documentation/data-sheets/TMP35\\_36\\_37.pdf](http://www.analog.com/media/en/technical-documentation/data-sheets/TMP35_36_37.pdf).

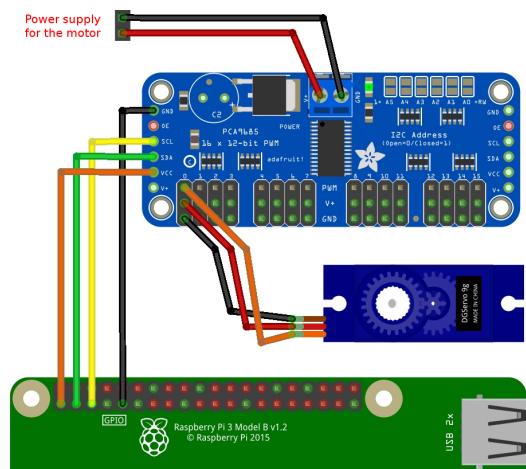
- [Ana17] Analog Devices. AD8346 Low Cost, Low Power, True RMS-to-DC Converter. 2017. url: <http://www.analog.com/en/products/analog-functions/rms-to-dc-converters/ad8346.html>.
- [Ana18] Analog Devices. ADE9153A Energy Metering IC with Autocalibration. 2018. url: <http://www.analog.com/en/products/analog-to-digital-converters/integrated-special-purpose-converters/energy-metering-ics/ADE9153A.html>.
- [STM16] STMicroelectronics. LIS3DH MEMS digital output motion sensor: ultra-low-power high-performance 3-axis nano accelerometer. 2016. url: <http://www.st.com/en/mems-and-sensors/lis3dh.html>.
- [Tex17] Texas Instruments. LM358, LM258, LM158, LM2904 Dual Operational Amplifiers. 2017. url: <http://www.ti.com/lit/ds/symlink/lm258a.pdf>.
- [YHD18] YHDC. SCT013-000-100A-50mA Split Core Current Transformer. 2018. url: <http://en.yhdc.com/product/SCT013-401.html>.

# အခန်း ၁၂

## Actuators

### ၁၂.၁ Servo motor ကိုထိန်းချုပ်ခြင်း

Raspberry Pi ကို သုံးပြီး servo motor အသေးလေး တစ်ခု ကို ထိန်းချုပ် ကြည့်ပါမယ်။ သူမှာ ပါဝါ အတွက်  $V_m$  (အနီရောင်ဝါယာ) နဲ့ GND (အညီရောင်ဝါယာ) ရယ်၊ control အတွက် ငုတ် (လိမ္မာ်ရောင်ဝါယာ) ရယ် စုစုပေါင်း ငုတ် သုံးခု ပါပါတယ်။ အခုံ ဒီ [SG90 Tower Pro micro servo motor](#) လေးက ၉ g ပဲ လေးပြီး၊ အလုပ်လုပ် တဲ့ ဗိုအား က 4.8 V ကနေ 6 V အထိ ဖြစ် ပါတယ်။ လုညွှေပေးနိုင် တဲ့ အမြန်နှုန်းက  $500^{\circ}/\text{s}$  ဖြစ် ပါတယ်။ သူကို Raspberry Pi နဲ့ ဆက်သွယ်တဲ့ ပုံကို ပုံ ၁၂.၁ မှာ ပြထား ပါတယ်။ အပိုင်း ၄.၄.၂ မှာ ဖော်ပြ ခဲ့တဲ့ PCA9685 PWM ကို ပြန်သုံး ထားပါတယ်။



ပုံ ၁၂.၁: Servo motor ကို ထိန်းချုပ်ခြင်း။

ဒီ servo motor လေးကို ထိန်းချုပ် တဲ့အခါ 50 Hz PWM wave ကို သုံးမှာ ဖြစ်လို့ သူရဲ့ period က 20 ms ဖြစ်ပါတယ်။ Pulse width 0.6 ms က မောင်တာ ရဲ့  $-90^\circ$  ဖြစ်ပြီး 2.4 ms က  $+90^\circ$  အသီးသီး အချိုးကျ ဖြစ်ပါတယ် [Hom13]။ အဲဒါက duty cycle 0.03% နဲ့ 0.12% ဖြစ်ပြီး 4096 clock cycles ရှိတဲ့ PCA9685 PWM ၏ 123 နဲ့ 492 clock ticks အသီးသီး ဖြစ်ပါတယ်။ Servo motor ကို  $-90^\circ$  ကနေ  $+90^\circ$  ထိ အဆင့်ဆင့် လှည့်ပြတဲ့ နမူနာ ပရိုဂရမ် servo-motor.cpp ကို စာရင်း ၁၂.၁ မှာ ပြထားပါတယ်။ PCA9685 PWM ကို အသုံးပြု ဖို့ အတွက် ကုဒ်တွေက ce\_pca9685.h (စာရင်း ၁၂.၂) မှာ ဖြစ်ပါတယ်။

```

1 #include <stdio.h>
2 #include "ce_pca9685.h"
3 using namespace std;
4 int main()
5 {
6     CE_PCA9685 servo_motor(1,0x40,50); //i2c-1, address 0x40, frequency 50 Hz
7     for (int i = -90; i <= 90; i++) {
8         servo_motor.SetAngle(0,i);
9         printf("Angle: %d \n",i);
10        usleep(100000);
11    }
12    return 0;
13 }
```

စာရင်း ၁၂.၁: servo-motor.cpp

```

1 //File: ce_pca9685.h
2 //Description: Controlling PCA9685 PWM
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye
6
7 #ifndef CE_PCA9685_H
8 #define CE_PCA9685_H
9
10 #include <string>
11 #include <sstream>
```

```

12 #include <stdio.h>
13 #include <stdlib.h>
14 #include <unistd.h>
15 #include <linux/i2c-dev.h>
16 #include <sys/ioctl.h>
17 #include <fcntl.h>
18 #include <math.h>
19 #include "ce_i2c.h"
20
21 using namespace std;
22
23 class CE_PCA9685 :public CE_I2C
24 {
25     public:
26         CE_PCA9685();
27         CE_PCA9685(int bus_id,int slave_address,int frequency);
28         ~CE_PCA9685();
29         bool SetFrequency(int frequency);
30         bool Begin(int bus_id,int slave_address,int frequency);
31         bool SetDuty(int channel,float percent);
32         bool SetAngle(int channel,float degree); //for servo motor
33     private:
34         char d[8];
35 };
36
37 //-----
38
39 CE_PCA9685::CE_PCA9685():CE_I2C()
40 {
41 }
42
43
44 CE_PCA9685::CE_PCA9685(int bus_id,int slave_address,int frequency):CE_I2C(
    bus_id,slave_address)

```

```

45 {
46     SetFrequency(freq);
47 }
48
49 CE_PCA9685::~CE_PCA9685()
50 {
51 }
52
53
54 bool CE_PCA9685::Begin(int bus_id, int slave_address, int freq)
55 {
56     bool r;
57     r = CE_I2C::Begin(bus_id, slave_address);
58     if(!r){return false;}
59
60     //Output settings
61     //MODE2 register, address 0x01
62     //MODE2 = /Reserved /INVRT/OCH/OUTDRV/OUTNE/
63     //Default = 10 10 10 10 10 1 100 1
64     //Reserved=000=> Reserved
65     //INVRT = 0 => Output logic state not inverted
66     //OCH = 0 => Output change on stop command instead of ACK
67     //OUTDRV = 1 => push-pull output instead of open-drain
68     //OUTNE = 00 => output 0 when output is disabled (i.e. LEDn = 0 when #OE
69     // = 1)
70     d[0] = 1;// address
71     //Value to set = 0000 0100
72     d[1] = 0x04;
73     r=Write(d,2);
74     if(!r){return false;}
75
76     return SetFrequency(freq);
77 }
78
79 bool CE_PCA9685::SetFrequency(int freq)
80 {

```

```

80     bool r;
81
82     //MODE1 register, address 0x00
83     //MODE1 = /Restart/ExtClk/AI /SLEEP/SUB1/SUB2/SUB3/ALLCALL /
84     //Default = 00000000000000000000000000000000
85     //Restart = 0 => Restart disabled
86     //ExtClk = 0 => External clock disabled
87     //AI = 1 => Auto increment enabled
88     //SLEEP = 0 => Normal mode (don't sleep)
89     //SUB1 = 0 => Do not respond to sub address group 1
90     //SUB2 = 0 => Do not respond to sub address group 2
91     //SUB3 = 0 => Do not respond to sub address group 3
92     //ALLCALL = 1 => Respond to all call address
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

```

```

116     if(!r){return false;}
117     //Need at least 500us to wake up from sleep
118     usleep(1000);
119     return true;
120 }
121
122 bool CE_PCA9685::SetDuty(int channel,float percent)
123 {
124     if(percent < 0) percent = 0;
125     else if(percent >100) percent =100;
126
127     if(channel < 0) percent = 0;
128     else if(channel >15) channel =15;
129
130     //Control PWM
131     //LED on = on at 0 of 0-4095 clock cycles
132     //LED off = off at n of 0-4095 clock cycles
133     float cp=percent*4095.0/100.0;
134     int n=floor(cp+0.5); //LED off
135     d[0] = 0x06+(channel*4); //address of LED0 ON_L is at 0x06, 4 bytes per
136     //channel
137     d[1] = 0x00; //ON_L
138     d[2] = 0x00; //ON_H
139     d[3] = char(n & 0xFF); //OFF_L
140     d[4] = char((n>>8) & 0x0F); //OFF_H - (Note: overwriting full off bit 4
141     //to 0)
142     //OFF_H = 0x10 means always off,
143     //always off has higher priority than always on
144     return Write(d,5);
145 }
146
147 bool CE_PCA9685::SetAngle(int channel,float degree) //for servo motor
148 {
149     if(degree<-90) degree=-90;
150     else if(degree > 90) degree =90;

```

```

150 //servo -90 to 90 is duty cycle 3% to 12% @ 50 Hz
151 float duty=(degree+90.0)/180.0;
152 float p = 3.0 + duty*9.0;
153 return SetDuty(channel,p);
154 }
155 //
-----
```

156

```

157 #endif // CE_PCA9685_H
```

စာရင်း ၁၂၂: ce\_pca9685.h

```

Angle: -90
Angle: -89
Angle: -88
Angle: -87
Angle: -86
Angle: -85
Angle: -84
Angle: -83
Angle: -82
Angle: -81
Angle: -80
```

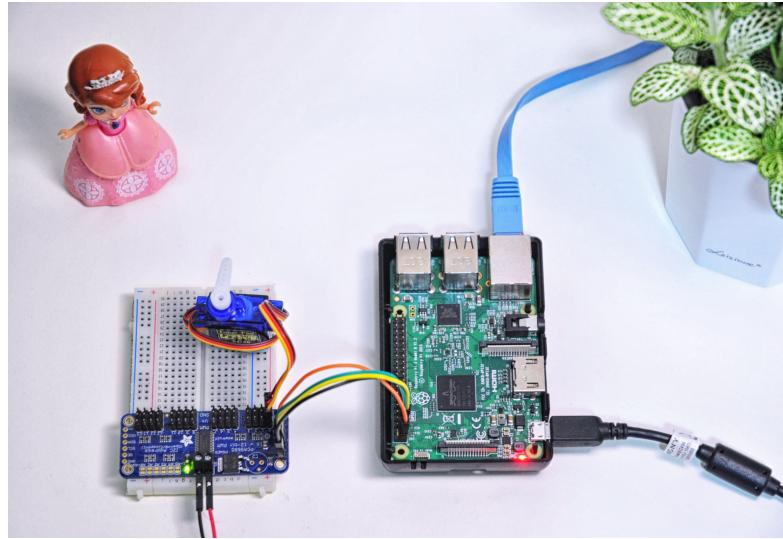
ပုံ ၁၂၂: servo-motor.cpp ရှုလဒ်။

ပရီဂရမ် ကိုအောက်ပါ အတိုင်း build လုပ်ပြီး run နိုင် ပါတယ်။

```

$ g++ servo-motor.cpp -o servo-motor
$ ./servo-motor
```

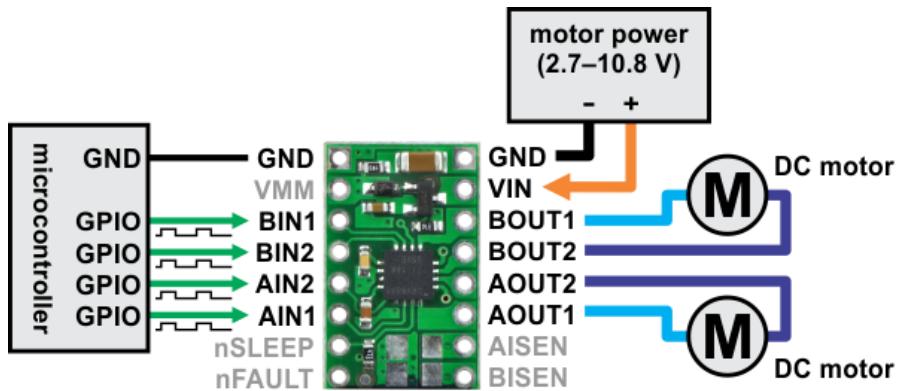
ပရီဂရမ် ရဲ့ ရှုလဒ် နဲ့ micro servo motor ကို Raspberry Pi နဲ့ဆက်ထား တာကို ပုံ ၁၂၃ နဲ့ ပုံ ၁၂၄ မှာ ထွေနိုင် ပါတယ်။



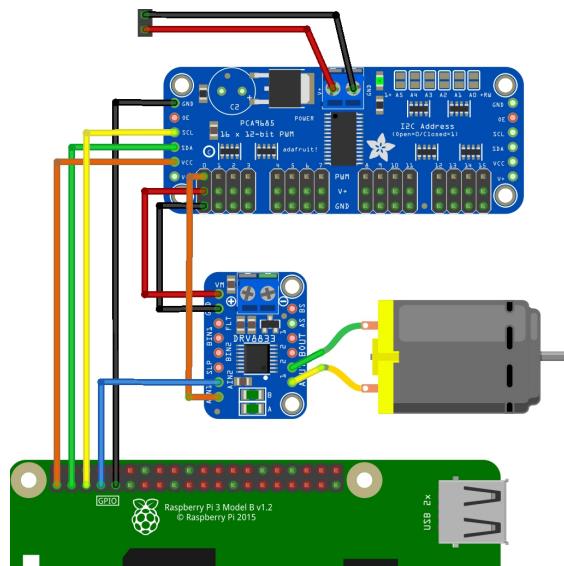
ပုံ ၁၂.၃: Servo motor ကို Raspberry Pi နှင့်ဆက်သွယ်ပဲ။

## ၁၂.၄ DC motor များသုံးခြင်း

DC motor တွေကို ထိန်းချုပ် ဖို့ အတွက် Pololu ကလို DRV8833 Dual Motor Driver လေး ကို အလွယ် တကူ သုံးနိုင် ပါတယ်။ သူက power supply မို့အား 2.7 V ကနေ 10.8 V အထိ ပေးနိုင် ပါတယ် [Ins15a]။ DRV8833 breakout ရဲ့ pin တွေကို ပုံ ၁၂.၄ မှာ ပြထား ပြီး၊ နမူနာ အနေ နဲ့ DC motor တစ်ခု ကို ထိန်းချုပ် ဖို့ Raspberry Pi နဲ့ ဆက်သွယ်ပဲ ကို ပုံ ၁၂.၅ နဲ့ ပုံ ၁၂.၆ မှာ ပြထား ပါတယ်။ သုံးထား တဲ့ DC Motor လေးက 3 V ကနေ 6 V အထိ ပေးလို့ ရပြီး၊ current ကလည်း 100 mA ကနေ 200 mA လောက်ပဲ သုံး ပါတယ်။



ပုံ ၁၂၄: DRV8833 breakout ၏ pin များ။ (pololu.com မှ ပုံဖြစ်သည်။)



ပုံ ၁၂၅: DC motor ကို ထိန်းချုပ်ခြင်း။

အဲဒီ breakout လေး၏ schematic diagram မှာ ပြထား တဲ့ အရ  $V_{in}$  ကို ပေးလိုက် ရင် motor အတွက် ပါ supply လုပ်ပေးနိုင် ပါတယ်။ ထိန်းချုပ်နိုင် တဲ့ မောင်တာ နှစ်ခု A နဲ့ B မှာ channel A ကို နမူနာ အနေနဲ့ သုံးလိုက် ပါတယ်။ အဲဒီ မှာ A အတွက် အဝင် နှစ်ခု  $A_1$  နဲ့  $A_2$  ဆိုပြီး ရှိ ပါတယ်။  $A_1$  မှာ ရှိတဲ့ ဗိုအား များ နေရင် forward direction ကို လည်မှာ ဖြစ်ပြီး၊  $A_2$  က ပိုများ ရင် reverse direction ကို လည်မှာ ဖြစ်ပါတယ်။ အဲဒီ အဝင် နှစ်ခု မှာ ရှိနေတဲ့ ဗိုအား တွေ တူနေရင် မောင်တာ က ရပ် နေ ပါမယ်။ နမူနာ အနေနဲ့ PWM အတွက် တစ်ခု ရယ်၊ digital အတွက် တစ်ခု ရယ် ကို သုံးပြီး DC မောင်တာ

ကိုထိန်းချုပ်ပါမယ်။ PWM channel 0 (LEDO) ကိုမောတာရဲ့ speed ကိုထိန်းဖို့သုံးပြီး pin 7 (GPIO4) ကိုမောတာရဲ့ direction ကိုထိန်းဖို့သုံးလိုက်ပါမယ်။ မောတာရဲ့ full speed ကို FS လိုက်လိုက်ပြီး Duty cycle 0 ကနေ 100% အတွက် တန်ဖိုး 0 ကနေ 1 ကြေား ပြောင်းလဲနေတဲ့ PWM ရဲ့ အတွက် တန်ဖိုးကို d လိုက်လိုက်ပါမယ်။ ဥပမာ duty cycle 50% ဆိုရင် d ရဲ့ တန်ဖိုးက 0.5 ပါ။ အဲဒါ ဆိုရင် ဖောတာရဲ့လည်ပတ်နှစ်း s ကိုအောက်က ပေါ်သော်လည်ပတ်နှစ်း s မှာ အတိုင်းဖော်ပြန်ပါတယ်။

@ccb:cjc: SPI mode 1

Duty cycle Input $A1 = d$	Digital Input $A2 = GPIO4$	Speed Output $s$	Direction Output
0	0	0	stop
d	0	$s = d \times FS$	forward
d	1	$s = (1 - d) \times FS$	reverse
1	1	0	stop

## C.J.J.C C++ ဖြင့်သုံးခြင်း

```
1 #include <stdio.h>
2 #include "ce_pca9685.h"
3 #include "ce_io.h"
4 using namespace std;
5 int main()
6 {
7     CE_PCA9685 dc_motor(1,0x40,100); //i2c-1, address 0x40, frequency 100 Hz
8     CE_IO dir(4,OUTPUT); //GPIO 4 as direction control
9
10    printf("Driving DC motor from 0 percent to 100 percent full speed.\n");
11
12    dir.Write(LOW);
13    printf("Forward direction.\n");
14    for (int i = 0; i <= 100; i+=10) {
15        dc_motor.SetDuty(0,i); //channel,percent duty
```

```

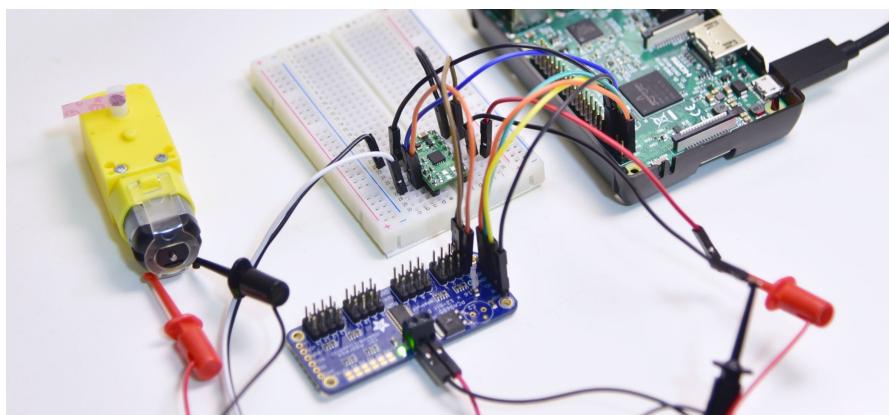
16     printf("Percent FS = %d \n", i);
17     usleep(1000000); //wait
18 }
19
20     dir.Write(HIGH);
21     printf("Reverse direction.\n");
22 for (int i = 0; i <= 100; i+=10) {
23     dc_motor.SetDuty(0,100-i); //channel, percent duty
24     printf("Percent FS = %d \n", (100-i));
25     usleep(1000000); //wait
26 }
27     return 0;
28 }
```

စာရင်း ၁၂.၃: dc-motor.cpp

ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run နိုင် ပါတယ်။

```

$ g++ dc-motor.cpp -o dc-motor
$ sudo ./dc-motor
```



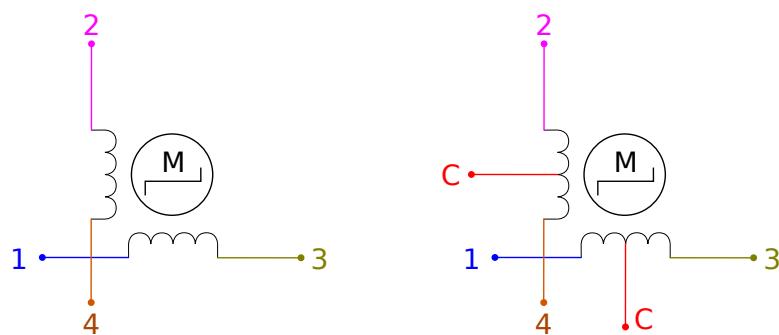
ပုံ ၁၂.၆: DC motor ကို RPi နှင့်ဆက်သွယ်ခြင်း။

## ၁၂.၃ Stepper motor များသုံးခြင်း

Stepper မော်တာ တွေမှာ မော်တာ ဝင်ရီး ရဲ့ လည်ပတ် မှုကို တိကျ တဲ့ step လေးတွေ နဲ့ လှည့်ပတ် လို့ ရပါတယ်။ မော်တာ ရဲ့ လှည့်ပတ် မှုကို square wave pulses လေးတွေ ပေးပြီး တိန်းချုပ် လေ့ ရှိ ပါတယ်။ ဥပမာ တစ်ပတ် တိတိ 360° လှည့်ဖို့ step 36 ခု ရှိတဲ့ stepper motor တစ်ခု မှာ step တစ်ဆင့် လှည့် လိုက်ရင် 10° လည်သွား ပြီး အဲဒီ angle မှာ အတိအကျ ရပ်နေ မှာပါ။ ဒါကြောင့် မော်တာ ကို သတ်မှတ် သလောက် လှည့်ဖို့ encoder စတဲ့ sensor တွေ သုံးဖို့ မလို တဲ့ အတွက် open-loop controller လို့ ပြောလို့ ရပါတယ်။ Stepper မော်တာ တွေမှာ ဝါယာခွေ နှစ်ခွေ ပါလေ့ ရှိပြီး၊ bipolar နဲ့ unipolar ဆိုပြီး နှစ်မျိုး ရှိပါတယ်။

Bipolar stepper motor တွေမှာ ဝါယာ နှစ်ခွေ အတွက် ပုံ ၁၂.၇a မှာ ပြထား သလို ဝါယာ ကြိုး အစ လေးခု ပါ ပါတယ်။ ဝါယာ ခွေရဲ့ သံလိုက် စက်ကွင်း ပြောင်းလဲ ဖို့ သူမှာ စီးဆင်း တဲ့ current ကို direction ပြောင်းဖို့ လိုပြီး အပေါင်း၊ အနှစ် ပြောင်းဖို့ လိုတာမို့ bipolar လို့ ခေါ် တာပါ။

Unipolar stepper motor က ပုံ ၁၂.၇b မှာ ပြထား တဲ့ အတိုင်း ဝါယာ ခွေ နှစ်ခွေ ရဲ့ အလယ်မှာ center tap ရှိတာမို့ ဝါယာ ကြိုးစ ခြောက်ခု ရှိပါတယ်။ အလယ် စ တွေကို မသုံးပဲ အစွမ်း လေးစ ကိုပဲ bipolar stepper motor အနေနဲ့ လည်း သုံးလို့ ရပါတယ်။ တချို့ မော်တာ တွေမှာ တော့ အလယ်စ တွေကို ပူးထား တတ်တာကြောင့် ဝါယာ ငါးစ ပဲ ပါလေ့ ရှိပါတယ်။ Unipolar stepper motor မှာ သံလိုက် စက်ကွင်း ပြောင်းလဲ ဖို့ အပေါင်း၊ အနှစ် ပြောင်း မပေးပဲ ဝါယာ ခွေ တခြမ်းစ ကိုပဲ တလှည့်စ ပြောင်းသုံး တာမို့ unipolar လို့ ခေါ်တာ ပါ။ သူကို ထိန်းကျောင်း မောင်းနှင့် ဖို့ transistor လေးတွေ ကို ပဲ အလွယ်တကူ အဖွင့် အပိတ် ပဲ လုပ်ပြီး ထိန်းနိုင် ပါတယ်။ ဒါပေမယ့် တခါသုံးရင် ဝါယာခွေ တွေရဲ့ တစ်ဝက်ပဲ သုံးတာမို့ bipolar တွေလို့ efficient မဖြစ် ပါဘူး။

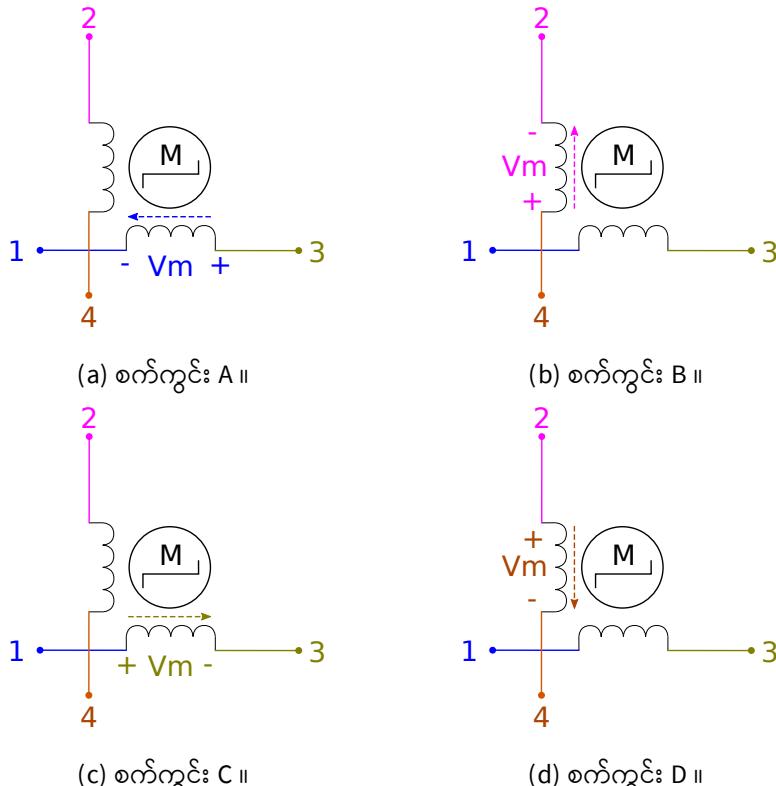


(a) Bipolar stepper motor

(b) Unipolar stepper motor

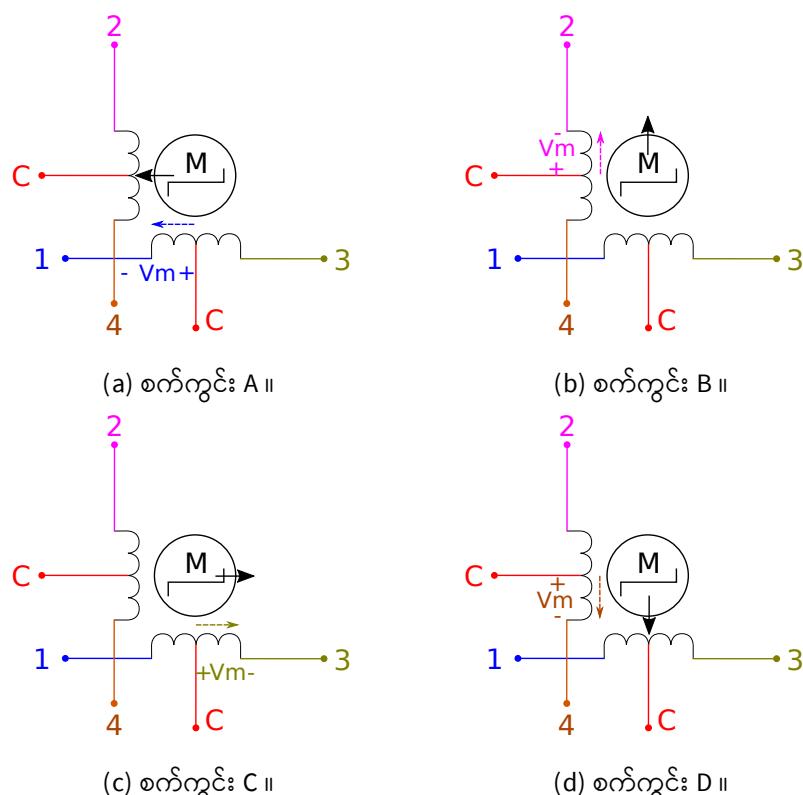
ပုံ ၁၂.၇: Stepper motor နှစ်မျိုး။

သံလိုက် စက်ကွင်း လေးမျိုး ရှိဖို့ Bipolar နဲ့ Unipolar မော်တာ တွေရဲ့ လျှပ်စစ် စီးဆင်းမှု ပုံစံ တွေကို အောက်က ပုံ ၁၂၈ တို့မှာ နှိုင်းယူဉ် ဖော်ပြ ထားပါတယ်။ Bipolar stepper motor အတွက် ဝါယာ အစ တွေမှာ အပေါင်း၊ အနှုတ် ပြောင်းပေး ဖို့ လို ပေမယ့် unipolar stepper motor နှမူနာ မှာ တော့ ဝါယာ အစ တွေမှာ အမြဲ တစ်း အနှုတ် ကိုပဲ နေရာ ပြောင်းပေး တာကို တွေ့ရ မှာပါ။

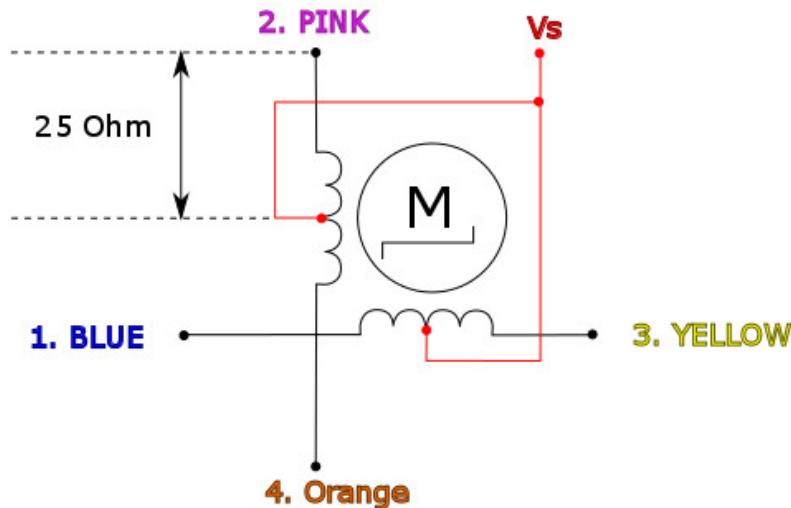


ပုံ ၁၂၈: Bipolar stepper motor အတွက် သံလိုက် စက်ကွင်း ပုံစံ များ။

အသုံးများပြီး၊ အလွယ်တကူ ဝယ်နှိုင်ရုံတင်မက၊ ဈေးလည်း တော်တော်ပေါ့ တဲ့ 28BYJ-48-5V Stepper Motor နဲ့ ULN2003A driver အသုံး ပြုတဲ့ အကြောင်း နှမူနာ အနေနဲ့ ပြောချင် ပါတယ်။ 28BYJ-48-5V Stepper Motor က 5V နဲ့ တိုက်ရှိက်မောင်းနှိုင်ပြီး၊ unipolar stepper motor အမျိုးအစားပါ။ သူ့ရဲ့ schematic ကို အောက်က ပုံ ၁၂၉ မှာပြထားပါတယ်။ ဝါယာ ခွေ ရဲ့ resistance က 50 Ω ရှိတဲ့ အတွက်၊ တစ်ခြမ်းကို 25 Ω ဖြစ် ပါတယ်။ သူ့ရဲ့ torque က 34.3 mN.m ရှိပါတယ်။ 2048 full steps per revolution ဖြစ်ပါတယ်။

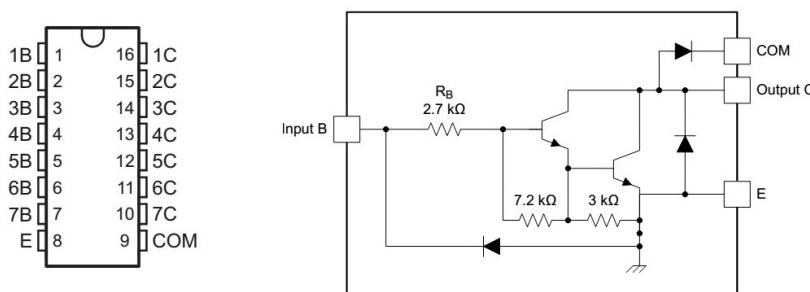


ပုံ ၁၂.၆: Unipolar stepper motor အတွက် သံလိုက် စက်ကွင်း ပုံစံ များ။



ပုံ ၁၂၀။ 28BYJ-48-5V stepper motor ၏ schematic ပုံ။

ULN2003A Transistor Arrays ကတေသူ inductive load တွေကို drive လုပ်ဖို့ ဒီဇိုင်းလုပ်ထားတာမှို့ အထဲမှာ free wheeling diode ပါ ပါပါတယ်။ Darlington pair သံဃွင်းထားတာမှို့ အဝင် ဗို့ 1.4V လောက်ကနေ 30V အထိ ကြိုက်တဲ့ ဗို့နဲ့ တိုက်ရိုက်ဆက်ပြီး ထိန်းနိုင်ပါတယ်။ သူရဲ့ pin တွေနဲ့ တည်ဆောက်ပဲ ကို ပုံ ၁၂၁ မှာ ပြထား ပါတယ်။ အထွက် တစ်ခု စီက 500 mA ထိ မောင်းနှင် ပေးနိုင် ပြီး အများဆုံး ခံနိုင် တဲ့ ဗို့အား က 50 V ဖြစ် ပါတယ်။ အဝင် တွေမှာ လည်း အများဆုံး 30 V ထိ ပေးနိုင် ပါတယ်။



ပုံ ၁၂၁။ ULN2003A driver ၏ pin များနှင့် တည်ဆောက်ပဲ။

Stepper တွက် မောင်းနှင် တဲ့ အခါ Wave drive, Full step drive နဲ့ Half step drive ဆိုပြီး ပုံစံ အမျိုးမျိုး နဲ့ မောင်းနှင်လို့ ရပါတယ်။

### ၁၂.၃.၁ Wave drive

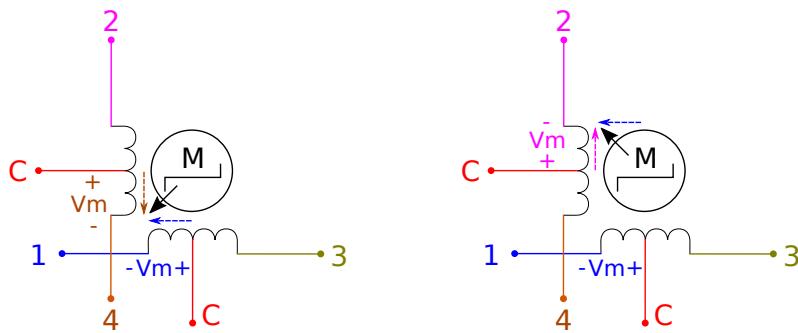
Wave drive ၂ step တစ်ခု ဖိမ္မာ သံလိုက် စက်ကွင်း တစ်မျိုး စီ activate လုပ်ပြီး motor ကို လှည့်သွား တာပါ။ ပုံ ၁၂.၉ မှာ ပြထား သလို စက်ကွင်း A1 B1 C1 D1 A ... စသဖြင့် အဆင့်ဆင့် ပြောင်းပေး လိုက်ရင် မောင်တာ မှာ ပြထားတဲ့ မှုပျား လိုပဲ သံလိုက် စက်ကွင်း ဆွဲငင် တဲ့ အတိုင်း clockwise တစ်ဆင့်ခြင်း လည် သွား ပါလိမ့်မယ်။ အဲဒီလို မဟုတ်ပဲ စက်ကွင်း D1 C1 B1 A1 D ... အစဉ် အတိုင်း ပေးရင် တော့ counter clockwise အတိုင်း လည် မှာပါ။ ဖွင့်ပေး ရမယ့် သံလိုက် စက်ကွင်း နဲ့ မောင်တာ ရဲ့ step ဆက်သွယ် မှု ကို ပေါ်လို့ မှာ ပြထား ပါတယ်။

ပေါ်လို့ မှာ ပြထား ပါတယ်။ ၁၂.၂: Stepper motor ကို wave drive ပုံစံ ဖွင့် မောင်းနှင်ရန် activate လုပ်ပေး ရမည့် သံလိုက် စက်ကွင်း ကို ၁ ဖွင့် ဖော်ပြ ထားသည်။

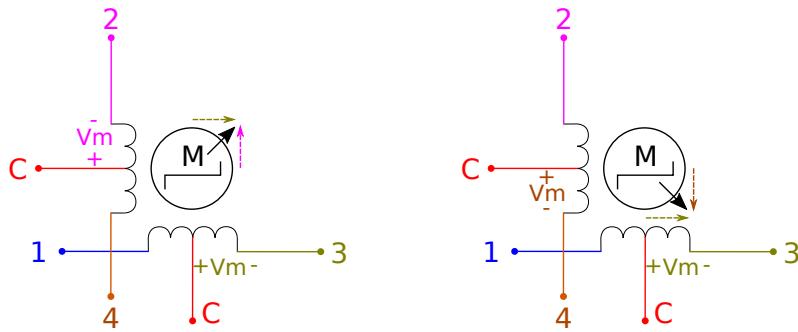
Step	0	1	2	3	0	...
စက်ကွင်း A	1	0	0	0	1	...
စက်ကွင်း B	0	1	0	0	0	...
စက်ကွင်း C	0	0	1	0	0	...
စက်ကွင်း D	0	0	0	1	0	...

### ၁၂.၃.၂ Full step drive

Full step drive မှာတော့ မောင်တာ လည်မယ့် step တစ်ခု စီ အတွက် စက်ကွင်း နှစ်ခု ကို ပြုင်တူ ဖွင့်ပေးတာ မို့ ပို့မို့ အားကောင်းတဲ့ torque ကို ရနိုင် ပါတယ်။ ဖွင့်ပေး တဲ့ သံလိုက် စက်ကွင်း နဲ့ မောင်တာ လည်တဲ့ အဆင့်ဆင့် ကို ပုံ ၁၂.၁ မှာ သရုပ်ဖော် ထားပြီး၊ ဖွင့်ပေး ရမယ့် စက်ကွင်း နဲ့ step ရဲ့ ဆက်သွယ်မှု ကို ပေါ်လို့ မှာဖော်ပြ ထားပါတယ်။



(a) Step 0 အတွက် စက်ကွင်း A နှင့် (b) Step 1 အတွက် စက်ကွင်း A နှင့်  
စက်ကွင်း D ။



(c) Step 2 အတွက် စက်ကွင်း B နှင့် (d) Step 3 အတွက် စက်ကွင်း C နှင့်  
စက်ကွင်း D ။

ဗိုလ်ချုပ်: Full step drive အတွက် သံလိုက် စက်ကွင်း ဖွင့်ပေးပုံ အဆင့်ဆင့်။

သေား ၁၂.၃: Stepper motor ကို full step drive ပုံစံ ဖြင့် မောင်းနှင်ရန် activate လုပ်ပေး ရမည့် သံလိုက် စက်ကွင်း များ ကို ၁ ဖြင့် ဖော်ပြ ထားသည်။

Step	0	1	2	3	0	...
စက်ကွင်း A	1	1	0	0	1	...
စက်ကွင်း B	0	1	1	0	0	...
စက်ကွင်း C	0	0	1	1	0	...
စက်ကွင်း D	1	0	0	1	1	...

### ၁၂.၄ Half step drive

Half step drive ကဲ full step drive နဲ့ wave drive ကို ပေါင်းစပ် အသုံး ပြု ပါတယ်။ Full step drive ရဲ့ step တွေကြား မှာ wave drive ရဲ့ step တွေကို ညုပ်ထည့် လိုက်တာ ပါ။ ဥပမာ full step 0 နဲ့ 1 ကြား မှာ wave drive ရဲ့ step 0 ကို ထည့်လိုက် တယ် ဆိုရင် ပဲ ၁၂.၂၁a ပြထားတဲ့ step ပြီးတဲ့ အခါ တစ်ခု အပြည့် မလည်ပဲ ပဲ ၁၂.၂၁a ပြထား တဲ့ အတိုင်း half step ပဲ လည်မှာ ဖြစ်တဲ့ အတွက် ပိုကောင်း တဲ့ resolution ကို ရ နိုင် ပါတယ်။ Half step drive အတွက် ဖွင့်ပေး ရမယ့် စက်ကွင်း နဲ့ step ရဲ့ ဆက်သွယ်မှု ကို သေား ၁၂.၄ မှာဖော်ပြ ထားပါတယ်။

သေား ၁၂.၄: Stepper motor ကို half step drive ပုံစံ ဖြင့် မောင်းနှင်ရန် activate လုပ်ပေး ရမည့် သံလိုက် စက်ကွင်း များ ကို ၁ ဖြင့် ဖော်ပြ ထားသည်။

Step	0	1	2	3	4	5	6	7	0	...
စက်ကွင်း A	1	1	1	0	0	0	0	0	1	...
စက်ကွင်း B	0	0	1	1	1	0	0	0	0	...
စက်ကွင်း C	0	0	0	0	1	1	1	0	0	...
စက်ကွင်း D	1	0	0	0	0	0	1	1	1	...

### ၁၂.၅ C++ ဖြင့်သုံးခြင်း

Stepper မော်တာ ကို C++ နဲ့ သုံးတဲ့ နမူနာ ပရိုဂရမ် တစ်ခု ကို stepper-motor.cpp (စာရင်း ၁၂.၅) မှာ ပြထား ပါတယ်။ ce\_stepper.h (စာရင်း ၁၂.၅) ကိုသုံး ထား ပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
```

```

3 #include "ce_stepper.h"
4 using namespace std;
5 int main()
6 {
7     CE_STEPPER stepper_motor(CE_HALFSTEP,17,18,23,24); // full step drive,
8     using GPIO 17,18,23,24
9     printf("Forward.\n");
10    stepper_motor.Step(2048,2500); //turn 2048 steps forward with 2500 us
11    period for each step
12    printf("Reverse.\n");
13    stepper_motor.Step(-2048,2500); //turn 2048 steps backward with 2500 us
14    period for each step
15    return 0;
16 }
```

စွာရင်း ၁၂၃: stepper-motor.cpp

```

1 //File: ce_stepper.h
2 //Description: stepper motor driver
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2018 Yan Naing Aye
6
7 #ifndef CE_STEPPER_H
8 #define CE_STEPPER_H
9
10 #include <unistd.h>
11 #include "ce_io.h"
12 using namespace std;
13 typedef enum {CE_WAVEDRIVE=0,CE_FULLSTEP=1,CE_HALFSTEP} CE_STEP_MODE;
14 #define PRINT_MES 0
15
16 class CE_STEPPER{
17     CE_IO pinA;
18     CE_IO pinB;
```

```

19     CE_IO pinC;
20     CE_IO pinD;
21     int step_A[8];
22     int step_B[8];
23     int step_C[8];
24     int step_D[8];
25     int cs; //current step
26     int N; //number of steps for the current mode
27 public:
28     CE_STEPPER();
29     CE_STEPPER(CE_STEP_MODE mode, int gpio_no_A, int gpio_no_B, int gpio_no_C, int
30                 gpio_no_D);
31     ~CE_STEPPER();
32     void Begin(CE_STEP_MODE mode, int gpio_no_A, int gpio_no_B, int gpio_no_C, int
33                 gpio_no_D);
34     void Mode(CE_STEP_MODE mode);
35     void Step(int n, int t);
36 };
37
38
39
40 CE_STEPPER::CE_STEPPER()
41 {
42     Begin(mode, gpio_no_A, gpio_no_B, gpio_no_C, gpio_no_D);
43 }
44
45 CE_STEPPER::~CE_STEPPER()
46 {
47
48 }
49
50 void CE_STEPPER::Begin(CE_STEP_MODE mode, int gpio_no_A, int gpio_no_B, int
51                         gpio_no_C, int gpio_no_D)

```

```

51 {
52     pinA.Begin(gpio_no_A, OUTPUT);
53     pinB.Begin(gpio_no_B, OUTPUT);
54     pinC.Begin(gpio_no_C, OUTPUT);
55     pinD.Begin(gpio_no_D, OUTPUT);
56     cs=0; //start current step at 0
57     Mode(mode);
58 }
59
60 void CE_STEPPER::Mode(CE_STEP_MODE mode)
61 {
62     if(mode==CE_WAVEDRIVE) {
63         step_A[0]=1;step_A[1]=0;step_A[2]=0;step_A[3]=0;
64         step_B[0]=0;step_B[1]=1;step_B[2]=0;step_B[3]=0;
65         step_C[0]=0;step_C[1]=0;step_C[2]=1;step_C[3]=0;
66         step_D[0]=0;step_D[1]=0;step_D[2]=0;step_D[3]=1;
67         N=4;
68     }
69     else if(mode==CE_FULLSTEP) {
70         step_A[0]=1;step_A[1]=1;step_A[2]=0;step_A[3]=0;
71         step_B[0]=0;step_B[1]=1;step_B[2]=1;step_B[3]=0;
72         step_C[0]=0;step_C[1]=0;step_C[2]=1;step_C[3]=1;
73         step_D[0]=1;step_D[1]=0;step_D[2]=0;step_D[3]=1;
74         N=4;
75     }
76     else {
77         step_A[0]=1;step_A[1]=1;step_A[2]=1;step_A[3]=0;
78         step_B[0]=0;step_B[1]=0;step_B[2]=1;step_B[3]=1;
79         step_C[0]=0;step_C[1]=0;step_C[2]=0;step_C[3]=0;
80         step_D[0]=1;step_D[1]=0;step_D[2]=0;step_D[3]=0;
81
82         step_A[4]=0;step_A[5]=0;step_A[6]=0;step_A[7]=0;
83         step_B[4]=1;step_B[5]=0;step_B[6]=0;step_B[7]=0;
84         step_C[4]=1;step_C[5]=1;step_C[6]=1;step_C[7]=0;
85         step_D[4]=0;step_D[5]=0;step_D[6]=1;step_D[7]=1;
86         N=8;
}

```

```

87 }
88 #if PRINT_MES==1
89 for(int i=0;i<N;i++){
90     printf("%d : %d %d %d %d\n",i,step_A[i],step_B[i],step_C[i],step_D[i]
91     ]);
92 }
93 #endif
94 }
95 //-----
96 //Turn stepper motor n steps
97 //with t microseconds period for each step
98 //positive n for forward dir and negative n for backward dir
99 void CE_STEPPER::Step(int n,int t) {
100     int CD=1; //count down
101     if(n<0) {n*=-1; CD=-1;}
102     for(int i=0;i<n;i++) {
103         cs=(cs+N+CD)%N;//find step number
104         pinA.Write(step_A[cs]?HIGH:LOW);
105         pinB.Write(step_B[cs]?HIGH:LOW);
106         pinC.Write(step_C[cs]?HIGH:LOW);
107         pinD.Write(step_D[cs]?HIGH:LOW);
108         #if PRINT_MES==1
109         printf("%d > %d : %d %d %d %d\n",i,cs,step_A[cs],step_B[cs],step_C[cs],
110             step_D[cs]);
111         #endif
112         usleep(t);
113     }
114 //-----
115
116 #endif

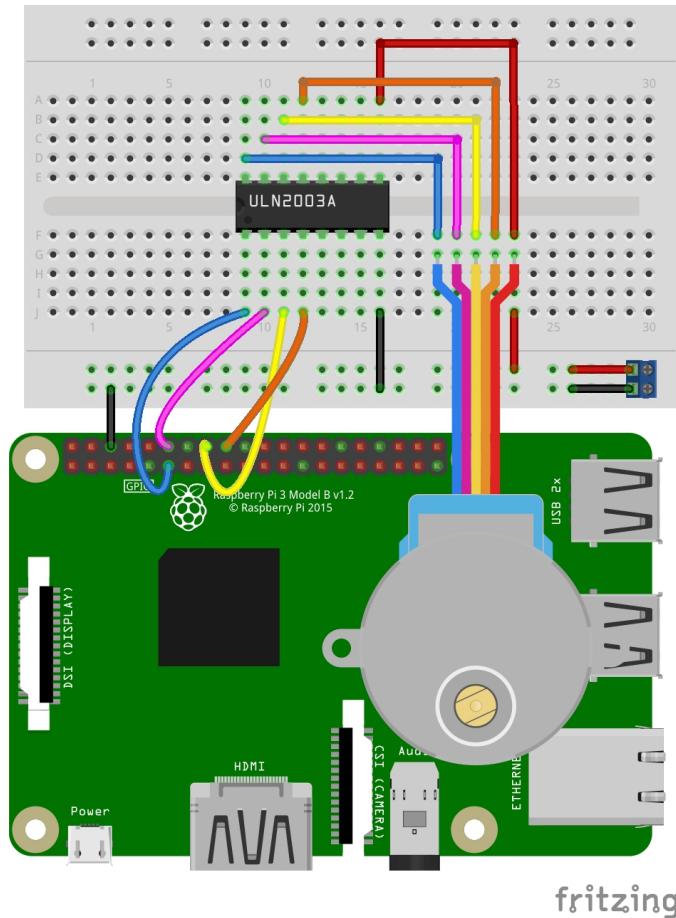
```

ઓન્સુસ ઓફિશિયલ એન્ટ્રી: ce\_stepper.h

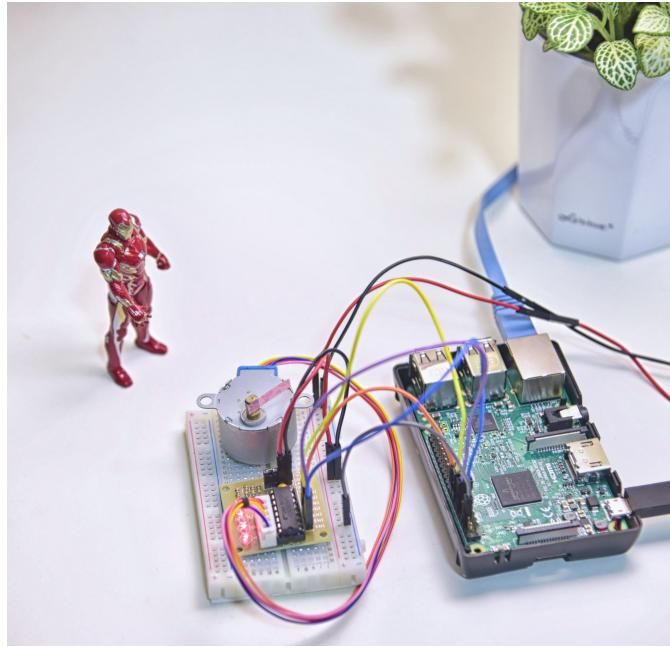
ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး run နိုင် ပါတယ်။

```
$ g++ stepper-motor.cpp -o stepper-motor
$ sudo ./stepper-motor
```

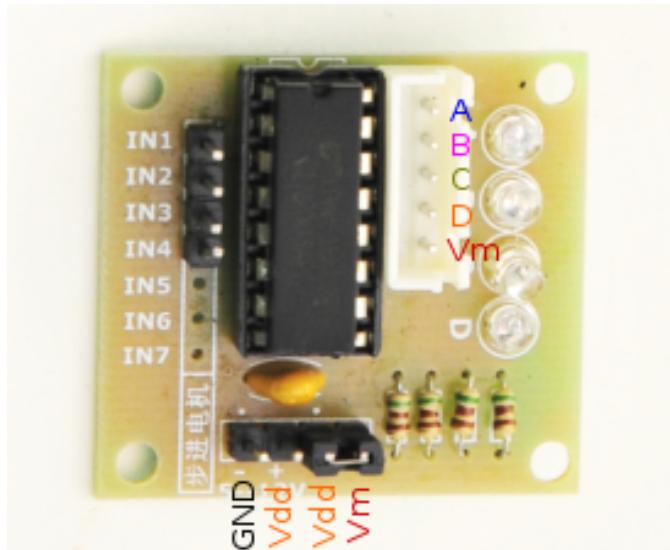
ပရိုဂရမ် အတွက် နမူနာ အနေနဲ့ 28BYJ-48-5V Stepper Motor လေးကို ULN2003A Transistor Arrays နဲ့ မောင်းနှင် အသုံးပြု ထား ပါတယ်။ သူတို့ အတွက် ဆက်သွယ်မှု ပုံစံ ကို ပုံ ပုံ ၁၂၃ နဲ့ ပုံ ပုံ ၁၂၄ မှာ ပြထား ပါတယ်။ A, B, C, D ဆိုတဲ့ collector အတွက် လေးခု ကို မောင်းနှင် ဖို့ GPIO 17, 18, 23 , 24 တိုကို Darlington transistors တွေရဲ့ base အဝင် တွေမှာ ဆက်သွယ် ထား ပါတယ် (ပုံ ၁၂၅)။



ပုံ ၁၂၃: Stepper motor အေး RPi ဖြင့် ဆက်သွယ်ပုံ။



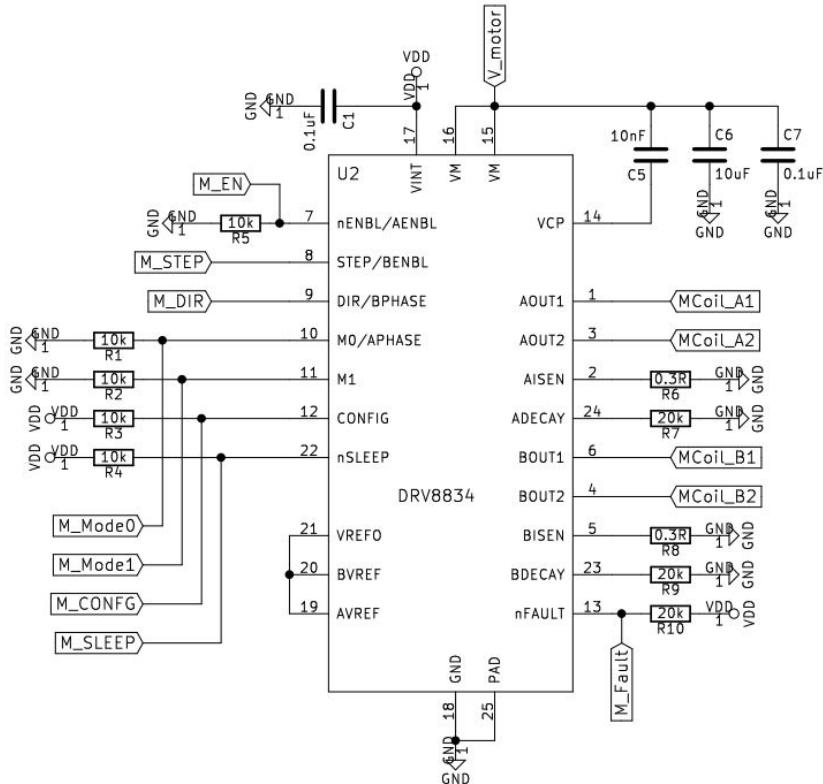
ပုံ ၁၂.၁၃: Stepper motor 28BYJ-48-5V နှင့် Rpi II



ပုံ ၁၂.၁၄: ULN2003A breakout တွင် A, B, C, D ဆိုသည့် collector အတွက် လေးခု အတွက် IN 1,2,3,4 base အဝင် လေးခု ပါရီပြီး၊ မောင်တာ power supply ဖူး Vm နှင့် Vdd (COM) ဖူး အတွက် power supply အတူတူ သုံးလို ပါက jumper ကို ဆက်သွယ် ထားနိုင် သည်။

### ၁၂၃.၅ Stepper Motor Driver များသုံးခြင်း

လက်တွေ့ မှာ stepper motor တွေကို ထိန်းကျောင်း မောင်းနှင် တဲ့ အခါ မှာတော့ သူတို့ အတွက် သီးသန် hardware တွေ ဖြစ်တဲ့ stepper motor driver တွေကို သုံးသင့် ပါတယ်။



ပုံ ၁၂၃.၆: DRV8834 အတွက် နမူနာ schematic ပုံ။

Allegro ရဲ A4988 driver က အသုံးများ ခေတ်စား တဲ့ stepper motor driver တစ်ခု ဖြစ်ပါတယ်။ သူ မောင်းနှင် ပေးနိုင်တဲ့ motor အတွက် supply voltage က 8V - 35V ဖြစ်နေ တဲ့ အတွက် 5 V ဝန်းကျင် သုံးတဲ့ motor တွေနဲ့ တော့ ကိုက်မှာ မဟုတ် ပါဘူး။ chip ရဲ အရွယ်က လည်း 5 mm x 5 mm ဖြစ် ပါတယ်။ TI ရဲ DRV8835 driver ကျပြန်တော့ 2 mm x 3 mm အရွယ်လေးကို သဘောကျပေမယ့် DC motor တွေအတွက်အခိုက ရည်ရွယ်ထားတာမဲ့ stepper motor ကို အလွယ်တကူ ထိန်းဖို့ indexer မပါတာကို တွေ့ရပါတယ်။ DRV8834 stepper motor driver ကတော့ 4 mm x 4 mm အရွယ်၊ motor supply voltage 2.5 V - 10.8 V ရပြီး၊ 1.5 A per coil ရတဲ့ အတွက် နမူနာ အနေ နဲ့ သုံးဖို့ ရွေးချယ်

လိုက်ပါတယ် [Ins15b]။ DRV8834 ကို Indexer mode မှာ ထားပြီး ထိန်းဖို့ အတွက် schematic နမူနာ ပုံစံတရာ့ကို ပုံ ၁၂.၁၆ မှာတွေ့နှင့်ပါတယ်။

အဲဒီမှာ motor supply pin, VM မှာ အနည်းဆုံး 10  $\mu\text{F}$  ချိတ်ဖို့ ညွှန်းပါတယ်။ Charge pump pin, VCP ကိုတော့ 0.01  $\mu\text{F}$  နဲ့ VM ကို လုမ်းဆက်ဖို့ လိုပါတယ်။ နောက် VREFO ကတော့ အကိုးအကား ဗို့ အနေနဲ့ သုံးချင် သုံးလို့ရအောင် IC ကနေ 2 V ထုတ်ပေးထားတဲ့ pin ဖြစ်ပါတယ်။ ဒီနမူနာမှာတော့ coil A နဲ့ coil B တို့အတွက် current limit လုပ်ဖို့ ကိုးကားစရာ့ ဗို့ ပေးရမယ့် AVREF နဲ့ BVREF pin တွေကို VREFO နဲ့ တိုက်ရိုက်ချိတ်ပြီး 2 V ပေးလိုက်ပါတယ်။ ပုံမှန်ဆိုရင်တော့ potentiometer လေးတရာ့ကို voltage divider အနေနဲ့ ခံပြီး ဆက်လေ့ရှုပါတယ်။ AVREF နဲ့ BVREF pin တွေကို ပေးတဲ့ ဗို့  $V_r$  । AISEN နဲ့ BISEN pin တွေမှာ ဆက်မယ့် resistor,  $R_s$  တို့ နဲ့ current limit,  $I_{lim}$  တို့ ရဲ့ ဆက်သွယ်မှု ကို အောက်ကအတိုင်း တွေ့ရပါတယ်။

$$I_{lim} = \frac{V_r}{5.R_s} \quad (၁၂.၁)$$

$R_s$  မှာ စီးတဲ့ current  $I_{lim}$  ကကြီးတဲ့ အတွက် အဲဒီ resistor ရဲ့ power က လုံလောက်အောင်ကြီးဖို့ သတိပြုဖို့ လိုပါတယ်။

နောက်တရာ့က PWM cycle ရဲ့ ဘယ်လောက် ရာခိုင်နှုန်း fast decay ဖြစ်မလဲ ဆိုတာကို ADECAY နဲ့ BDECAY pin တွေမှာ ဆက်ထားတဲ့ resistor တန်ဖိုး နဲ့ ဆုံးဖြတ်ပါတယ်။ ဒီနမူနာမှာတော့ 25% fast decay အတွက် 20 kΩ သုံးလိုက်ပါတယ်။ Indexer mode မှာတော့ ADECAY pin တရာ့ကိုပဲ သုံးပါတယ်။

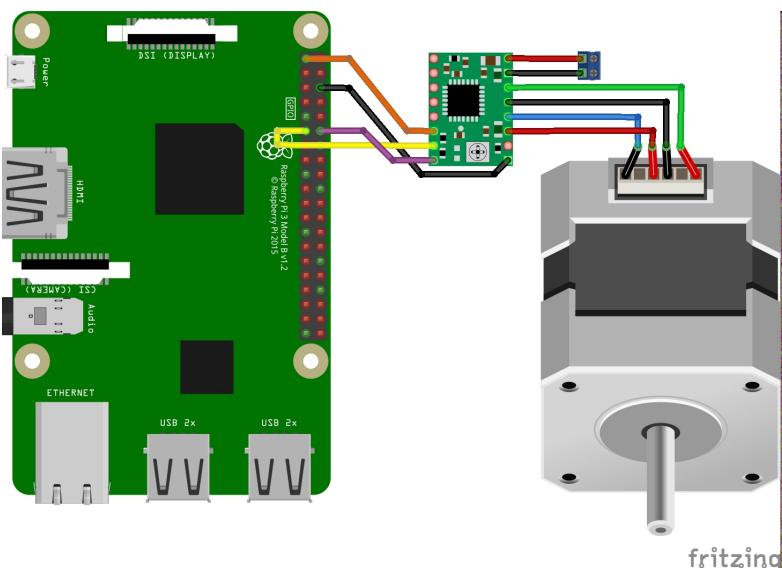
VM က motor အတွက် power supply ဖြစ်ပြီး absolute maximum ratings က -0.3 V ကနေ 11.8 V အထိ ဖြစ်ပါတယ်။ VINT က VM ကို အထဲမှာ regulate လုပ်ပြီး ထွက်လာ တဲ့ အတွက် ပါ။ Logic circuit တွေ အတွက် internal supply အနေနဲ့ သုံးပါတယ်။ Stepper motor ကို ထိန်းဖို့ အတွက် nENBL ကို 0 ပေးပြီး output တွေကို enable လုပ် ထားဖို့ လိုသလို၊ CONFIG pin ကို 1 ပေးပြီး indexer mode မှာ ထားဖို့ လို ပါတယ်။ nSLEEP pin ကို 1 ပေးထား မှ enable ဖြစ်မှာ ဖြစ်ပြီး 0 ပေးလိုက် ရင် internal logic အားလုံး reset ဖြစ်သွား ပါမယ်။

M0 နဲ့ M1 က stepper motor ရဲ့ mode ကို သတ်မှတ်ဖို့ ဖြစ်ပြီး ပေါး ၁၂.၅ မှာ ဖော်ပြထားပါတယ်။ Z က high impedance ကို ဆိုလိုပြီး ဘာမှ မဆက်ပဲ (not connected) ထားရင် လည်း အတူတူ ပါပဲ။

အယား ၁၂၄: Stepper motor mode နှင့် M0, M1 pin များ၏ ဆက်သွယ်မှု။

M1	M0	Step mode
0	0	Full step
0	1	1/2 step
0	Z	1/4 step
1	0	8 microsteps/step
1	1	16 microsteps/step
1	Z	32 microsteps/step

ကျွန်ုတ် အပိုင်းကတော့ သိပ်ပြီး ထွေထွေထူးထူး မရှိပဲ STEP pin မှာ pulse တွေပေးသလောက် motor လည်မှာ ဖြစ်ပြီး DIR pin နဲ့ လည်တဲ့ direction ကို ထိန်းရုံပါပဲ။ Pololu DRV8834 breakout နဲ့ 4.5V bipolar stepper motor ကို မောင်းနှင့် ကြည့်ဖို့ အတွက် ပုံ ၁၂၇ အတိုင်း ချိတ်ဆက် နိုင် ပါတယ်။ GPIO နှစ်ခု ပဲ သုံးဖို့ လိပ်ပြီး pin 11 (GPIO17) ကို STEP အတွက် နဲ့ pin 12 (GPIO18) ကို DIR အတွက် ထိန်းချုပ် ဖို့ သုံးလိုက် ပါမယ်။



ပုံ ၁၂၇: DRV8834 နှင့် BBB ကို ချိတ်ဆက်ခြင်း။

Stepper မောင်တာ ကို DRV8834 stepper motor driver သုံးပြီး ထိန်းချုပ်တဲ့ နမူနာ ပရိုဂရမ် တစ်ခု ကို stepper-driver.cpp (စာရင်း ၁၂၆) မှာ ပြထား ပါတယ်။ GPIO တွေကို ထိန်းချုပ် ဖို့ အတွက် တော့ ce\_io.h (စာရင်း ၃၂) ကို ပြန်သုံး ထား ပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include "ce_io.h"
4 using namespace std;
5
6 CE_IO step_pin(17, OUTPUT); //using GPIO 17 for step
7 CE_IO dir_pin(18, OUTPUT); //using GPIO 18 for dir
8
9 // to move n steps forward with period t
10 // positive n -> forward, negative n -> backward
11 void Step(int n, int t)
12 {
13     int d = 0; //direction
14     if (n<0) { n *= -1; d = 1; }
15     t >>= 1; // divided by 2
16     dir_pin.Write(d ? LOW : HIGH);
17     for (int i = 0; i < n; i++) {
18         step_pin.Write(HIGH);
19         usleep(t);
20         step_pin.Write(LOW);
21         usleep(t);
22     }
23 }
24
25 int main()
26 {
27     printf("Forward.\n");
28     Step(2000,2000); //turn 200 steps forward with 5000 us period for each step
29     printf("Reverse.\n");
30     Step(-2000,2000); //turn 2048 steps backward with 5000 us period for each
31     step
32     return 0;
}

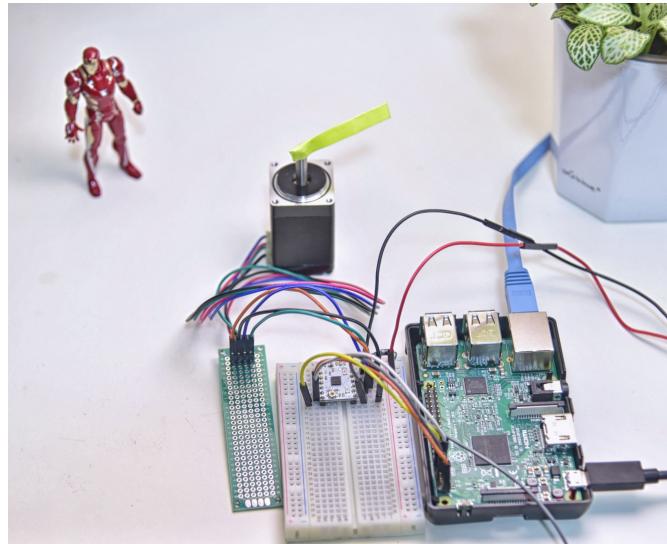
```

օօର୍ଦ୍ଦିଃ ଚାର୍ଜ୍: stepper-driver.cpp

ပရီဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run နိုင် ပါတယ်။

```
$ g++ stepper-driver.cpp -o stepper-driver
$ sudo ./stepper-driver
```

BBB, DRV8834, နဲ့ bipolar stepper motor ဆက်သွယ် ထားတာ ကို ပုံ မှာ တွေ့နိုင် ပါတယ်။



ပုံ ၁၂.၁၈: Stepper motor နဲ့ DRV8834 ဆက်သွယ်ပုံ။

## အကိုးအကားများ

- [Hom13] HomoFaciens. Servos - working principle and homemade types. 2013. url: <https://www.youtube.com/watch?v=v2jpnyKPH64>.
- [Ins15a] Texas Instruments. DRV8833 Dual H-Bridge Motor Driver. 2015. url: <http://www.ti.com/lit/ds/symlink/drv8833.pdf>.
- [Ins15b] Texas Instruments. DRV8834 Dual-Bridge Stepper or DC Motor Driver. 2015. url: <http://www.ti.com/lit/ds/symlink/drv8834.pdf>.

# Appendix A

## Code Listing

### 5.5 Minimal wxWidgets sample

```
1
2 // Name:           minimal.cpp
3 // Purpose:        Minimal wxWidgets sample
4 // Author:         Julian Smart
5 // Modified by:
6 // Created:        04/01/98
7 // RCS-ID:         $Id$
8 // Copyright:      (c) Julian Smart
9 // Licence:        wxWindows licence
10
11
12 // For compilers that support precompilation, includes "wx/wx.h".
13 #include "wx/wxprec.h"
14
15 #ifdef __BORLANDC__
16     #pragma hdrstop
17 #endif
18
19 // for all others, include the necessary headers (this file is usually all
    you
```

```
20 // need because it includes almost all "standard" wxWidgets headers)
21 #ifndef WX_PRECOMP
22     #include "wx/wx.h"
23 #endif
24
25 // the application icon (under Windows and OS/2 it is in resources and even
26 // though we could still include the XPM here it would be unused.)
27 #ifndef wxHAS_IMAGES_IN_RESOURCES
28     #include "./sample.xpm"
29 #endif
30
31 // Define a new application type, each program should derive a class from
32 // wxApp
33 class MyApp : public wxApp
34 {
35     public:
36         // override base class virtuals
37         // -----
38
39         // this one is called on application startup and is a good place for the
40         // app
41         // initialization (doing it here and not in the ctor allows to have an
42         // error
43         // return: if OnInit() returns false, the application terminates)
44         virtual bool OnInit();
45
46     };
47
48 // Define a new frame type: this is going to be our main frame
49 class MyFrame : public wxFrame
50 {
51     public:
52         // ctor(s)
53         MyFrame(const wxString& title);
54
55         // event handlers (these functions should _not_ be virtual)
56         void OnQuit(wxCommandEvent& event);
```

```
53     void OnAbout(wxCommandEvent& event);
54
55 private:
56     // any class wishing to process wxWidgets events must use this macro
57     wxDECLARE_EVENT_TABLE();
58 };
59
60 // IDs for the controls and the menu commands
61 enum
62 {
63     // menu items
64     Minimal_Quit = wxID_EXIT,
65
66     // it is important for the id corresponding to the "About" command to
67     // have
68     // this standard value as otherwise it won't be handled properly under
69     // Mac
70     // (where it is special and put into the "Apple" menu)
71     Minimal_About = wxID_ABOUT
72 };
73
74 // the event tables connect the wxWidgets events with the functions (event
75 // handlers) which process them. It can be also done at run-time, but for the
76 // simple menu events like this the static method is much simpler.
77 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
78     EVT_MENU(Minimal_Quit, MyFrame::OnQuit)
79     EVT_MENU(Minimal_About, MyFrame::OnAbout)
80 wxEND_EVENT_TABLE()
81
82 // Create a new application object: this macro will allow wxWidgets to create
83 // the application object during program execution (it's better than using a
84 // static object for many reasons) and also implements the accessor function
85 // wxGetApp() which will return the reference of the right type (i.e. MyApp
86 // and
87 // not wxApp)
88 IMPLEMENT_APP(MyApp)
```

```

86
87
88 // 'Main program' equivalent: the program execution "starts" here
89 bool MyApp::OnInit()
90 {
91     // call the base class initialization method, currently it only parses a
92     // few common command-line options but it could be do more in the future
93     if ( !wxApp::OnInit() )
94         return false;
95
96     // create the main application window
97     MyFrame *frame = new MyFrame("Minimal wxWidgets App");
98
99     // and show it (the frames, unlike simple controls, are not shown when
100    // created initially)
101    frame->Show(true);
102
103    // success: wxApp::OnRun() will be called which will enter the main
104    // message
105    // loop and the application will run. If we returned false here, the
106    // application would exit immediately.
107    return true;
108 }
109
110 // frame constructor
111 MyFrame::MyFrame(const wxString& title)
112     : wxFrame(NULL, wxID_ANY, title)
113 {
114     // set the frame icon
115     SetIcon(wxICON(sample));
116
117 #if wxUSE_MENUS
118     // create a menu bar
119     wxMenu *fileMenu = new wxMenu;
120
121     // the "About" item should be in the help menu

```

```
121 wxMenu *helpMenu = new wxMenu;
122 helpMenu->Append(Minimal_About, "&About\tF1", "Show about dialog");
123
124 fileMenu->Append(Minimal_Quit, "E&xit\tAlt-X", "Quit this program");
125
126 // now append the freshly created menu to the menu bar...
127 wxMenuBar *menuBar = new wxMenuBar();
128 menuBar->Append(fileMenu, "&File");
129 menuBar->Append(helpMenu, "&Help");
130
131 // ... and attach this menu bar to the frame
132 SetMenuBar(menuBar);
133 #endif // wxUSE_MENUS
134
135 #if wxUSE_STATUSBAR
136 // create a status bar just for fun (by default with 1 pane only)
137 CreateStatusBar(2);
138 SetStatusText("Welcome to wxWidgets!");
139 #endif // wxUSE_STATUSBAR
140 }
141
142
143 // event handlers
144
145 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
146 {
147 // true is to force the frame to close
148 Close(true);
149 }
150
151 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
152 {
153 wxMessageBox(wxString::Format
154 (
155 "Welcome to %s!\n"
156 "\n"
```

```

157         "This is the minimal wxWidgets sample\n"
158         "running under %s.",  

159         wxVERSION_STRING,  

160         wxGetOsDescription()  

161     ),  

162     "About wxWidgets minimal sample",  

163     wxOK | wxICON_INFORMATION,  

164     this);  

165 }
```

စာရင်း ၁.၁: Minimal wxWidgets sample, minimal.cpp

## ၁.J Alpha channel ပါရိုသည့် ပုံရိပ် များကို အသုံးပြုသည့် နမူနာ minimal.cpp

```

1 //File: minimal.cpp
2 //Description: A simple example to use OpenCV with wxWidgets
3 //Author: Yan Naing Aye
4 //Date: 2017 November 07
5 //MIT License - Copyright (c) 2017 Yan Naing Aye
6
7 #include <wx/wx.h>
8 #include <opencv2/opencv.hpp>
9 #include "util.h"
10 #include <string>
11 using namespace std;
12 using namespace cv;
13
14 class MyFrame : public wxFrame
15 {
16     wxStaticBitmap *thiri;
17 public:
18     MyFrame(const wxString& title);
```

```
19
20 };
21 MyFrame::MyFrame(const wxString& title)
22   : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(512, 600))
23 {
24   Centre();
25   thiri = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("./thiri.png"),
26   wxBITMAP_TYPE_PNG), wxPoint(256, 0), wxSize(512,512));
27
28   //From opencv to wx
29   Mat imcv1 = imread("./thiri.png",IMREAD_UNCHANGED);
30   string str = "Channels:" + to_string(imcv1.channels());
31   putText(imcv1, str, Point(100, 100), FONT_HERSHEY_PLAIN, 4.0, CV_RGB(128,
32   0, 128), 4.0);
33   wxBitmap imwx1 = wx_from_mat(imcv1);
34   thiri->SetBitmap(imwx1);
35 }
36
37 class MyApp : public wxApp
38 {
39 public:
40   virtual bool OnInit();
41 };
42 IMPLEMENT_APP(MyApp)
43 bool MyApp::OnInit()
44 {
45   if (!wxApp::OnInit())
46     return false;
47   wxInitAllImageHandlers();
48   MyFrame *frame = new MyFrame(wxT("Simple wxWidgets and OpenCV"));
49   frame->Show(true);
50
51 }
```

---

စာရင်း ၁၂: Alpha channel ပါရိုသည့် ပုံရိပ် များကို အသုံးပြုသည့် နမူနာ minimal.cpp

## ၁၃ Serial.h

```

1 //File: ce_serial.h
2 //Description: Serial communication class for Windows and Linux
3 //WebSite: http://cool-emerald.blogspot.sg/2017/05/serial-port-programming-in
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 // References
8 // https://en.wikibooks.org/wiki/Serial_Programming/termios
9 // http://www.silabs.com/documents/public/application-notes/an197.pdf
10 // https://msdn.microsoft.com/en-us/library/ff802693.aspx
11 // http://www.cplusplus.com/forum/unices/10491/
12
13 #ifndef SERIAL_H
14 #define SERIAL_H
15
16 #include <stdlib.h>
17 #include <stdio.h>
18 #include <string.h>
19 #include <string>
20 using namespace std;
21
22 #if defined (__WIN32__) || defined(_WIN32) || defined(WIN32) || defined(
23     __WINDOWS__) || defined(__TOS_WIN__)
24     #define CEWIN
25
26 #ifdef CEWIN

```

```
27
28 #include <windows.h>
29 #define READ_TIMEOUT 10      // milliseconds
30 inline void delay(unsigned long ms)
31 {
32     Sleep(ms);
33 }
34
35 #else
36
37 #include <unistd.h>
38 #include <fcntl.h>
39 #include <termios.h>
40 #include <sys/ioctl.h>
41 inline void delay(unsigned long ms)
42 {
43     usleep(ms*1000);
44 }
45
46 #endif
47
48 //Class definition
49 class Serial {
50     char rxchar;
51     string port;
52     long baud;
53     long dsizes;
54     char parity;
55     float stopbits;
56     #ifdef CEWIN
57         HANDLE hComm; //handle
58         OVERLAPPED osReader;
59         OVERLAPPED osWrite;
60         BOOL fWaitingOnRead;
61         COMMTIMEOUTS timeouts_ori;
62     #else
```

```
63 long fd;//serial_fd
64 #endif
65 public:
66 Serial();
67 Serial(string Device, long BaudRate, long DataSize, char ParityType, float
68 NStopBits);
69 ~Serial();
70 long Open(void);//return 0 if success
71 void Close();
72 char ReadChar(bool& success);//return read char if success
73 bool WriteChar(char ch);///return success flag
74 bool Write(char *data);//write null terminated string and return success
75 flag
76 bool SetRTS(bool value);//return success flag
77 bool SetDTR(bool value);//return success flag
78 bool GetCTS(bool& success);
79 bool GetDSR(bool& success);
80 bool GetRI(bool& success);
81 bool GetCD(bool& success);
82 bool IsOpened();
83 void SetPort(string Port);
84 string GetPort();
85 void SetBaudRate(long baudrate);
86 long GetBaudRate();
87 void SetDataSize(long nbits);
88 long GetDataSize();
89 void SetParity(char p);
90 char GetParity();
91 void SetStopBits(float nbts);
92 float GetStopBits();
93
94 Serial::Serial()
95 {
96 #ifdef CEWIN
```

```
97     hComm = INVALID_HANDLE_VALUE;
98     port = "\\\\.\\COM1";
99 #else
100    fd = -1;
101    port = "/dev/ttyS0";
102 #endif // defined
103    SetBaudRate(9600);
104    SetDataSize(8);
105    SetParity('N');
106    SetStopBits(1);
107 }
108
109 Serial::Serial(string Device, long BaudRate, long DataSize, char ParityType,
110                 float NStopBits)
111 {
112 #ifdef CEWIN
113     hComm = INVALID_HANDLE_VALUE;
114 #else
115     fd = -1;
116 #endif // defined
117     port = Device;
118     SetBaudRate(BaudRate);
119     SetDataSize(DataSize);
120     SetParity(ParityType);
121     SetStopBits(NStopBits);
122 }
123 Serial::~Serial()
124 {
125     Close();
126 }
127
128 void Serial::SetPort(string Device) {
129     port = Device;
130 }
```

```
132 string Serial::GetPort() {
133     return port;
134 }
135
136 void Serial::SetDataSize(long nbits) {
137     if ((nbits < 5) || (nbits > 8)) nbits = 8;
138     dsize=nbits;
139 }
140
141 long Serial::GetDataSize() {
142     return dsize;
143 }
144
145 void Serial::SetParity(char p) {
146     if ((p != 'N') && (p != 'E') && (p != 'O')) {
147 #ifdef CEWIN
148         if ((p != 'M') && (p != 'S')) p = 'N';
149 #else
150         p = 'N';
151 #endif
152     }
153     parity = p;
154 }
155
156 char Serial::GetParity() {
157     return parity;
158 }
159
160 void Serial::SetStopBits(float nbits) {
161     if (nbits >= 2) stopbits = 2;
162 #ifdef CEWIN
163     else if(nbits >= 1.5) stopbits = 1.5;
164 #endif
165     else stopbits = 1;
166 }
```

```
168 float Serial::GetStopBits() {
169     return stopbits;
170 }
171
172
173 #ifdef CEWIN
174
175 void Serial::SetBaudRate(long baudrate) {
176     if (baudrate < 300) baud = CBR_110;
177     else if (baudrate < 600) baud = CBR_300;
178     else if (baudrate < 1200) baud = CBR_600;
179     else if (baudrate < 2400) baud = CBR_1200;
180     else if (baudrate < 4800) baud = CBR_2400;
181     else if (baudrate < 9600) baud = CBR_4800;
182     else if (baudrate < 14400) baud = CBR_9600;
183     else if (baudrate < 19200) baud = CBR_14400;
184     else if (baudrate < 38400) baud = CBR_19200;
185     else if (baudrate < 57600) baud = CBR_38400;
186     else if (baudrate < 115200) baud = CBR_57600;
187     else if (baudrate < 128000) baud = CBR_115200;
188     else if (baudrate < 256000) baud = CBR_128000;
189     else baud = CBR_256000;
190 }
191
192 long Serial::GetBaudRate() {
193     return baud;
194 }
195
196 long Serial::Open()
197 {
198     if (IsOpened()) return 0;
199 #ifdef UNICODE
200     wstring wtext(port.begin(),port.end());
201 #else
202     string wtext = port;
203 #endif
```

```
204     hComm = CreateFile(wtext.c_str(),
205         GENERIC_READ | GENERIC_WRITE,
206         0,
207         0,
208         OPEN_EXISTING,
209         FILE_ATTRIBUTE_NORMAL | FILE_FLAG_OVERLAPPED,
210         0);
211     if (hComm == INVALID_HANDLE_VALUE) {return 1;}
212
213     if (PurgeComm(hComm, PURGE_TXABORT | PURGE_RXABORT | PURGE_TXCLEAR |
214 PURGE_RXCLEAR) == 0) {return 2;} //purge
215
216     //get initial state
217     DCB dcbOri;
218     bool fSuccess;
219     fSuccess = GetCommState(hComm, &dcbOri);
220     if (!fSuccess) {return 3;}
221
222     DCB dcb1 = dcbOri;
223
224     dcb1.BaudRate = baud;
225
226     if (parity == 'E') dcb1.Parity = EVENPARITY;
227     else if (parity == 'O') dcb1.Parity = ODDPARITY;
228     else if (parity == 'M') dcb1.Parity = MARKPARITY;
229     else if (parity == 'S') dcb1.Parity = SPACEPARITY;
230     else dcb1.Parity = NOPARITY;
231
232     dcb1.ByteSize = (BYTE)dsize;
233
234     if (stopbits==2) dcb1.StopBits = TWOSTOPBITS;
235     else if (stopbits == 1.5) dcb1.StopBits = ONE5STOPBITS;
236     else dcb1.StopBits = ONESTOPBIT;
237
238     dcb1.fOutxCtsFlow = false;
239     dcb1.fOutxDsrFlow = false;
```

```
239     dcb1.fOutX = false;
240     dcb1.fDtrControl = DTR_CONTROL_DISABLE;
241     dcb1.fRtsControl = RTS_CONTROL_DISABLE;
242     fSuccess = SetCommState(hComm, &dcb1);
243     delay(60);
244     if (!fSuccess) {return 4;}
245
246     fSuccess = GetCommState(hComm, &dcb1);
247     if (!fSuccess) {return 5;}
248
249     osReader = { 0 }; // Create the overlapped event.
250     // Must be closed before exiting to avoid a handle leak.
251     osReader.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
252
253     if (osReader.hEvent == NULL) {return 6;} // Error creating overlapped
254     event; abort.
255     fWaitingOnRead = FALSE;
256
257     osWrite = { 0 };
258     osWrite.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
259     if (osWrite.hEvent == NULL) {return 7;}
260
261     if (!GetCommTimeouts(hComm, &timeouts_ori)) { return 8; } // Error
262     getting time-outs.
263     COMMTIMEOUTS timeouts;
264     timeouts.ReadIntervalTimeout = 20;
265     timeouts.ReadTotalTimeoutMultiplier = 15;
266     timeouts.ReadTotalTimeoutConstant = 100;
267     timeouts.WriteTotalTimeoutMultiplier = 15;
268     timeouts.WriteTotalTimeoutConstant = 100;
269     if (!SetCommTimeouts(hComm, &timeouts)) { return 9; } // Error setting
270     time-outs.
271
272     return 0;
273 }
274
275 void Serial::Close()
```

```

272 {
273     if (IsOpened())
274     {
275         SetCommTimeouts(hComm, &timeouts_ori);
276         CloseHandle(osReader.hEvent);
277         CloseHandle(osWrite.hEvent);
278         CloseHandle(hComm); //close comm port
279         hComm = INVALID_HANDLE_VALUE;
280     }
281 }
282
283 bool Serial::IsOpened()
284 {
285     if (hComm == INVALID_HANDLE_VALUE) return false;
286     else return true;
287 }
288
289
290
291 bool Serial::Write(char *data)
292 {
293     if (!IsOpened()) {
294         return false;
295     }
296     BOOL fRes;
297     DWORD dwWritten;
298     long n = strlen(data);
299     if (n < 0) n = 0;
300     else if (n > 1024) n = 1024;
301
302     // Issue write.
303     if (!WriteFile(hComm, data, n, &dwWritten, &osWrite)) {
304         // WriteFile failed, but it isn't delayed. Report error and abort.
305         if (GetLastError() != ERROR_IO_PENDING) {fRes = FALSE;}
306         else { // Write is pending.

```

```
307     if (!GetOverlappedResult(hComm, &osWrite, &dwWritten, TRUE)) fRes =
308         FALSE;
309     else fRes = TRUE;// Write operation completed successfully.
310 }
311 else fRes = TRUE;// WriteFile completed immediately.
312 return fRes;
313 }
314
315 bool Serial::WriteChar(char ch)
316 {
317     char s[2];
318     s[0]=ch;
319     s[1]=0;//null terminated
320     return Write(s);
321 }
322
323 char Serial::ReadChar(bool& success)
324 {
325     success = false;
326     if (!IsOpened()) {return 0;}
327
328     DWORD dwRead;
329     DWORD length=1;
330     BYTE* data = (BYTE*)(&rxchar);
//the creation of the overlapped read operation
332     if (!fWaitingOnRead) {
333         // Issue read operation.
334         if (!ReadFile(hComm, data, length, &dwRead, &osReader)) {
335             if (GetLastError() != ERROR_IO_PENDING) { /*Error*/}
336             else { fWaitingOnRead = TRUE; /*Waiting*/}
337         }
338         else {if(dwRead==length) success = true;}//success
339     }
340
341 }
```

```
342 //detection of the completion of an overlapped read operation
343 DWORD dwRes;
344 if (fWaitingOnRead) {
345     dwRes = WaitForSingleObject(osReader.hEvent, READ_TIMEOUT);
346     switch (dwRes)
347     {
348         // Read completed.
349         case WAIT_OBJECT_0:
350             if (!GetOverlappedResult(hComm, &osReader, &dwRead, FALSE)) /*Error*/
351             }
352         else {
353             if (dwRead == length) success = true;
354             fWaitingOnRead = FALSE;
355             // Reset flag so that another operation can be issued.
356             } // Read completed successfully.
357             break;
358
359         case WAIT_TIMEOUT:
360             // Operation isn't complete yet.
361             break;
362
363         default:
364             // Error in the WaitForSingleObject;
365             break;
366         }
367     return rxchar;
368 }
369
370 bool Serial::SetRTS(bool value)
371 {
372     bool r = false;
373     if (IsOpened()) {
374         if (value) {
375             if (EscapeCommFunction(hComm, SETRTS)) r = true;
376         }
377     }
378 }
```

```
377     else {
378         if (EscapeCommFunction(hComm, CLRRTS)) r = true;
379     }
380 }
381 return r;
382 }

383

384 bool Serial::SetDTR(bool value)
385 {
386     bool r = false;
387     if (IsOpened()) {
388         if (value) {
389             if (EscapeCommFunction(hComm, SETDTR)) r = true;
390         }
391         else {
392             if (EscapeCommFunction(hComm, CLRDTR)) r = true;
393         }
394     }
395     return r;
396 }

397

398 bool Serial::GetCTS(bool& success)
399 {
400     success = false;
401     bool r = false;
402     if (IsOpened()) {
403         DWORD dwModemStatus;
404         if (GetCommModemStatus(hComm, &dwModemStatus)){
405             r = MS_CTS_ON & dwModemStatus;
406             success = true;
407         }
408     }
409     return r;
410 }

411

412 bool Serial::GetDSR(bool& success)
```

```
413 {
414     success = false;
415     bool r = false;
416     if (IsOpened()) {
417         DWORD dwModemStatus;
418         if (GetCommModemStatus(hComm, &dwModemStatus)) {
419             r = MS_DSR_ON & dwModemStatus;
420             success = true;
421         }
422     }
423     return r;
424 }
425
426 bool Serial::GetRI(bool& success)
427 {
428     success = false;
429     bool r = false;
430     if (IsOpened()) {
431         DWORD dwModemStatus;
432         if (GetCommModemStatus(hComm, &dwModemStatus)) {
433             r = MS_RING_ON & dwModemStatus;
434             success = true;
435         }
436     }
437     return r;
438 }
439
440 bool Serial::GetCD(bool& success)
441 {
442     success = false;
443     bool r = false;
444     if (IsOpened()) {
445         DWORD dwModemStatus;
446         if (GetCommModemStatus(hComm, &dwModemStatus)) {
447             r = MS_RLSD_ON & dwModemStatus;
448             success = true;
```

```
449     }
450 }
451 return r;
452 }

453

454 #else //for POSIX

455

456 long Serial::Open(void) {

457

458     struct termios settings;
459     memset(&settings, 0, sizeof(settings));
460     settings.c_iflag = 0;
461     settings.c_oflag = 0;

462

463     settings.c_cflag = CREAD | CLOCAL; //see termios.h for more information
464     if(dsizes==5) settings.c_cflag |= CS5; //no change
465     else if (dsizes == 6) settings.c_cflag |= CS6;
466     else if (dsizes == 7) settings.c_cflag |= CS7;
467     else settings.c_cflag |= CS8;

468

469     if(stopbits==2) settings.c_cflag |= CSTOPB;

470

471     if(parity != 'N') settings.c_cflag |= PARENBT;
472
473     if (parity == 'O') settings.c_cflag |= PARODD;

474

475     settings.c_lflag = 0;
476     settings.c_cc[VMIN] = 1;
477     settings.c_cc[VTIME] = 0;

478

479     fd = open(port.c_str(), O_RDWR | O_NONBLOCK);
480     if (fd == -1) {
481         return -1;
482     }

483     cfsetospeed(&settings, baud);
484     cfsetispeed(&settings, baud);
```

```
485
486     tcsetattr(fd, TCSANOW, &settings);
487
488     int flags = fcntl(fd, F_GETFL, 0);
489     fcntl(fd, F_SETFL, flags | O_NONBLOCK);
490
491     return 0;
492 }
493
494 void Serial::Close() {
495     if(IsOpened()) close(fd);
496     fd=-1;
497 }
498
499 bool Serial::IsOpened()
500 {
501     if(fd==(-1)) return false;
502     else return true;
503 }
504
505 void Serial::SetBaudRate(long baudrate) {
506     if (baudrate < 50) baud = B0;
507     else if (baudrate < 75) baud = B50;
508     else if (baudrate < 110) baud = B75;
509     else if (baudrate < 134) baud = B110;
510     else if (baudrate < 150) baud = B134;
511     else if (baudrate < 200) baud = B150;
512     else if (baudrate < 300) baud = B200;
513     else if (baudrate < 600) baud = B300;
514     else if (baudrate < 1200) baud = B600;
515     else if (baudrate < 2400) baud = B1200;
516     else if (baudrate < 4800) baud = B2400;
517     else if (baudrate < 9600) baud = B4800;
518     else if (baudrate < 19200) baud = B9600;
519     else if (baudrate < 38400) baud = B19200;
520     else if (baudrate < 57600) baud = B38400;
```

```
521     else if (baudrate < 115200) baud = B57600;
522     else if (baudrate < 230400) baud = B115200;
523     else baud = B230400;
524 }
525
526 long Serial::GetBaudRate() {
527     long baudrate=9600;
528     if (baud < B50) baudrate = 0;
529     else if (baud < B75) baudrate = 50;
530     else if (baud < B110) baudrate = 75;
531     else if (baud < B134) baudrate = 110;
532     else if (baud < B150) baudrate = 134;
533     else if (baud < B200) baudrate = 150;
534     else if (baud < B300) baudrate = 200;
535     else if (baud < B600) baudrate = 300;
536     else if (baud < B1200) baudrate = 600;
537     else if (baud < B2400) baudrate = 1200;
538     else if (baud < B4800) baudrate = 2400;
539     else if (baud < B9600) baudrate = 4800;
540     else if (baud < B19200) baudrate = 9600;
541     else if (baud < B38400) baudrate = 19200;
542     else if (baud < B57600) baudrate = 38400;
543     else if (baud < B115200) baudrate = 57600;
544     else if (baud < B230400) baudrate = 115200;
545     else baudrate = 230400;
546     return baudrate;
547 }
548 char Serial::ReadChar(bool& success)
549 {
550     success=false;
551     if (!IsOpened()) {return 0; }
552     success=read(fd, &rxchar, 1)==1;
553     return rxchar;
554 }
555
556 bool Serial::Write(char *data)
```

```
557 {
558     if (!IsOpened()) {return false; }
559     long n = strlen(data);
560     if (n < 0) n = 0;
561     else if(n > 1024) n = 1024;
562     return (write(fd, data, n)==n);
563 }
564
565 bool Serial::WriteChar(char ch)
566 {
567     char s[2];
568     s[0]=ch;
569     s[1]=0;//null terminated
570     return Write(s);
571 }
572
573 bool Serial::SetRTS(bool value) {
574     long RTS_flag = TIOCM_RTS;
575     bool success=true;
576     if (value) {//Set RTS pin
577         if (ioctl(fd, TIOCMBIS, &RTS_flag) == -1) success=false;
578     }
579     else {//Clear RTS pin
580         if (ioctl(fd, TIOCMBIC, &RTS_flag) == -1) success=false;
581     }
582     return success;
583 }
584
585 bool Serial::SetDTR(bool value) {
586     long DTR_flag = TIOCM_DTR;
587     bool success=true;
588     if (value) {//Set DTR pin
589         if (ioctl(fd, TIOCMBIS, &DTR_flag) == -1) success=false;
590     }
591     else {//Clear DTR pin
592         if (ioctl(fd, TIOCMBIC, &DTR_flag) == -1) success=false;
```

```
593 }
594     return success;
595 }
596
597 bool Serial::GetCTS(bool& success) {
598     success=true;
599     long status;
600     if(ioctl(fd, TIOCMGET, &status)== -1) success=false;
601     return ((status & TIOCM_CTS) != 0);
602 }
603
604 bool Serial::GetDSR(bool& success) {
605     success=true;
606     long status;
607     if(ioctl(fd, TIOCMGET, &status)== -1) success=false;
608     return ((status & TIOCM_DSR) != 0);
609 }
610
611 bool Serial::GetRI(bool& success) {
612     success=true;
613     long status;
614     if(ioctl(fd, TIOCMGET, &status)== -1) success=false;
615     return ((status & TIOCM_RI) != 0);
616 }
617
618 bool Serial::GetCD(bool& success) {
619     success=true;
620     long status;
621     if(ioctl(fd, TIOCMGET, &status)== -1) success=false;
622     return ((status & TIOCM_CD) != 0);
623 }
624
625
626 #endif
627
628
```

```
629 #endif // SERIAL_H
```

စာရင်း ၁.၃: Serial.h

## ၁.၆ frame.h

```

1 //File: frame.h
2 //Description: Byte stuffing- sending and receiving frames
3 //Author: Yan Naing Aye
4 //WebSite: http://cool-emerald.blogspot.com
5 //MIT License (https://opensource.org/licenses/MIT)
6 //Copyright (c) 2018 Yan Naing Aye
7
8 // References
9 // http://coolemerald.blogspot.com/2009/09/crc-calculation-in-vb-and-c.html
10
11 #ifndef FRAME_H
12 #define FRAME_H
13
14 #include <stdio.h>
15 #define STX 0x02
16 #define ETX 0x03
17 #define DLE 0x10
18
19 #define TX_BUF_SIZE 128
20 #define RX_BUF_SIZE 128
21 enum RX_STATE { IGNORE, RECEIVING, ESCAPE, RXCRC1, RXCRC2 };
22
23 class Frame {
24     RX_STATE rState;
25 protected:
26     int TxN;//number of transmitting bytes
27     int RxN;//number of receiving bytes
28     char tb[TX_BUF_SIZE];//transmit buffer

```

```
29     char rb[RX_BUF_SIZE];//receiving data
30 public:
31     Frame();
32     int setTxFrame(char* d,int n);
33     unsigned int CRC16CCITT_Calculate(char* s,unsigned char len,unsigned int
34     crc);
35     int getTxN();
36     int getRxN();
37     int receiveRxFrame(char c);//get receiving frame from received char
38     char* getTxBuf();
39     char* getRxBuf();
40 };
41 Frame::Frame():TxN(0),RxN(0),rState(IGNORE){}
42
43 char* Frame::getTxBuf(){
44     return tb;
45 }
46
47 char* Frame::getRxBuf(){
48     return rb;
49 }
50
51 //Prepare transmitting frame
52 int Frame::setTxFrame(char* d,int n)
53 {
54     unsigned int txcrc=0xFFFF;//initialize crc
55     char c;
56     int i=0,j=0;
57     tb[i++]=STX;//start of frame
58     for(j=0;j < n;j++) {
59         c=d[j];
60         if((c==STX) || (c==ETX) || (c==DLE)) tb[i++]=(DLE);
61         tb[i++]=c;
62     }
63     tb[i++]=(ETX);//end of frame
```

```

64
65     txcrc=CRC16CCITT_Calculate(d,n,txcrc);//calculate crc
66     tb[i++]=txcrc & 0xFF;
67     tb[i++]=(txcrc >> 8) & 0xFF;
68     TxN=i;
69     return TxN;
70 }
71
72 //Inputs
73 //s : pointer to input char string
74 //len: string len (maximum 255)
75 //crc: initial CRC value
76
77 //Output
78 //Returns calculated CRC
79 unsigned int Frame::CRC16CCITT_Calculate(char* s,unsigned char len,unsigned
    int crc)
80 {
81     //CRC Order: 16
82     //CCITT(recommendation) : F(x)= x16 + x12 + x5 + 1
83     //CRC Poly: 0x1021
84     //Operational initial value: 0xFFFF
85     //Final xor value: 0
86     unsigned char i,j;
87     for(i=0;i < len;i++,s++) {
88         crc^=((unsigned int)( *s ) & 0xFF) << 8;
89         for(j=0;j<8;j++) {
90             if(crc & 0x8000) crc=(crc << 1)^0x1021;
91             else crc <<=1;
92         }
93     }
94     return (crc & 0xFFFF);//truncate last 16 bit
95 }
96
97 //get number of transmitting bytes
98 int Frame::getTxN()

```

```
99  {
100     return TxN;
101 }
102
103 //get number of transmitting bytes
104 int Frame::getRxN()
105 {
106     return RxN;
107 }
108
109 //process receiving char
110 int Frame::receiveRxFrame(char c)
111 {
112     static char b;
113     unsigned int crc;
114     unsigned int rxcrc=0xFFFF; //initialize CRC
115     switch(rState){
116         case IGNORE:
117             if(c==STX) { rState=RECEIVING; RxN=0; }
118             break;
119         case RECEIVING:
120             if(c==STX) { rState=RECEIVING; RxN=0; }
121             else if(c==ETX){rState=RXCRC1;}
122             else if(c==DLE){ rState=ESCAPE; }
123             else { rb[RxN++]=c; }
124             break;
125         case ESCAPE:
126             rb[RxN++]=c; rState=RECEIVING;
127             break;
128         case RXCRC1:
129             b=c; rState=RXCRC2;
130             break;
131         case RXCRC2:
132             rState=IGNORE;
133             crc=( (int)c << 8 | ((int)b & 0xFF) ) & 0xFFFF; //get received crc
134             rxcrc=CRC16CCITT_Calculate(rb,RxN,rxcrc); //calculate crc
```

```
135     //printf("crc: %x  rxrcrc:%x \n",crc,rxrcrc);
136     if(rxrcrc==crc){return RxN;}//if crc is correct
137     else {RxN=0;}//discard the frame
138
139     break;
140 }
141 return 0;
142 }
143
144 class Frame2:public Frame {
145     char Dt[20];//transmitting data
146 public:
147     Frame2();
148     void printTxFrame();
149     void printRxFrame();
150     void printRxData();
151     void setTxData(float x,float y,float z,float b,float t);
152 };
153
154 Frame2::Frame2():Frame(),Dt(""){}
155
156 //Print out frame content
157 void Frame2::printTxFrame()
158 {
159     printf("Tx frame buffer: ");
160     for(int j=0;j < TxN;j++) printf("%02X ",(unsigned char)tb[j]);
161     printf("\n");
162 }
163
164 //Print out frame content
165 void Frame2::printRxFrame()
166 {
167     printf("Rx data buffer: ");
168     for(int j=0;j < RxN;j++) printf("%02X ",(unsigned char)rb[j]);
169     printf("\n");
170 }
```

```
171  
172 //Set transmitting data  
173 void Frame2::setTxData(float x, float y, float z, float b, float t)  
174 {  
175     *(float*)(Dt)=x;  
176     *(float*)(Dt+4)=y;  
177     *(float*)(Dt+8)=z;  
178     *(float*)(Dt+12)=b;  
179     *(float*)(Dt+16)=t;  
180     Frame::setTxFrame(Dt,20);  
181 }  
182  
183 //Print out received data  
184 void Frame2::printRxData()  
185 {  
186     float x,y,z,b,t;  
187     x=*(float*)(Dt);  
188     y=*(float*)(Dt+4);  
189     z=*(float*)(Dt+8);  
190     b=*(float*)(Dt+12);  
191     t=*(float*)(Dt+16);  
192     printf("Rx data: %f %f %f %f %f \n",x,y,z,b,t);  
193 }  
194  
195 #endif // FRAME_H
```

ወርሃዊ የC: frame.h

## Appendix B

### စကားလုံးဖွင့်ဆိုချက်များ

ADC ..... Analog to Digital Converter

DAC ..... Digital to Analog Converter

DHCP ..... Dynamic Host Configuration Protocol

DNS ..... Domain Name System

GPIO ..... General Purpose Input Output

I2C ..... Inter-integrated circuit

IDE ..... Integrated Development Environment

IoT ..... Internet of things

MAC address ..... Media access control address

OpenCV ..... Open source computer vision

PIR sensor .....	Passive infrared sensor
PWM .....	Pulse-width modulation
RPi .....	Raspberry Pi
SBC .....	Single Board Computer
SFTP .....	SSH File Transfer Protocol or Secure File Transfer Protocol
SPI .....	Serial peripheral interface bus
SSH .....	Secure Shell
VNC .....	Virtual Network Computing
vs15 .....	Visual studio 2017
x86 .....	A family of backward compatible 32 bit instruction set architectures based on the Intel 80386 CPU
x64 .....	The 64-bit version of the x86 instruction set
စက်အမြင် .....	Machine vision
တိုင် .....	Column
တန်း .....	Row
ပုံရိပ်ပြုစပ်ခြင်း .....	Image Processing
ပြောက် .....	Frame

ပြန်နှုံး ..... Frame rate

ပြားစွဲဘက်ထရီ ..... Button cell, coin battery

ဖန်ရှင် ..... Function

ဖယ်တာ ..... Filter

ဖြည့်စွက်အာရုံရိပ် ..... Augmented reality

နာရီတိုက်ခြင်း ..... Clock Synchronization



သေးငယ်ပြီး၊ ပါဝါစား သက်သာ၊ ရွှေးလည်း သက်သာတဲ့ single-board computer (SBC) လေးတွေဟာ robotic system တွေ တည်ဆောက် ရာမှာ အသုံးတည့် ပါတယ်။ ဒါ စာအုပ် မှာတော့ Raspberry Pi ကို အသုံးပြုပြီး hardware တွေနဲ့ interface လုပ်တာ၊ OpenCV ကိုသုံးပြီး image processing လုပ်တာ စတဲ့ အကြောင်း တွေကို ခွေးနေးထားပါတယ်။ Robotic system တွေ အတွက် အခြေခံ ဖြစ်တဲ့ sensor/actuator တွေကို ဆက်သွယ် အသုံးပြုတာ နဲ့ control အတွက် Raspberry Pi ကို အသုံးပြုပုံ အခြေခံ တွေကို ပြုလုပ် စို ရည်ရွယ်ပါတယ်။

---

### အခြားရေးသားထားသောစာအုပ်များ

KiCad အားစတင်အသုံးပြုခြင်း - [yan9a.github.io/KiCad/kicadmm.pdf](https://yan9a.github.io/KiCad/kicadmm.pdf)

OpenCV ကိုလေ့လာခြင်း - [yan9a.github.io/opencv/opencv.pdf](https://yan9a.github.io/opencv/opencv.pdf)

BeagleBoard များအား C++ ဖြင့်အသုံးပြုခြင်း - [yan9a.github.io/beagle/beagle.pdf](https://yan9a.github.io/beagle/beagle.pdf)