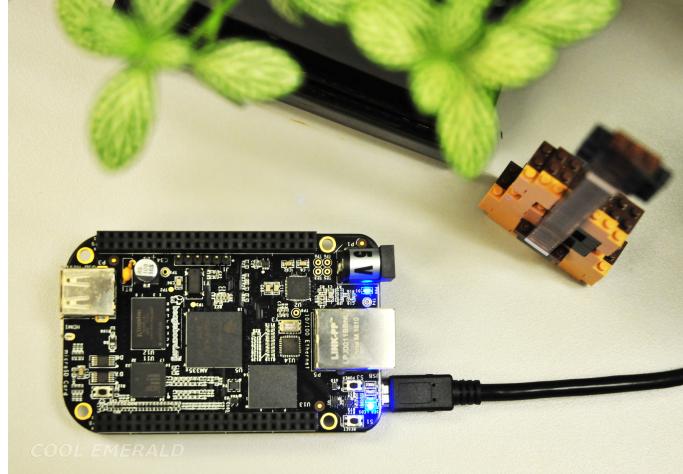


BeagleBoard များအား C++ ဖြင့် အသုံးပြုခြင်း



beagleboard.org

Robotic System များတည်ဆောက်မှုအတွက်

ကိုအေး

စာရေးသူ - Yan Naing Aye @ ကိုအေး

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

တည်းဖွတ်မှု - ပထမအကြိမ်

မတ် ၂၀၁၈

Typeset with \LaTeX and Pyidaungsu font.

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

နောက်ဆုံး တည်းဖွတ်မှု - <https://yan9a.github.io/beagle/beagle.pdf>

စာအုပ်ပါ source code များ - <https://github.com/yan9a/beagle>

မာတိကာ

| | |
|---|----|
| မာတိကာ | i |
| ၁ မိတ်ဆက် | ၁ |
| ၁.၁ နှိပ်လုပ်များ | ၂ |
| ၁.၂ စတင်တပ်ဆင်အသုံးပြုခြင်း | ၅ |
| ၁.၂.၁ ကွန်ပျူးတာဖြင့်ချိတ်ဆက်ခြင်း | ၆ |
| ၁.၂.၂ Cloud9 | ၈ |
| J BoneScript | ၁၁ |
| J.၁ Cloud9 ကို သုံးခြင်း | ၁၁ |
| J.၂ Autorun | ၁၃ |
| J.၃ Digital output ထုတ်ချုပ် hardware များကို တိန်းချုပ်ခြင်း | ၁၄ |
| J.၄ Digital တန်ဖိုးများဖတ်ခြင်း | ၁၈ |
| J.၅ Analog တန်ဖိုးများ ရေးဖတ်ခြင်း | ၂၁ |
| J.၆ Interrupt အသုံးပြုခြင်း | ၂၂ |
| J.၇ Text ဖိုင်များကိုရေးဖတ်ခြင်း | ၂၆ |
| ၂ အခြေခံလုပ်ဆောင်မှုများ | ၂၉ |
| ၂.၁ SSH ဆက်သွယ်အသုံးပြုခြင်း | ၂၉ |
| ၂.၁.၁ Password ကို ပြောင်းခြင်း | ၃၂ |
| ၂.၁.၂ Password ကိုဖြုတ်ခြင်း | ၃၂ |
| ၂.၁.၃ User အသစ်ဖန်တီးခြင်း | ၃၃ |

| | |
|---|-----------|
| ၃.၁.၄ Sudoers | ၃၃ |
| ၃.၁.၅ User ဖျက်ခြင်း | ၃၃ |
| ၃.၂ ဖိုင်များပေးပို့ရယူခြင်း | ၃၄ |
| ၃.၃ ဘုတ်ကို update လုပ်ခြင်း | ၃၆ |
| ၃.၃.၀ eMMC ပေါ်သို့ ထည့်သွင်းခြင်း | ၃၆ |
| ၃.၃.၂ µSD Card ၏ File System Partition ကို ချဲခြင်း | ၄၀ |
| ၃.၄ Backup နှင့် Restore ပြုလုပ်ခြင်း | ၄၃ |
| ၃.၄.၀ Image ဖိုင်များယူရှုကူးခြင်း | ၄၃ |
| ၃.၅ Network ပြင်ဆင်ခြင်း | ၄၅ |
| ၃.၅.၀ Stretch အတွက် Network ပြင်ဆင်ခြင်း | ၄၅ |
| ၃.၅.၂ Wheezy/Jessie အတွက် Network ပြင်ဆင်ခြင်း | ၄၆ |
| ၃.၅.၃ Network ဆက်သွယ်မှုကိုစစ်ခြင်း | ၄၇ |
| ၃.၆ VNC ဖြင့်အသုံးပြုခြင်း | ၄၇ |
| ၃.၇ Autostart | ၅၀ |
| ၃.၈ Locale ပြင်ဆင်ခြင်း | ၅၃ |
| ၃.၉ C/C++ ပရိုဂရမ်ရေးသားခြင်း | ၅၅ |
| ၃.၁၀ Linux Virtual Machine တစ်ခု တပ်ဆင်ခြင်း | ၅၈ |
| ၃.၁၁ Cross Compilation Tool Chain | ၆၂ |
| ၃.၁၂ QEMU - Quick EMULATOR | ၆၆ |
| ၃.၁၃ Code::Blocks IDE တပ်ဆင်ခြင်း | ၆၆ |
| ၃.၁၃.၁ Code::Blocks Project နမူနာ | ၇၀ |
| ၄ Input/Output များအသုံးပြုခြင်း | ၇၇ |
| ၄.၁ Digital IO | ၇၇ |
| ၄.၁.၀ C++ ဖြင့်သုံးခြင်း | ၈၀ |
| ၄.၁.၂ Light sensor ဖြင့်အလင်းရောင်ကိုအာရုံခံခြင်း | ၈၈ |
| ၄.၁.၃ Relay ဖြင့်ပါဝါအဖွင့်အပိတ်ထိန်းချုပ်ခြင်း | ၉၉ |
| ၄.၁.၄ Light Activated Switch | ၉၀ |
| ၄.၂ Analog input များကိုဖတ်ခြင်း | ၉၁ |

| | |
|---|------------|
| ၄.၂.၁ C++ ဖြင့်အသုံးပြုခြင်း | ၉၃ |
| ၄.၂.၂ Temperature sensor ဖြင့်အပူချိန်ကိုအာရုံခံခြင်း | ၉၇ |
| ၄.၃ Analog output ထုတ်ခြင်း | ၁၀၀ |
| ၄.၃.၁ C++ ဖြင့်အသုံးပြုခြင်း | ၁၀၁ |
| ၄.၃.၂ Servo motor ကိုထိန်းချုပ်ခြင်း | ၁၀၆ |
| ၅ Serial Bus အနီးဆက်သွယ်ရေးများ | ၁၁၁ |
| ၅.၁ I2C | ၁၁၁ |
| ၅.၁.၁ C++ ဖြင့်သုံးခြင်း | ၁၁၅ |
| ၅.၁.၂ Gyroscope ကိုအသုံးပြုခြင်း | ၁၁၇ |
| ၅.၂ SPI | ၁၂၄ |
| ၅.၂.၁ Gyroscope ကိုအသုံးပြုခြင်း | ၁၂၇ |
| ၆ စက်အမြင် | ၁၃၅ |
| ၆.၁ နောက်ခံအကြောင်းအရာ | ၁၃၅ |
| ၆.၂ ဖွဲ့စည်းပုံ | ၁၃၆ |
| ၆.၃ တပ်ဆင်ခြင်း | ၁၃၇ |
| ၆.၃.၁ ရှိထားရန်လိုအပ်သည့် packages များ | ၁၃၇ |
| ၆.၃.၂ Apt ဖြင့်တပ်ဆင်ခြင်း | ၁၃၈ |
| ၆.၃.၃ Source မှ build လုပ်ခြင်း | ၁၃၉ |
| ၆.၃.၄ GCC ၊ CMake တို့ဖြင့် အသုံးပြုခြင်း | ၁၄၀ |
| ၆.၄ ကင်မရာကိုသုံးခြင်း | ၁၄၃ |
| ၆.၅ Real-time Face Detection | ၁၄၇ |
| ၆.၆ Smart Surveillance ကင်မရာပြုလုပ်ခြင်း | ၁၅၁ |
| ၆.၇ Crontab ကိုသုံးခြင်း | ၁၅၇ |
| ၆.၇.၁ Crontab တွင်လုပ်ငန်းသတ်မှတ်ခြင်း | ၁၅၈ |
| ၇ GUI application များရေးသားခြင်း | ၁၆၁ |
| ၇.၁ wxWidgets ကိုတပ်ဆင်ခြင်း | ၁၆၂ |
| ၇.၂ wxWidgets ကို source မှ build လုပ်ခြင်း | ၁၆၄ |

| | | |
|----------|--|------------|
| ၇.၃ | OpenCV နှင့် wxWidgets ကိုထွေသုံးခြင်း | ၁၆၅ |
| ၇.၃.၁ | Code::Blocks | ၁၆၉ |
| ၈ | Serial Port ကိုသုံးခြင်း | ၁၇၇ |
| ၈.၁ | UART | ၁၇၇ |
| ၈.၁.၁ | Start Bit | ၁၇၇ |
| ၈.၁.၂ | Baud Rate | ၁၇၈ |
| ၈.၁.၃ | Data Bits | ၁၇၈ |
| ၈.၁.၄ | Parity Bit | ၁၇၈ |
| ၈.၁.၅ | Stop Bit | ၁၇၈ |
| ၈.၂ | RS-232 | ၁၇၉ |
| ၈.၂.၁ | Handshaking | ၁၈၀ |
| ၈.၃ | Hardware များ | ၁၈၂ |
| ၈.၃.၁ | Windows တွင်အသုံးပြုခြင်း | ၁၈၄ |
| ၈.၃.၂ | Linux တွင်အသုံးပြုခြင်း | ၁၈၆ |
| ၈.၄ | BBB ၏ serial port ကိုသုံးခြင်း | ၁၈၈ |
| ၈.၄.၁ | Linux Console | ၁၉၉ |
| ၈.၄.၂ | Serial Interface | ၁၉၂ |
| ၈.၅ | Programming Serial Port | ၁၉၄ |
| ၈.၅.၁ | Linux နှင့် Windows အတွက် Serial Class | ၁၉၈ |
| ၈.၅.၂ | Console | ၁၉၉ |
| ၈.၅.၃ | GUI | ၂၀၁ |
| ၉ | Internet of Things | ၂၀၇ |
| ၉.၁ | UDP | ၂၀၇ |
| ၉.၂ | TCP | ၂၂၈ |
| ၉.၂.၁ | TCP Server | ၂၂၈ |
| ၉.၂.၂ | TCP Client | ၂၄၀ |
| ၉.၃ | FTP Server တပ်ဆင်ခြင်း | ၂၅၂ |
| ၉.၄ | Web Server တပ်ဆင်ခြင်း | ၂၅၅ |

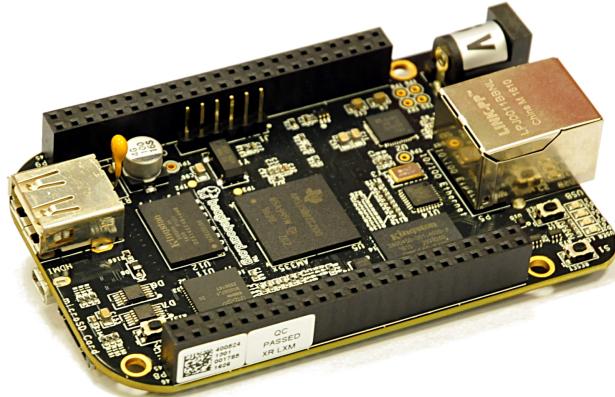
| | |
|---|-----|
| ၉.၄.၁ PHP ကို အသုံးပြုခြင်း | J၅၆ |
| ၉.၄.၂ Bone101 Server ကို Apache ဖြင့် အစားထိုးခြင်း | J၅၇ |
| ၉.၅ Google Charts | J၅၈ |
| ၉.၅.၁ Chart လိုင်ဘရီယည့်ခြင်း | J၆၁ |
| ၉.၅.၂ ဒေတာများပြင်ဆင်ခြင်း | J၆၂ |
| ၉.၅.၃ Chart ကို ဆွဲခြင်း | J၆၂ |
| ၉.၆ D3.js | J၆၃ |
| ၉.၇ Email ပြုခြင်း | J၆၇ |
| ၉.၇.၁ ဖိုင်ကိုပို့ခြင်း | J၆၈ |
| ၉.၇.၂ C++ ဖြင့်ပို့ခြင်း | J၆၉ |
| ၉.၈ Watchdog | J၆၉ |
| ၁၀ အချိန်တိကျမှုကိုထိန်းသိမ်းခြင်း | J၇၇ |
| ၁၀.၁ Real Time Clock အသုံးပြုခြင်း | J၇၇ |
| ၁၀.၁.၁ ဝါယာဆက်သွယ်မှု | J၇၈ |
| ၁၀.၁.၂ I2C ဆက်သွယ်မှု | J၇၉ |
| ၁၀.၁.၃ RTC ကို setup လုပ်ခြင်း | J၈၀ |
| ၁၀.၁.၄ Service ဖန်တီးခြင်း | J၈၂ |
| ၁၀.၂ NTP Server | J၈၃ |
| ၁၀.၂.၁ Basic Time Commands | J၈၄ |
| ၁၀.၂.၂ ntpd ပြောင်းသုံးခြင်း | J၈၆ |
| ၁၀.၂.၃ ntpd ကို ပုံစံသွင်းခြင်း | J၈၇ |
| ၁၀.၃ NTP Client | J၈၉ |
| ၁၀.၃.၁ Linux | J၉၀ |
| ၁၀.၃.၂ Windows | J၉၂ |
| A Kernel Overlays | J၉၅ |
| ၁.၁ UART | J၉၆ |
| ၁.၂ Disabling HDMI Cape | J၉၆ |
| ၁.၃ Analog Inputs | J၉၇ |

| | |
|---|------------|
| ၁.၄ Analog Outputs | ၂၉၈ |
| ၁.၅ I2C | ၃၀၁ |
| ၁.၆ SPI | ၃၀၁ |
| B Code Listing | ၃၀၃ |
| J.၁ Minimal wxWidgets sample | ၃၀၃ |
| J.၂ Alpha channel ပါရီသည့် ပုံရိပ် များကို အသုံးပြုသည့် နမူနာ minimal.cpp | ၃၀၈ |
| J.၃ Serial.h | ၃၀၀ |
| C စကားလုံးဖွင့်ဆိုချက်များ | ၃၂၉ |

အခန်း ၁

မိတ်ဆက်

BeagleBoard က Texas Instruments နဲ့ Digi-Key စ Newark element14 တို့ ပေါင်းပြီး ထုတ်လုပ်ထားတဲ့ ကွန်ပျူးတာ ဘုတ် (open source hardware single board computer) ဖြစ်ပါတယ် (ပုံ ၁.၁)။



ပုံ ၁.၁: ခရက်ဒစ် ကုဒ် အရွယ် အစား သာ ရှိပြီး ပါဝါ အစား နည်း ပါး သော BeagleBone Black ကွန်ပျူးတာဘုတ်။

သင်္ကားရေး အတွက် အထောက် အကူ အနေ နဲ့ရော၊ ဝါသနာ ရှင်တွေ၊ developer တွေ အလွယ်တကူ အသုံးပြုဖို့ အတွက် ရော ရည်ရွယ် ဒီဇိုင်း လုပ်ပြီး ဈေး သက်သက် သာသာ ထုတ်လုပ်ထားတာ ဖြစ်ပါ တယ်။

BeagleBoard.org ကထုတ်တဲ့ ဘုတ် အမျိုးမျိုး ရှိပြီး အဲဒီ ထဲမှာ BeagleBone Black က လူသိများ ကျော်ကြား ပါတယ်။ BeagleBoard တွေရဲ့ ဒီဇိုင်း ကို ဖွင့်ပေး ထားပြီး လွတ်လပ် စွာ ယူငင်

အသုံးပြု နှင့်တဲ့ Creative Commons Share-Alike လိုင်စင် ပေးထား ပါတယ်။ ဥပမာ BeagleBone Black Rev C ရဲ့ ပတ်လမ်း သက်ကဲ ပုံ (schematic) ၊ ပစ္စည်း စာရင်း (BoM) စတဲ့ အသေးစိတ် တွေကို <http://elinux.org/Beagleboard> မှာ တွေ့နိုင် ပါတယ် [Eli14; Col14]။

BeagleBoard.org ရဲ့ဘူတ် အသစ် တစ်ခု ဖြစ်တဲ့ PocketBeagle က တော်တော် လေး သေးငယ်ပြီး USB သေ့တဲ့ လိုတောင် ခေါ်ကြ ပါတယ်။ လေ့လာ သူတွေ၊ developer တွေ အတွက် သဘောကျ စရာ သေးငယ်၊ ဈေးသက်သာ တဲ့ Linux ကွန်ပျူးတာ လေး တစ်ခု လို့ ဆိုနိုင် ပါတယ်။

BeagleBone Blue ကတော့ robotic ပရောဂျက် တွေ အတွက် တော်တော် ကောင်းမွန် ပြည့်စုံ တဲ့ open source Linux ကွန်ပျူးတာ တစ်ခု လို့ ပြောနိုင် ပါတယ်။ ဘက်ထရီ တပ်ဖို့ charger input ပါတယ်။ Server မော်တာ၊ DC မော်တာ တွေ အတွက် hardware control တွေ ပါတယ်။ Wireless နဲ့ Bluetooth တွေ ပါတယ်။ GPIO နဲ့ Sensor တွေ ပါ ပါတဲ့ အတွက် drone စတဲ့ ပရောဂျက် တွေ လုပ်မယ် ဆိုရင် တောင် နောက်ထပ် hardware တွေတပ်ဆင်ရ တဲ့ ဒုက္ခ၊ နေရာ အရွယ် မလောက် တဲ့ ပြဿနာ တွေ တော်တော် သက်သာ သွားနိုင် ပါတယ်။

ဒါ စာအုပ် ထဲက အကြောင်း အရာ၊ နမူနာ တွေက ခေတ်စား၊ ဈေးသက်သာပြီး တဗြား electronic components လေးတွေ နဲ့ ချိတ်ဆက်ရ အဆင်ပြေ တဲ့ header တွေပါတဲ့ BeagleBone Black ကို အဓိက အခြေခံ ဆွေးနွေး ထား ပါတယ်။ တချို့ robotics နဲ့ ဆိုင်တဲ့ နမူနာ တွေကို တော့ BeagleBone Blue ကို သုံးထား ပါတယ်။ သဘော တရား အတူတူ ဖြစ်တဲ့ အတွက် PocketBeagle အတွက် လည်း အကျိုးဝင်ပြီး အသုံးပြု နိုင် မှာပါ။ သူတို့ရဲ့ firmware တွေက Debian Linux ကို သုံးထားတာ ဖြစ်တဲ့ အတွက် Debian ပေါ်မှာ အခြေခံ ထားတဲ့ Raspbian, Ubuntu တွေပေါ်မှာ လည်း တော်တော် များများ ဘာမှ ပြောင်းစရာ့ မလို ဒါမှမဟုတ် တဗြားတလေ လောက်ပဲ အနည်းငယ် ပြောင်းပြီး ပြန် အသုံးပြု နိုင်ပါတယ်။ BeagleBoard တွေ အတွက် နောက်ဆုံးထွက် firmware images တွေကို <https://beagleboard.org/latest-images> မှာ တွေ့နိုင် ပါတယ်။

၁.၁ နှင့်ယူဉ်မှုများ

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

Arduino (<https://www.arduino.cc/>) က microcontroller ကို အခြေခံ တာ ဖြစ်ပြီး electronic ဆားကစ် ပစ္စည်း တွေကို လွယ်လွယ် ကူကူ တိုက်ရိုက် ထိန်းကျောင်း ဖို့ သင့်တော် ပါတယ်။ တွက်ချက် ဖို့ power အများကြီး မလို၊ operating system တွေ မပါပဲ ပါဝါ ဖွင့်လိုက် တာနဲ့ ချက်ချင်း တိုက်ရိုက်

အသုံး ပြန်လည်ပါတယ်။

BeagleBone ကတော့ တွက်ချက်မှာ အများကြီး ပိုအားကောင်း တဲ့ microprocessor ကို သုံးပြီး ပိုမို ရှုပ်ထွေး အဆင့် မြင့်တဲ့ image processing လိုမျိုး တွက် သုံးတဲ့ embedded ပရောဂျက် တွေ အတွက် သင့်တော် ပါတယ် [Dah15]။ Operating system အမျိုးမျိုး တပ်ဆင် နှင့်ပြီး အခါ BeagleBone Black မှာတော့ Debian Linux ပါ ပါတယ်။

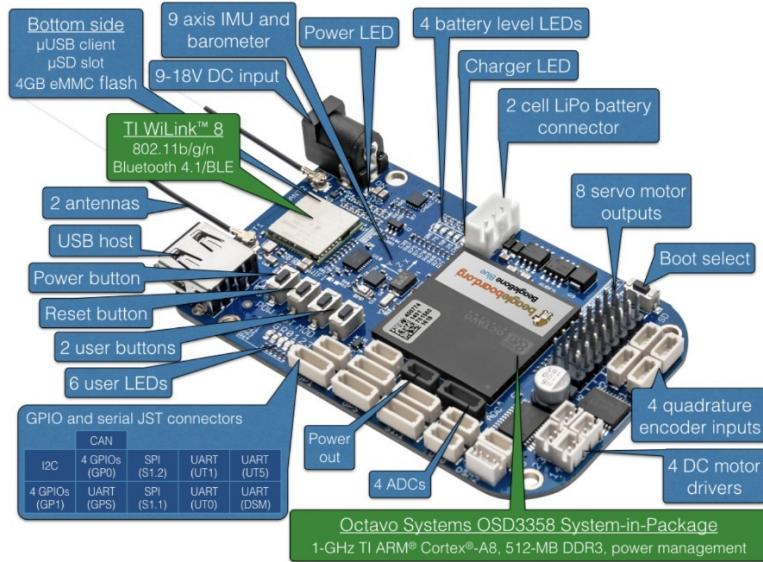
Single Board Computer တွက် လူ အများ ကြေား အလွယ် တကူ ဈေးပေါ်ပေါ် နဲ့ သုံးဖို့ ပထမ ဆုံး အောင်မြင်စွာ ဖန်တီး ထုတ်လုပ်နိုင် ခဲ့တဲ့ Raspberry Pi (<https://www.raspberrypi.org/>) က BeagleBone ထက် အရင် ပေါ် တာပါ။ ဈေး အနေနဲ့ လည်း BeagleBone က နဲ့ ပိုကြီး ပါတယ်။

စတင် အသုံးပြု ရ လွယ်ကူ တာခြင်း ယုံးရင် BeagleBone က ပို အဆင်ပြေ ပါတယ် [Leo14]။ Raspberry Pi က USB ကြိုး တခါစဲ ပါမလာ တဲ့ အပြင် သူကို စတင် လုပ်ဆောင်စွဲ SD Card လိုပြီး Operating System ကို အဲဒီထဲမှာ setup လုပ်ဖို့ လို ပါသေးတယ်။ BeagleBone ကတော့ eMMC ပါပြီး၊ ပါလာတဲ့ USB ကြိုးကို ကွန်ပျူးတာနဲ့ ဆက်လိုက် တာနဲ့ စပြီး သုံးလို့ ရပါပြီ။

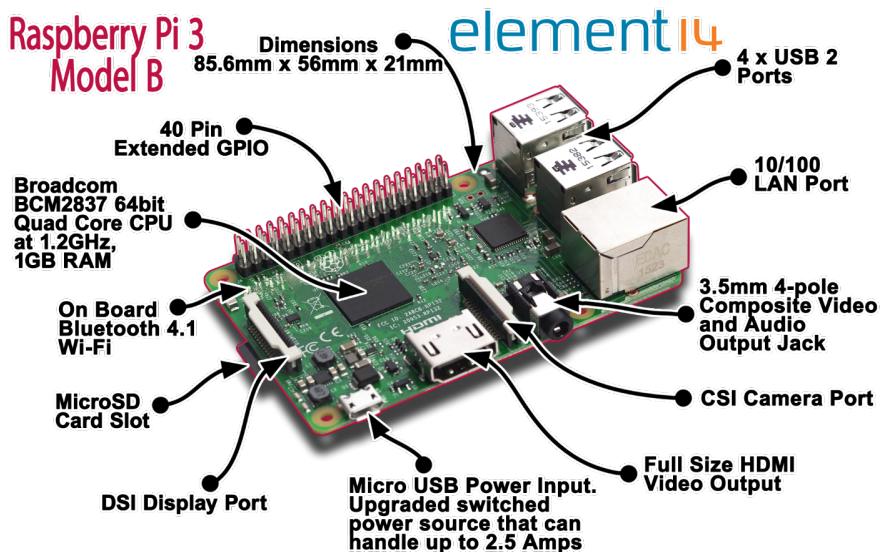
တွေား အီလက် ထရောနစ် ပစ္စည်း တွေနဲ့ ဆက်သွယ် အသုံးပြု ဖို့ interface ခြင်း ယုံးရင်တော့ BeagleBone က သာတာကို တွေ့ရ ပါတယ် (ပုံ ၁.၂ နှင့် ယေား ၁.၁)။

Multimedia နဲ့ ပတ်သက်တဲ့ လုပ်ဆောင်မှု တွေမှာတော့ Raspberry Pi က ပိုကောင်း ပါတယ်။ Raspberry Pi မှာ HDMI output | camera port | ၃.၅ မမ audio jack စော တွေ ပါပြီး BeagleBone မှာ မပါ ပါဘူး။ Graphics processing တွေမှာလည်း Raspberry Pi က ပိုပြီး စွမ်းဆောင်ရည် မြင့်မား ပါတယ်။

Hardware ကိုပါ ကိုယ်တိုင် ထုတ်မယ် ဆိုရင်တော့ BeagleBone က ပိုပြီး လွတ်လပ် လွယ်ကူ ပါတယ်။ Raspberry Pi က proprietary processor platform ကို အခြေခံ ထားတာမို့ processor ရဲ့ datasheet အပြည့်အစုံ ရဖို့ ခက်ခဲ နိုင်ပါတယ်။ အနဲ့ဆုံး Broadcom နဲ့ non-disclosure agreement ကို sign ထိုးဖို့ လိုမယ်။ Broadcom ကို business plan ချုပ် ရမယ်။ ပြီးရင် processor တွက် အသုတ်လိုက် ထောင်ချီ ဝယ်မှ ရမယ်။ BeagleBone မှာတော့ processor အတွက် datasheet အပြည့်အစုံ၊ user guide တွေက Texas Instruments ရဲ့ product page [Tex16] မှာ ရနိုင် ပါတယ်။ ဒုံးအပြင် Raspberry Pi foundation က RS တို့၊ Farnell တို့ နဲ့ exclusive manufacturing agreement ထားထားပြီး board ဒီဇိုင်း က အလွယ် တကူ ရဖို့ မလွယ် ပါဘူး။ BeagleBone ဆိုရင်တော့ layout, schematic, BoM, reference စိတ် ကိုယ်တိုင် board တစ်ခု လုပ်ဖို့ လိုအပ် သမျှ စာရွက် စာတမ်း အပြည့် အစုံ က BeagleBone wiki page မှာ ပေးထား တာကို တွေ့နိုင် ပါတယ် [Eli14]။



(a) BeagleBone Blue Technical Specifications (Image from https://github.com/beagleboard/beaglebone-blue/blob/master/docs/BeagleBone_Blue_ShortSpec.pdf)



(b) Raspberry Pi 3 Model B Technical Specifications (Image from <https://www.element14.com/community/docs/DOC-80899/l/raspberry-pi-3-model-b-technical-specifications>)

ဥ ၁၂: BeagleBone Blue နဲ့ Raspberry Pi 3 Model B ဗိုင်းယဉ်များ

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

| 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 | SD Card Slot | 4 GB on-board eMMC & 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 |

အင်တာနက် နဲ့ ချိတ်ဆက် အသုံးပြု ရတဲ့ IoT ပရောဂျက် တွေ အတွက် တော့ နှစ်ခုလုံး က အဆင် ပြောငြှာ တာကို တွေ့ရ ပါတယ်။

၁.၂ စတင်တပ်ဆင်အသုံးပြုခြင်း

BeagleBone ကောင်းမွန် တဲ့ အချက် တစ်ခု က ပစ္စည်း ရတာ နဲ့ ဗုံးဖွင့် ပြီး ချက်ချင်း အလွယ် တကူ တန်းသုံး လို့ ရတဲ့ အချက် ပါ။ ဗုံး ထဲမှာ BeagleBone နဲ့ အတူ USB ကြိုး၊ Quick start guide တို့ တခါတည်း ပါ ပါတယ်။

BeagleBone ကို စတင် အသုံးပြုလို ရတဲ့ နည်း နှစ်နည်း ရှိ ပါတယ်။ ပထမ နည်းက ဘာမှ မလိုပဲ USB ကြိုး နဲ့ ဂွန်ပျူးတာ ကို ချိတ်ပြီး အသုံးပြု တဲ့ နည်းပါ။ ဒုတိယ နည်းက တော့ BeagleBone ကို မော်နိတာ၊ ကိုးဘူတာ၊ မောက်စ် တို့ ချိတ်၊ ပါဝါပေးပြီး သူ ဟာသူ တစ်ခု တည်း သီးသန့် သုံးတာပါ။ ပါဝါ adapter မရှိရင် USB ကြိုး ကနေ ပါဝါ ပေးလို ရပါတယ်။ မော်နိတာ အတွက် ကတော့ micro HDMI

ကြိုး ဒါမှမဟုတ် adapter လိုပါတယ်။

ဒီနေရာ မှာတော့ ပိုမို လွယ်ကူ အဆင်ပြေ တဲ့ ဘာမှ ထွေထွေ ထူးထူး မလိုပဲ box တဲ့မှာ တစ်ခါကဲ ပါလာတဲ့ USB ကြိုး ကိုသုံးပြီး ကွန်ပျူးတာ နဲ့ချိတ်ဆက်တဲ့ နည်းကို သုံးပါမယ်။

၁.၂.၁ ကွန်ပျူးတာဖြင့်ချိတ်ဆက်ခြင်း

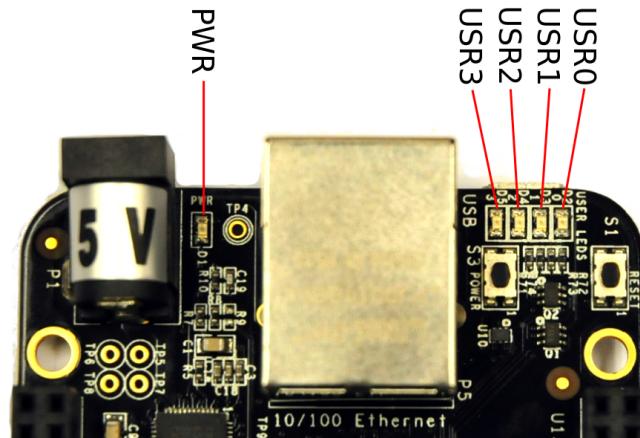
BeagleBone ကို USB ကြိုး သုံးပြီး ကွန်ပျူးတာ နဲ့ ချိတ် လိုက် တာနဲ့ USB ပါဝါ သုံးပြီး အလုပ် လုပ်ပါ လိမ့်မယ်။ ဘုတ် ပေါ်က User LED လို ခေါ်တဲ့ မီးလုံး အပြာ တွေ မိုတ်တူတ်၊ မိုတ်တူတ် ဖြစ်ပြီး သူ့ရဲ့ Linux စနစ်ကို boot လုပ်တာ ဝေ စကြွန် လောက်ပဲ ဖြော ပါတယ်။ သူ့ရဲ့ LED မီးလုံး တွေက အောက်က အတိုင်း ဖြစ်ပါတယ် (ပုံ ၁.၃)။

USR0: အလုပ် လုပ် နေတဲ့ အကြောင်း နှလုံး ခုန် သလို မိုတ်တူတ် မိုတ်တူတ် ပုံမှန် ပြန် ပါမယ်။

USR1: microSD card ကို ဖတ်တဲ့ အခါ လင်းပါမယ်။

USR2: CPU ရဲ့လုပ်ဆောင်မှု ကို ဖော်ပြတာပါ။

USR3: eMMC ကို ဖတ်တဲ့ အခါ လင်းပါမယ်။



ပုံ ၁.၃: BBB ၏ user LED များ။

အဲဒီနောက် ကွန်ပျူးတာ မှာ BeagleBone Getting Started ဆိုတဲ့ နာမည် နဲ့ mass storage drive တစ်ခု ပေါ်လာ ပါလိမ့်မယ်။ အဲဒီ drive ကို ဖွင့်ပြီး START.HTM ဆိုတဲ့ ဖိုင်ကို Firefox စား ပါတယ်။

နောက်ပြီး အဲဒီ စာမျက်နှာ အဆင့် J မှာ ပြထားတဲ့ ညွှန်ကြား ချက်တွေ အတိုင်း USB ပေါ်က နေ network ချိတ်ဆက် ပြီး သုံးဖို့ ကိုယ့်စက် နဲ့ ကိုက်ညီတဲ့ driver ပေါ်ကို နှိပ်ပြီး install လုပ်လိုက် ပါမယ် (ပုံ ၁.၄)။ <http://beagleboard.org/getting-started#step2> ကနေ download လုပ် install လုပ်ရင်လည်း ရပြီး၊ Windows 10 အတွက် အဲဒီက driver ကပဲ အဆင်ပြော တာကို တွေ့ရ ပါတယ်။

| Operating System | USB Drivers | Comments |
|------------------|-------------------------------|--|
| Windows (64-bit) | 64-bit installer | If in doubt, try the 64-bit installer first. <ul style="list-style-type: none"> Note #1: Windows Driver Certification warning may "Ignore", "Install" or "Run" Note #2: To check if you're running 32 or 64-bit Win http://support.microsoft.com/kb/827218. Note #3: On systems without the latest service release (0xc000007b). In that case, please install the follow http://www.microsoft.com/en-us/download/confirm Note #4: You may need to reboot Windows. |
| Windows (32-bit) | 32-bit installer | |
| Mac OS X | Network Serial | Install both sets of drivers. |
| Linux | mkudevrule.sh | Driver installation isn't required, but you might helpful . |

ပုံ ၁.၄: လိုအပ်သော driver များ တပ်ဆင်ခြင်း။

Linux စက် ဆိုရင် တော့ driver တွေ တပ်ဆင် စရာ မလို ပါဘူး။ Linux အတွက် ပေးထား တဲ့ `mkudevrule.sh` ကို [online](#) (<http://beagleboard.org/static/Drivers/Linux/FTDI/mkudevrule.sh>) ကနေ download လုပ်ပြီး terminal မှာ အောက်က စာရင်း ၁.၁ အတိုင်း ရိုက်ထည့်ပြီး BeagleBone ကို USB နဲ့ ပြန် ဆက်နိုင် ပါတယ် [Shi13]။

```

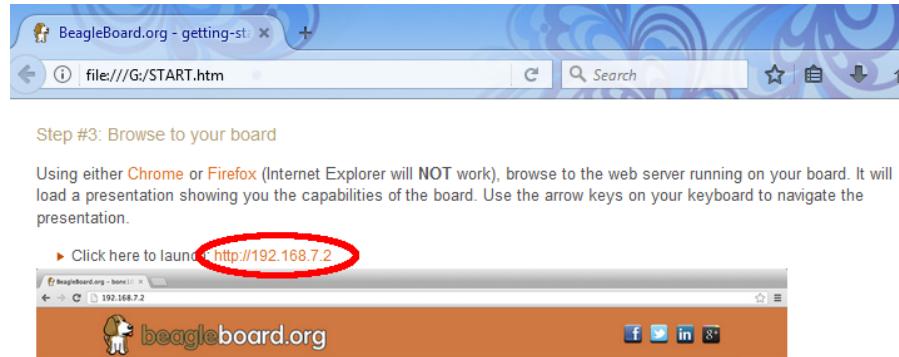
1 $ chmod 755 mkudevrule.sh
2 $ sudo ./mkudevrule.sh

```

စာရင်း ၁.၁: BeagleBone ကို Linux တွင် တပ်ဆင် အသုံးပြုခြင်း။

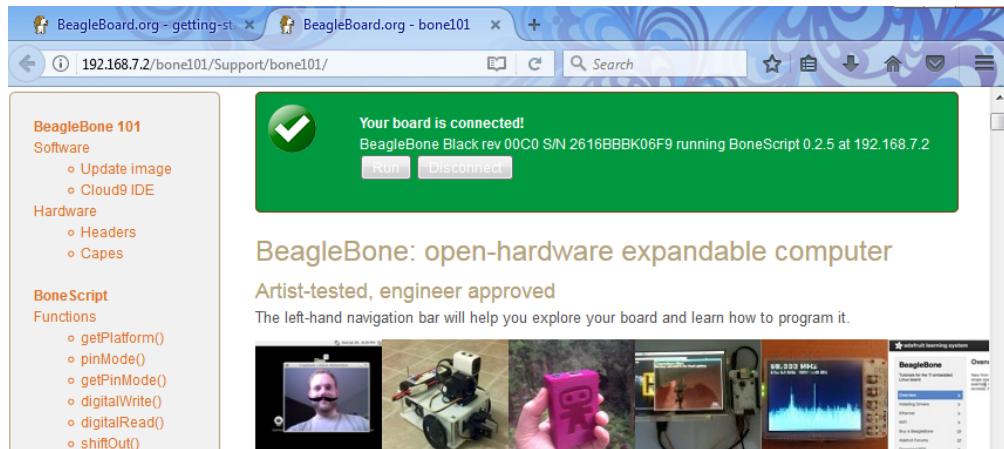
၁.J.J Cloud9

ပုံ ၁.၅ က အဆင့် ၃ မှာ ပြထား တဲ့ [ဆိုတဲ့ link ကို ဖွင့်လိုက်ရင် BBB ပေါ်မှာ run နေတဲ့ web server ကို သွားပြီး ဖွင့်ပေးပါ လိမ့်မယ်။](http://192.168.7.2/)



ပုံ ၁.၅: BBB ဘူတ်၏ web server ကိုလုမ်းဆက်သွယ်ပြင်း။

အဲဒီ အခါ BeagleBone 101 ဆိုတဲ့ စာမျက်နှာ ပွင့်လာပြီး BBB ရဲ့ software, hardware နဲ့
ပတ်သက်တဲ့ အချက် အလက် တွေ ကို တွေ့နှိပ်ပါတယ် (ပုံ ၁.၆)။ ဘယ်ဘက် ကော်လံ ရဲ့ အောက်ဖက်
နားမှာ စိတ်ဝင် စား စရာ နမူနာ (Demos) တွေ လည်း တွေ့နှိပ်ပါတယ်။



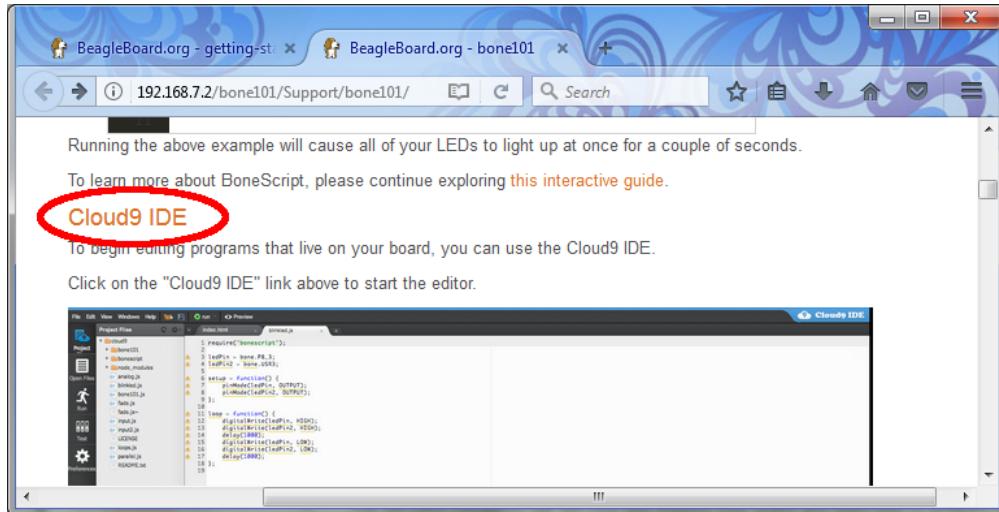
ပုံ ၁.၆: BBB ဘူတ်၏ web စာမျက်နှာ။

အဲဒီ စာမျက်နှာ ကို ဆက်ဖွင့် ကြည့်ပြီး ပုံ ၁.၇ မှာ ပြထား တဲ့ အတိုင်း Cloud9 IDE - <http://192.168.7.2:3000/> ဆိုတဲ့ link ကို နှိပ်လိုက် ရင် BBB အတွက် ပရိုဂရမ် တွေကို တည်းဖွတ်

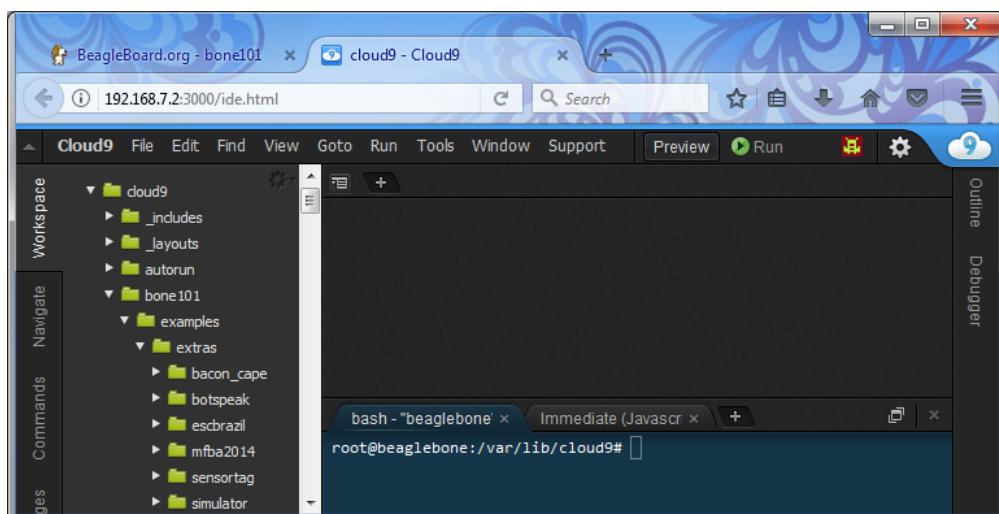
၁.၂. စတင်တပ်ဆင်အသုံးပြုခြင်း

၃

ရေးသား ဖို့ Cloud9 IDE ပေါ်လာပါမယ် (ပုံ ၁.၈)။



ပုံ ၁.၈: Cloud9 IDE ကို ဖွင့်ခြင်း။



ပုံ ၁.၉: <http://192.168.7.2:3000/> တွင် တွေ့မြင်ရသော Cloud9 IDE။

BBB အတွက် ပရိုဂရမ် တွေ့ကို Cloud9 ပေါ်မှာ ရေးလို ရပြီး၊ ဘယ်ဘက် နားမှာ ဖိုင်တွေ့ကို လိုသလို စီမံ လိုရတဲ့ file explorer တစ်ခု ကို တွေ့နိုင် ပါတယ်။ အောက်ဘက် နားမှာ တော့ bash command တွေ သုံးလို ရတဲ့ terminal တစ်ခု တွေ့နိုင် ပါတယ်။ နောက် အခန်း မှာ BoneScript လို ခေါ်တဲ့ Javascript လိုင်ဘရီ ကို သုံးပြီး BBB ကို အသုံးပြု တဲ့ အကြောင်း ဆက်ဆွေးနွေး ပါမယ်။

အကိုးအကားများ

- [Col14] Gerald Coley. System Reference Manual Rev C.1. 2014. url: https://github.com/CircuitCo/BeagleBone-Black/blob/master/BBB_SRM.pdf?raw=true.
- [Dah15] Oyvind Nydal Dahl. Raspberry Pi or Arduino? 2015. url: <https://www.build-electronic-circuits.com/raspberry-pi-or-arduino/>.
- [Eli14] Elinux. Beagleboard BeagleBoneBlack. 2014. url: <https://elinux.org/Beagleboard:BeagleBoneBlack>.
- [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/>.
- [Shi13] Shibu. How to Install and Test Beaglebone black in Ubuntu / Debian Linux. 2013. url: <http://www.shibuvarkala.com/2013/06/how-to-install-and-test-beaglebone.html>.
- [Eli14] Elinux. Beagleboard:Main Page. 2014. url: https://elinux.org/Beagleboard:Main_Page.
- [Tex16] Texas Instruments. AM3359 Sitara Processor: ARM Cortex-A8, EtherCAT, 3D, PRU-ICSS. 2016. url: <http://www.ti.com/product/am3359>.

အခန်း J

BoneScript

BoneScript ကတေသာ embedded Linux ပေါ်မှာ တွက်ချက် ပြုလုပ်မှု တွေကို ရိုးရှင်းစာ အလွယ် တကူလေ့လာ ရေးသား နိုင်ဖို့ အတွက် ဖန်တီးထားတဲ့ Javascript လိုင်ဘရီ တစ်ခု ပါ [Bea14]။

J.၁ Cloud9 ကို သံဃားခြင်း

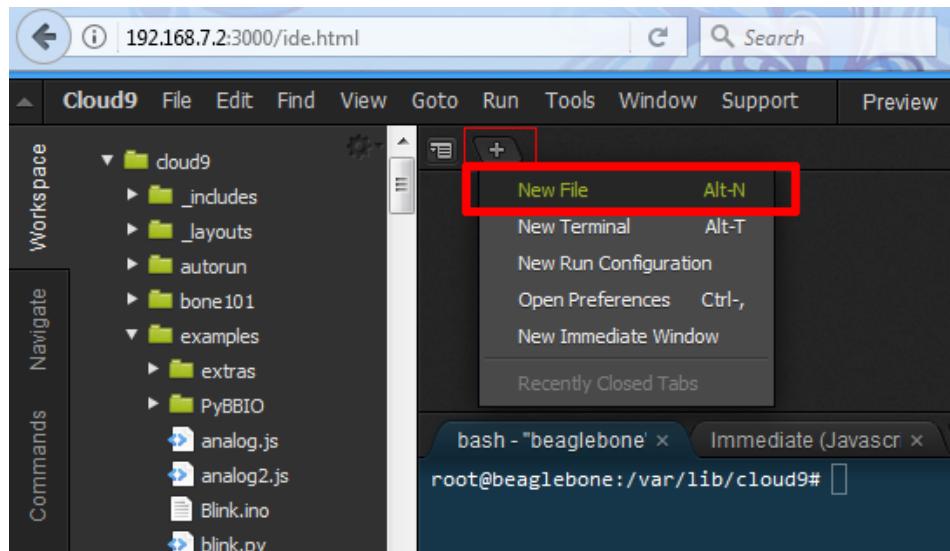
BBB ကို MiniUSB နဲ့ ချိတ်ဆက် ထားပြီး Web browser ပေါ်မှာ <http://192.168.7.2:3000/> ဆိုတဲ့ address ကို ဖွင့်လိုက်ရင် Cloud9 IDE ပေါ်လာ ပါမယ်။ ပွင့် နေတဲ့ tab တွေ အကုန် ပိတ်ပြီး တဲ့ အခါ ပဲ J.၁ မှာ ပြထား သလို File→New File ကို နှိပ်ပြီး blink3.js ဆိုတဲ့ ဖိုင် အသစ် တစ်ခု ကို ဖန်တီး လိုက် ပါမယ်။ ပြီးတဲ့ အခါ စာရင်း J.၁ မှာ ပြထား တဲ့ ကုဒ် တွေကို ရေးလိုက် ပါမယ်။

```
1 var b = require('bonescript');
2 var state = 0;//initialize LED state
3 b.pinMode("USR3", b.OUTPUT);//define USR3 as output
4 setInterval(toggle, 1000);//set timer
5 //to call toggle function every 1 second
6 //toggle USR3 led
7 function toggle() {
8     state ^= 1;//toggle the LED state
9     b.digitalWrite("USR3", state);//write the resulting state
10 }
```

စာရင်း J.၁: LED blink BoneScript ပရိုဂရမ် blink3.js ။

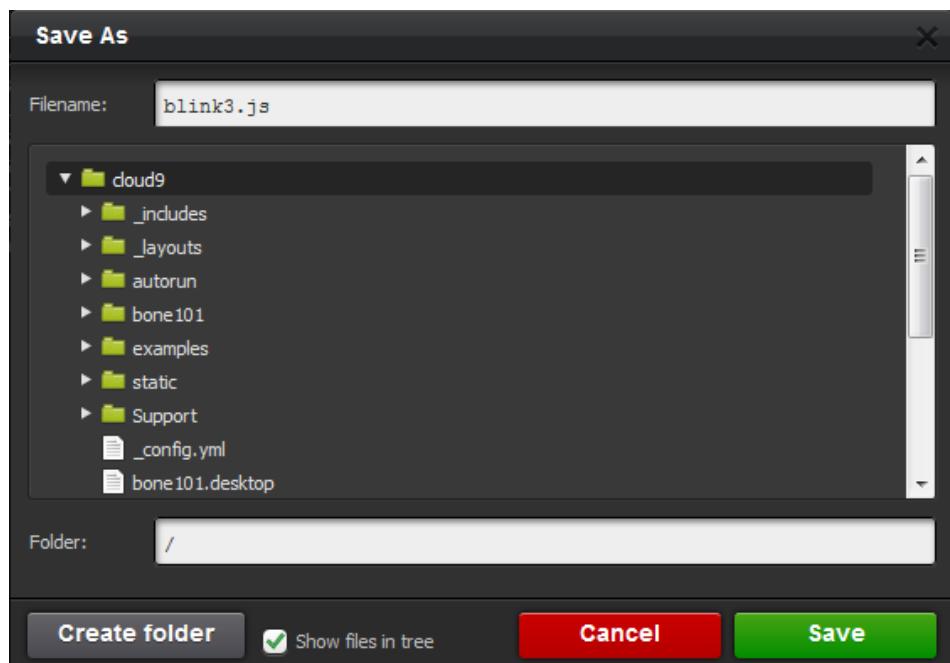
၁၂

အခါး J. BONESCRIPT



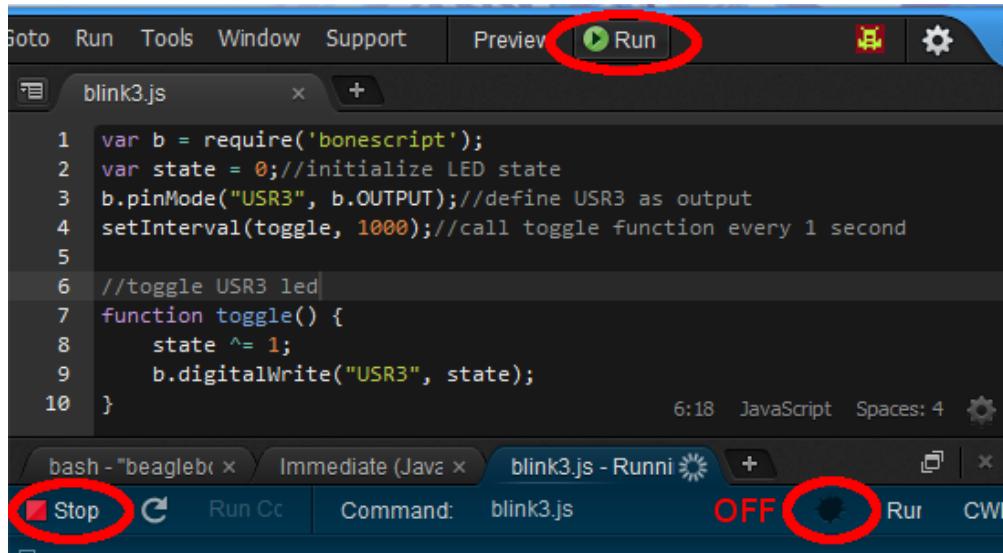
ဗုဒ္ဓဘား: Cloud9 တွင် ပရိဂရမ် အသစ်တစ်ခု ဖန်တီးခြင်း။

အဲဒီ နောက်မှာ File → Save ကို နိုပ်ပြီး .js extension နဲ့ ဖိုင် နာမည် တစ်ခု ပေးပြီး သိမ်းလိုက် ပါမယ်
(ဗုဒ္ဓဘား)



ဗုဒ္ဓဘား: BoneScript ဖိုင်ကို သိမ်းခြင်း။

ပြီးရင် အပေါ် ဘားက Run ကို နှိပ်ပြီး ရေးထား တဲ့ BoneScript ပရိုဂရမ် ကို လုပ်ဆောင် နှင့် ပါတယ်။ ပုံ J.၃ မှာ ပြထား တဲ့ အတိုင်း ညာဖက် အောက်နား က ပိုးကောင် ပုံ အိုက်ကွန် လေးကို နှိပ်ပြီး ပိတ်ထား ဖို့ လို ပါတယ်။ ပရိုဂရမ် စပြီး run တဲ့ အခါ USR3 LED မီးလုံးလေး က ၁ စက္ကန် စီ မိတ်လိုက် လင်းလိုက် ဖြစ်နေ တာကို တွေ့ရ မှာပါ။ ပြီးတဲ့ အခါ Stop ကို နှိပ်ပြီး ရင်နိုင်ပါတယ်။



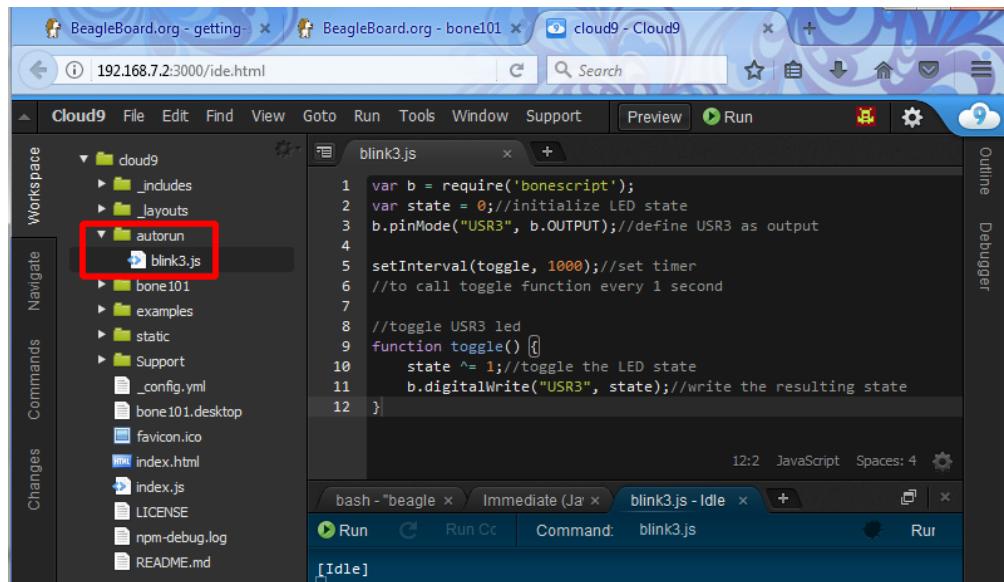
ပုံ J.၃: BoneScript ဖိုင်ကို run ခြင်း။

ပရိုဂရမ် ကို အသေးစိတ် ကြည့်လိုက်ရင် လိုင်း နံပါတ် ၃ မှာ USR3 pin ကို အထွက် (output) အနေနဲ့ သုံးမယ့် အကြောင်း pinMode နဲ့ သတ်မှတ် ပါတယ်။ နောက်တစ်ခါ လိုင်း နံပါတ် ၄ မှာ toggle ဆိုတဲ့ function ကို ၁ဝဝဝ မီလီစက္ကန် (၁ စက္ကန်) တစ်ခါ ခေါ်သုံးဖို့ setInterval နဲ့ သတ်မှတ် လိုက် ပါတယ်။ toggle ဆိုတဲ့ function ထဲမှာ LED ရဲ့ state ကို 1 နဲ့ exclusive-OR operation (သက်တဲ့) လုပ် တဲ့ အတွက် 0 နဲ့ 1 တလုညွှန်စီ ပြောင်း နေပါမယ်။ ရလာတဲ့ state ကို digitalWrite သုံးပြီး USR3 မှာ ရေးလိုက်တဲ့ အခါ LED မီးလုံး အဖွင့် အပိတ် ဖြစ်သွား ပါတယ်။

J.J Autorun

JavaScript ပရိုဂရမ် တစ်ခု ရေးပြီး တဲ့ အခါ အဲဒီ ပရိုဂရမ် ကို BBB boot-up လုပ်တိုင်း စလုပ် စေချင်ရင် ဖိုင် system ထဲက 'autorun' အခန်း /var/lib/cloud9/autorun ဆိုတဲ့ နေရာ မှာ ပုံ J.၄ လို သိမ်းပေး လိုက်ရုံ ပါပဲ။

BBB ရဲ့ အစမှာ လုပ်ဆောင်တဲ့ bonescript-autorun.service က /usr/lib/node_modules/bonescript/autorun.js ဆိုတဲ့ script ကို သုံးပြီး အလို အလျောက် စတင် ရမယ့် ပရိုဂရမ် တွေကို အဲဒီ အခန်းမှာ ရှာ ပါတယ်။ ပြီးတဲ့ အခါ node.js process တွေ အနေနဲ့ သပ်သပ်စီ လုပ်ဆောင် ပေးမှာ ဖြစ်ပါတယ်။ အဲဒီ autorun အခန်း ထဲက ဖိုင်ကို ဖျက်လိုက်ရင် အဲဖိုင်ရဲ့ process ကို အလို အလျောက် အဆုံးသတ် ပေးမှာ ဖြစ်ပါတယ်။



ပုံ J.၄: BoneScript ဖိုင် တစ်ခု ကို အလို အလျောက် စတင်ရန် autorun တွင် သိမ်းခြင်း။

J.၃ Digital output ထုတ်၍ hardware များကို ထိန်းချုပ်ခြင်း

BeagleBone Black မှာ ပြင်ပ hardware တွေကို ဆက်သွယ် ထိန်းချုပ် ဖို့ interface တွေ အများကြီး ပါ ပါတယ်။ အဲဒီ ထဲမှာ GPIO (General purpose input output) pin တွေ ကလည်း တစ်ခု အပါအဝင် ဖြစ်ပါတယ်။ နမူနာ အနေနဲ့ BBB သုံးပြီး Breadboard တစ်ခု ပေါ်မှာ ကိုယ့်ဟာ ကိုယ် ချိတ်ဆက် ထားတဲ့ LED မီးလုံး လေးကို အဖွင့် အပိတ် ထိန်းချုပ် ကြည့် ပါမယ်။

ပထမ အဆင့် အနေနဲ့ ပုံ J.၅ မှာ ပြထားတဲ့ BeagleBone 101 ဆိုတဲ့ ဝက်ဘ် စာမျက်နှာ ရဲ့ ဘယ်ဘက် ကော်လံ အောက်နားက Demos ထဲမှာ Blink external LED ဆိုတဲ့ နမူနာ ကို နိုပ်ပြီး ဖွင့်လိုက် ပါမယ်။

J.২. DIGITAL OUTPUT দ্বারা কোড তৈরি করা এবং পরিষেবা

১৯

```

var b = require('bonescript');
var led = "P8_13";
var state = 0;

b.pinMode(led, 'out');
toggleLED = function() {
    state = state ? 0 : 1;
    b.digitalWrite(led, state);
};

timer = setInterval(toggleLED, 100);

stopTimer = function() {
    clearInterval(timer);
};

setTimeout(stopTimer, 30000);

```

Bonescript: initialized
Bonescript: initialized

Build and execute instructions

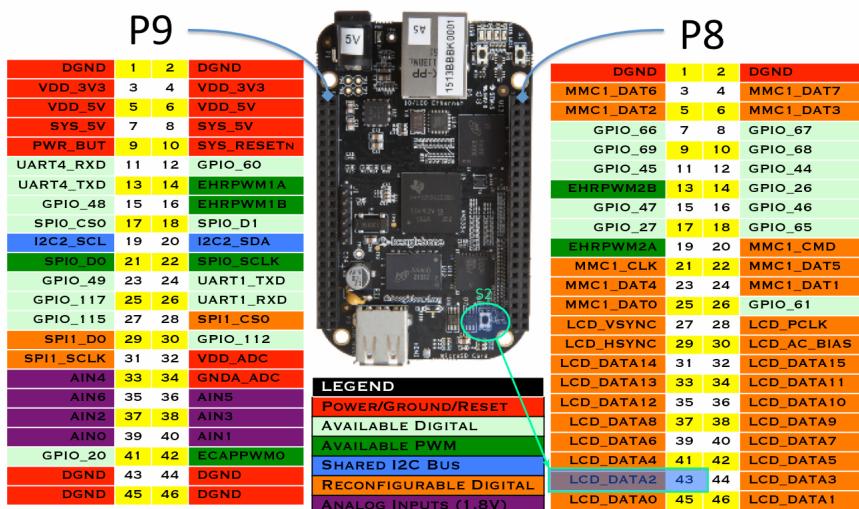
- Connect an LED and resistor as shown being sure to get the anode/cathode orientation correct.
- Run the example code and observe the blinking.
- Alter the frequency and re-run the example.

See also

Topics

- BeagleBone expansion headers
- Digital I/O

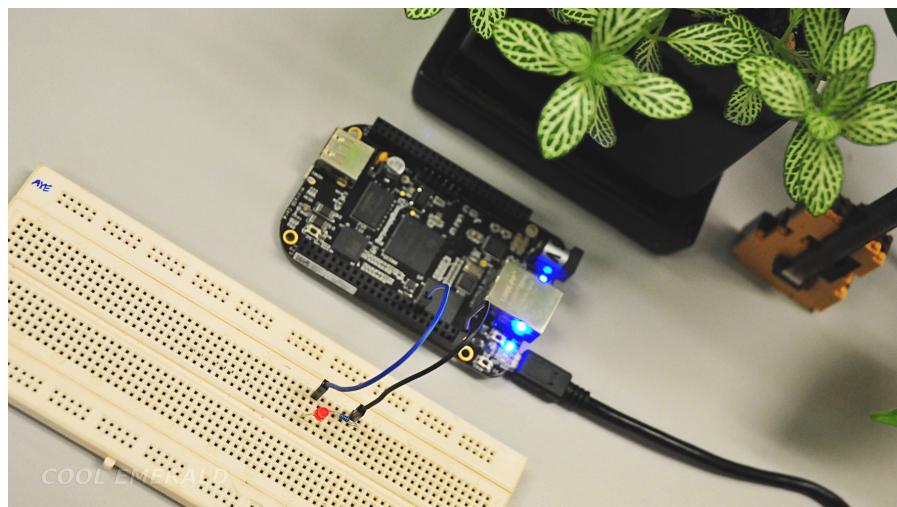
ঁ J.৩: Blink external LED কোড



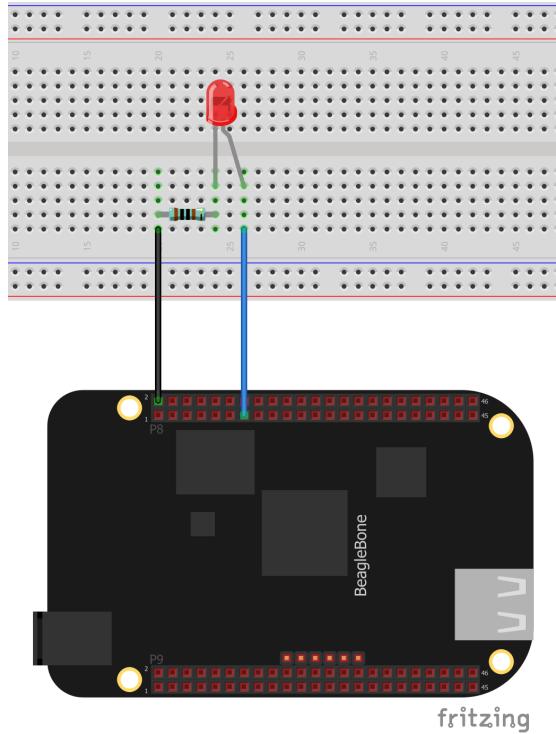
| P9 | | | P8 | | |
|----------|----|----|------------|---|---|
| DGND | 1 | 2 | DGND | 1 | 2 |
| VDD_3V3 | 3 | 4 | VDD_3V3 | | |
| VDD_5V | 5 | 6 | VDD_5V | | |
| SYS_5V | 7 | 8 | SYS_5V | | |
| PWR_BUT | 9 | 10 | SYS_RESETN | | |
| GPIO_30 | 11 | 12 | GPIO_60 | | |
| GPIO_31 | 13 | 14 | GPIO_50 | | |
| GPIO_48 | 15 | 16 | GPIO_51 | | |
| GPIO_5 | 17 | 18 | GPIO_4 | | |
| I2C2_SCL | 19 | 20 | I2C2_SDA | | |
| GPIO_3 | 21 | 22 | GPIO_2 | | |
| GPIO_49 | 23 | 24 | GPIO_15 | | |
| GPIO_117 | 25 | 26 | GPIO_14 | | |
| GPIO_115 | 27 | 28 | GPIO_113 | | |
| GPIO_111 | 29 | 30 | GPIO_112 | | |
| GPIO_110 | 31 | 32 | VDD_ADC | | |
| AIN4 | 33 | 34 | GND_ADC | | |
| AIN6 | 35 | 36 | AIN5 | | |
| AIN2 | 37 | 38 | AIN3 | | |
| AIN0 | 39 | 40 | AIN1 | | |
| GPIO_20 | 41 | 42 | GPIO_7 | | |
| DGND | 43 | 44 | DGND | | |
| DGND | 45 | 46 | DGND | | |

ံ J.ဂ: BBB ၏ GPIO pin များ။

BBB မှာ ပုံ J.၆ မှာ ပြထားသလို P8 နဲ့ P9 ဆိုတဲ့ header နှစ်ခု ပါ ပါတယ်။ အဲဒီ header တွေမှာ pin တွေ အများကြီး ပါပြီး အဲဒီ ထဲမှာ GPIO အနေ နဲ့ သုံးလို ရတဲ့ pin တွေကို ပုံ J.၇ မှာ ပြထား ပါတယ်။ ဒီ နှမူနာ မှာတော့ P8 ရဲ့ pin 13 ကို သုံးပြီး LED နဲ့ ချိတ်ဆက် မှာ ဖြစ်ပါတယ်။ ချိတ်ဆက်ပုံ ပုံ J.၉ နဲ့ ပုံ J.၈ မှာ ပြထား ပါတယ်။



ံ J.၈: BBB နှင့် External LED တစ်ခုကို breadboard သုံး၍ ချိတ်ဆက် ထားသည်။



ပုံ J.၉: External LED ချိတ်ဆက်ပုံ။

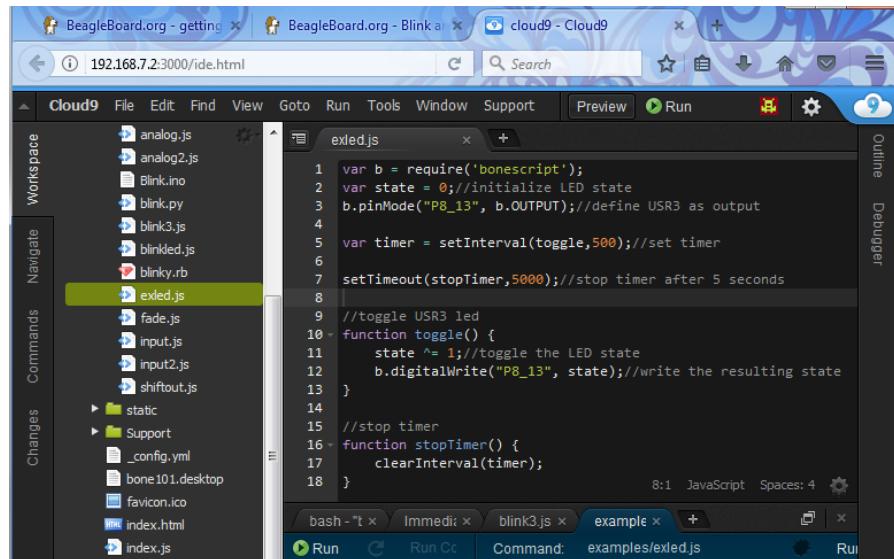
ပြီးတဲ့ အခါ Cloud9 IDE ကို ပြန်သွားပြီး exled.js ဆိုတဲ့ ဖိုင် အသစ် တစ်ခု ဖန်တီး ပါမယ်(ပုံ J.10)။ စာရင်း J.J အတိုင်း ကုဒ်တွေ ရေးထည့် ပြီး run လိုက်တဲ့ အခါ LED မီးလုံး ငါးခါ မြှတ်တုတ် မြှတ်တုတ် ဖြစ်သွား တာကို တွေ့ရ ပါမယ်။ ဒီ နမူနာ မှာ setInterval ကို သုံးပြီး စတုန်း ဝက် တခါ pin ကို အဖွင့် အပိတ် ပြောင်းပေး တာကို တွေ့နိုင် ပါတယ်။ နောက် တစ်ခါ setTimeout ကို သုံးပြီး ၅ စတုန်း တိတိ ကြောတဲ့ အခါ timer ကို ရပ်လိုက် တာကို တွေ့ရ မှာပါ။

```
1 var b = require('bonescript');
2 var state = 0;
3 b.pinMode("P8_13", b.OUTPUT);
4 var timer = setInterval(toggle,500);
5 setTimeout(stopTimer,5000);
6 function toggle() {
7     state ^= 1;
8     b.digitalWrite("P8_13", state);
9 }
10 function stopTimer() {
```

```

11     clearInterval(timer);
12 }
```

စာရင်း J.J: Blinking external LED BoneScript ပရိုဂရမ် exled.js ။



ပုံ J.၁၀: P8 pin 13 ကို စိန်းချပ်သည့် BoneScript ပရိုဂရမ်။

J.၄ Digital တန်ဖိုးများဖတ်ခြင်း

ပြင်ပ hardware တွေရဲ့ digital တန်ဖိုး တွေကို BBB သုံးပြီး ဖတ်ဖို့ GPIO pin တွေရဲ့ pinMode ကို Input အနေနဲ့ သတ်မှတ်ပြီး သုံးနိုင် ပါတယ်။ ဖတ်ဖို့ အတွက် digitalRead ဆိုတဲ့ ဖန်ရှင် ကို သုံးနိုင် ပါတယ်။ BeagleBone 101 ဝက်ဘ်စာမျက်နှာ ကို သွားပြီး ပုံ J.၁၁ မှာ ပြထားတဲ့ ဘယ်ဘက် ကော်လံ BoneScript အောက် Functions ထဲက digitalRead ကို နှိပ်ပြီး ဖွင့်လိုက် ပါမယ်။

အဲဒီနောက် စာရင်း J.၃ မှာ ပြထား တဲ့ P8 header ရဲ့ pin နံပါတ် 19 ကို ဖတ်တဲ့ ပရိုဂရမ် digital-Read.js ကို Cloud9 မှာ ဖန်တီး လိုက် ပါမယ်။ အဲဒီ P8_19 ကို ဘာမှ မဆက်ပဲ ဖတ်လိုက် ရင် LOW ဖြစ်နေ ပါမယ်။ နောက် တစ်ခါ Vdd တန်ဖိုး 3.3V ထုတ်ပေးတဲ့ P9 header ရဲ့ pin နံပါတ် 3 နဲ့ ဝါယာ ကြိုး တစ်ခု သုံးပြီး ဆက်လိုက် ပြီးမှ ဖတ်လိုက်ရင် တော့ HIGH ဖြစ် သွားတာကို တွေ့ရ ပါလိမ့် မယ်။

J.၄. DIGITAL တန်ဖိုးများဖတ်ခြင်း

၁၉

The screenshot shows the BeagleBoard.org documentation for the `digitalRead()` function. The left sidebar lists various functions under the `BoneScript Functions` category, with `digitalRead()` highlighted by a red rectangle. The main content area provides a detailed description of the `digitalRead(pin, [callback])` function, stating it reads the status of a digital I/O pin. It includes arguments (`pin` and `callback`), a return value section with `HIGH` and `LOW` descriptions, and a `callback(x)` section. Below this is an example code snippet:

```
1 var b = require('bonescript');
2 b.pinMode('P8_19', b.INPUT);
3 b.digitalRead('P8_19', printStatus);
4 function printStatus(x) {
5   console.log('x.value = ' + x.value);
6   console.log('x.err = ' + x.err);
7 }
```

ပုံ J.၃၁: `digitalRead` ဖန်ရှင်း။

The screenshot shows a terminal window titled `digitalRead.js` containing the following JavaScript code:

```
1 var b = require('bonescript');
2 b.pinMode('P8_19', b.INPUT);
3 var val = b.digitalRead('P8_19');
4 if(val) console.log('HIGH');
5 else console.log('LOW');
```

The terminal output shows the word `HIGH`, which is circled in red.

ပုံ J.၃၂: P8_19 အဲ တန်ဖိုးကို `console` တွင် ပြပေးသော ပရိုဂရမ်။

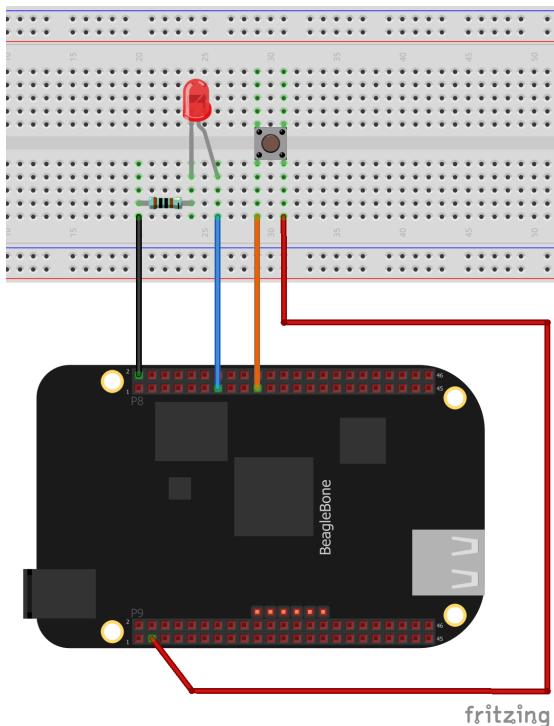
ပရိုဂရမ် ကို `run` လိုက်တဲ့ အခါ ဖတ်လို ရတဲ့ တန်ဖိုး ကို အောက်က `console` မှာ ပြနေ တာကို

ပုံ J.၁၂ အတိုင်း တွေ့နိုင် ပါတယ်။

```
1 var b = require('bonescript');
2 b.pinMode('P8_19', b.INPUT);
3 var val = b.digitalRead('P8_19');
4 if(val) console.log('HIGH');
5 else console.log('LOW');
```

စာရင်း J.၃: Digital pin ၅၏ တန်ဖိုးကို ဖတ်ပြီး console ဘွင် ပြပေးသော ပရိုဂရမ် digitalRead.js ။

Push button တစ်ခု သုံးပြီး၊ ခလုတ် နှိပ်လိုက် တဲ့ အခါ တိုင်း LED မီးလင်း ပေးတဲ့ နမူနာ pushbutton.js ကိုလည်း digitalRead ကို သုံးပြီး စာရင်း J.၄ မှာ ပြထား သလို ရေးနိုင် ပါတယ်။ အဲဒီ အတွက် ဆက်သွယ်မှု ပုံစံကို ပုံ J.၁၃ မှာ တွေ့နိုင်ပါတယ်။



ပုံ J.၁၃: Push button တစ်ခု ကို ဆက်သွယ်ပုံ နမူနာ။

```
1 var b = require('bonescript');
2 b.pinMode('P8_19', b.INPUT);
```

J.၅ ANALOG တန်ဖိုးများ ရေးဖတ်ခြင်း

၂၁

```
3 b.pinMode('P8_13', b.OUTPUT);
4 setInterval(check,100);
5
6 function check(){
7     var val = b.digitalRead('P8_19');
8     b.digitalWrite('P8_13', val);
9 }
```

စာရင်း J.၅: Push button ကို ဖတ်၍ LED ကို ထိန်းချုပ်သော ပရီဂရမ် pushbutton.js ။

J.၆ Analog တန်ဖိုးများ ရေးဖတ်ခြင်း

BBB မှာ analog input အနေနဲ့ သုံးနိုင် တဲ့ pin ၇ ခု ပါရှိ ပါတယ် (ပုံ J.၁၄)။ သူတို့ လက်ခံနိုင် တဲ့ ဗိုက် ၁ ကနေ ၁.၈ ဗို့ အထိ ပဲ ရှုတာ ကို သတိပြု သင့် ပါတယ်။ BoneScript မှာ analog input ကို ဖတ်ဖို့ အတွက် analogRead ဆိုတဲ့ ဖန်ရှင် ကို သုံးနိုင် ပါတယ်။ U-Boot Overlays ပြောင်းသုံး ထားတဲ့ firmware အသစ်တွေ မှာတော့ ရချင်မှ ရပါမယ်။

| P9 | | | P8 | | |
|----------|----|----|------------|--|--|
| DGND | 1 | 2 | DGND | | |
| VDD_3V3 | 3 | 4 | VDD_3V3 | | |
| VDD_5V | 5 | 6 | VDD_5V | | |
| SYS_5V | 7 | 8 | SYS_5V | | |
| PWR_BUT | 9 | 10 | SYS_RESETN | | |
| GPIO_30 | 11 | 12 | GPIO_60 | | |
| GPIO_31 | 13 | 14 | GPIO_50 | | |
| GPIO_48 | 15 | 16 | GPIO_51 | | |
| GPIO_5 | 17 | 18 | GPIO_4 | | |
| I2C2_SCL | 19 | 20 | I2C2_SDA | | |
| GPIO_3 | 21 | 22 | GPIO_2 | | |
| GPIO_49 | 23 | 24 | GPIO_15 | | |
| GPIO_117 | 25 | 26 | GPIO_14 | | |
| GPIO_115 | 27 | 28 | GPIO_113 | | |
| GPIO_111 | 29 | 30 | GPIO_112 | | |
| GPIO_110 | 31 | 32 | VDD_ADC | | |
| AIN4 | 33 | 34 | GNDA_ADC | | |
| AIN6 | 35 | 36 | AIN5 | | |
| AIN2 | 37 | 38 | AIN3 | | |
| AIN0 | 39 | 40 | AIN1 | | |
| GPIO_20 | 41 | 42 | GPIO_7 | | |
| DGND | 43 | 44 | DGND | | |
| DGND | 45 | 46 | DGND | | |

ပုံ J.၁၄: Analog input pin များ။

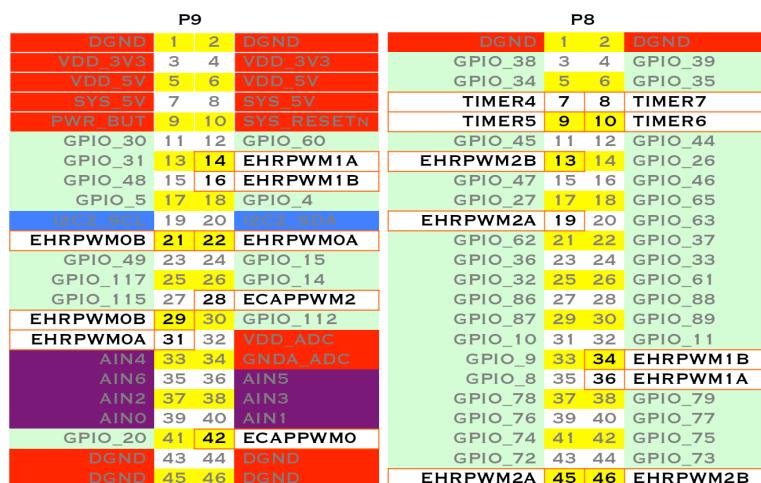
`analogRead` ရဲ့ အသုံးပြု ရတဲ့ ပုံစံ က အောက်က အတိုင်း ဖြစ် ပါတယ်။

```
analogRead(pin, [callback])
```

အဲဒီမှာ pin က ဖတ်ချင် တဲ့ analog pin ကို သတ်မှတ် ပေးဖို့ ဖြစ်ပြီး [callback] ကတော့ ထောင့်ကွင်းနဲ့ ပြထား တာမူး optional ဖြစ်ပြီး မသုံး ချင်ရင် ချိန်ထား လိုရ ပါတယ်။ ဖတ်ပြီး တဲ့ အချိန်မှာ ခေါ်ချင် တဲ့ ဖန်ရှင် ရှိရင်တော့ အဲဒီမှာ သတ်မှတ် နိုင် ပါတယ်။ `analogRead` က return ပြန် ပေးမယ့် တန်ဖိုးက 0 နဲ့ 1 အကြား တန်ဖိုး တစ်ခု ဖြစ်ပြီး၊ 0 မို့ နဲ့ ၁.၈ မို့ ကြေား က မို့ တစ်ခု ကို အချိုးကျ ကိုယ်စား ပြု ပါတယ်။ သူကို သုံးဖို့ အတွက် `pinMode` သတ်မှတ် စရာ မလို ပါဘူး။ [callback] ကို အသုံး ပြုတဲ့ နမူနာ တစ်ခု ကို စာရင်း J.၂ မှာ တွေ့နိုင် ပါတယ်။

```
1 var b = require('bonescript');
2 b.analogRead('P9_36', printStatus);
3 function printStatus(x) {
4 console.log('x.value = ' + x.value);
5 console.log('x.err = ' + x.err);
6 }
```

စာရင်း J.၂: `analogRead` နမူနာ။



ဗု J.၃: PWM နှင့် Timer pin များ။

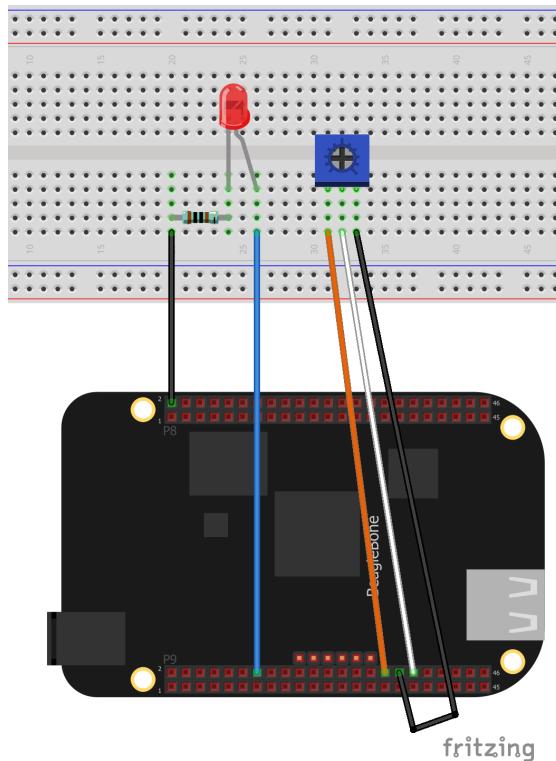
J.၅. ANALOG တန်ဖိုးများ ရေးဖတ်ခြင်း

JR

Analog တန်ဖိုး တစ်ခု ထုတ်ပေး ဖို့ အတွက် တော့ analogWrite ဆိုတဲ့ ဖန်ရှင် ကို သုံးနိုင် ပါတယ်။ ထုတ်ပေး တဲ့အခါ PWM လို့ခေါ်တဲ့ Pulse Width Modulation သုံးပြီး ထုတ်ပေး ပါတယ်။ BBB မှာ PWM စ ခု နဲ့ timer င ခု သုံးလို့ ရပြီး သူတို့ ကို ပုံ J.၁၅ မှာ ပြထား ပါတယ်။ analogWrite အသုံးပြု ဖို့ ပုံစံ ကို အောက်မှာ တွေ့နိုင် ပါတယ်။

```
analogWrite(pin, value, [freq], [callback])
```

အဲဒီက pin မှာ analog တန်ဖိုး ထုတ်ပေး ချင်တဲ့ pin ကို သတ်မှတ် ပေးနိုင် ပါတယ်။ pinMode သတ်မှတ် ဖို့ မလို ပါဘူး။ နောက် value ဆိုတဲ့ argument မှာတော့ PWM ရဲ့ duty cycle အတွက် 0 နဲ့ 1 ကြားက တန်ဖိုး တစ်ခု သတ်မှတ် ပေးနိုင် ပါတယ်။



ပုံ J.၁၆: Analog input အတွက် trimmer (potentiometer) တစ်ခု နှင့် analog output အတွက် LED တစ်ခု ကို ဆက်သွယ်ပုံ။

ထောင့်ကွင်း နဲ့ ပြထားတဲ့ optional arguments တွေက တော့ မသုံး ချင်ရင် ချိန်လှပ် ထားလို့ ရပြီး PWM ရဲ့ frequency နဲ့ လုပ်ဆောင် ပြီးတဲ့ အခါ ခေါ်ဖို့ callback function တွေကို သတ်မှတ် ပေးနိုင်

ပါတယ်။ PWM frequency ကို သတ်မှတ် မပေးရင် default တန်ဖိုး 2 kHz ကို သုံးမှာ ဖြစ် ပါတယ်။ ဖန်ရှင် လုပ်ဆောင်မှု အောင်မြင် ရင် true ကို return ပြန်ပေး မှာ ဖြစ်ပြီး၊ မဟုတ်ရင် false ကို ပြန်ပေး ပါမယ်။

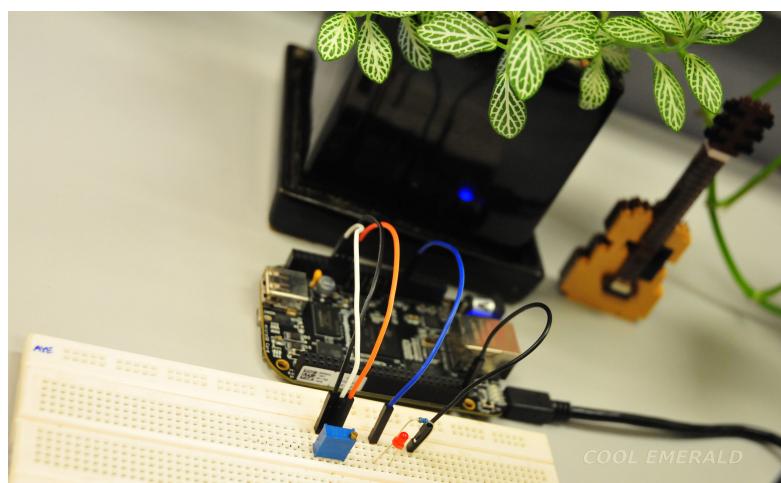
Potentiometer (variable resistor) လေး တစ်ခု က ပြောင်းလဲ ထုတ်လုပ် ပေးတဲ့ ဗိုအား တစ်ခု ကို analogRead နဲ့ ဖတ်ပြီး၊ LED မိုးလုံး ရဲ့ အလင်း အမိန့် ကို analogWrite နဲ့ ထိန်းချုပ် တဲ့ ဆက်သွယ်မှု နှမူနာ တစ်ခု ကို ပုံ J.၁၆ နဲ့ ပုံ J.၁၇ မှာ ပြထား ပါတယ်။ သူရဲ့ ပရီဂရမ် potentiometer.js ကိုတော့ စာရင်း J.၆ မှာ တွေ့နိုင် ပါတယ်။

```

1 var b = require('bonescript');
2 setInterval(rdAnalog,1000);
3 function rdAnalog(){
4     var val = b.analogRead('P9_36');
5     console.log('Value = ' + val)
6     b.analogWrite('P9_14', val);
7 }
```

စာရင်း J.၆: Analog ဗိုအား တစ်ခု ကို ဖတ်၍ LED အလင်းကို ထိန်းချုပ်သော ပရီဂရမ် potentiometer.js ။

Analog ground (P9_34) နဲ့ 1.8V analog Vdd (P9_32) တို့ ကို potentiometer အတွက် power supply ပေးဖို့ သုံးပြီး၊ ထွက် လာတဲ့ analog ဗိုကို AIN5 (P9_36) နဲ့ ဆက်သွယ် အသုံးပြု မှာပါ။



ပုံ J.၁၇: BBB ကို trimmer တစ်ခု နှင့် ဆက်သွယ်ပုံ။

J.6 Interrupt အသုံးပြုခြင်း

BoneScript မှာ pin တစ်ခု ရဲ့ ပြောင်းလဲ မှုကို ထိရောက် စွာ သိရှိ အသုံးချ နိုင်ဖို့ interrupt ဖန်ရှင် တစ်ခု နဲ့ ချိတ်ဆက် နိုင် ပါတယ်။ အဲဒီ အတွက် attachInterrupt ဆိုတဲ့ ဖန်ရှင် ကို သုံးနိုင် ပါတယ်။

```
attachInterrupt(pin, handler, mode, [callback])
```

အဲဒီမှာ ချိတ်ဆက်ချင်တဲ့ pin ကို သတ်မှတ်နိုင် ပါတယ်။ Interrupt event ဖြစ် ပေါ်တိုင်း callback ဖန်ရှင် ကို ခေါ်ချင်ရင် handler ကို true လို့ သတ်မှတ် နိုင် ပါတယ်။ string တစ်ခု ထည့် ပေးရင် တော့ သတ်မှတ် လိုက်တဲ့ အခြေအနေ နဲ့ ကိုက်ညီမှ ခေါ် ပါမယ်။ mode က RISING, FALLING, CHANGE တစ်ခုခု ကို သုံးနိုင် ပါတယ်။ ချိတ်ဆက် ထားတဲ့ interrupt ကို ပြန်ဖြူတ် ချင်ရင် တော့

```
detachInterrupt(pin, [callback])
```

ကို သုံးနိုင် ပါတယ်။

Pin တစ်ခု ရဲ့ ပြောင်းလဲ မှုကို interrupt နဲ့ ချိတ်ဆက် ပြီး LED ကို အဖွင့် အပိတ် လုပ်တဲ့ နမူနာ interrupteg.js interrupteg.js ကို စာရင်း J.7 မှာ တွေ့နိုင် ပါတယ်။ ရှုံးမှာ ဖော်ပြ ခဲ့တဲ့ ပုံ J.22 က ဆက်သွယ်မှု အတိုင်း ပြန် သုံးနိုင် ပါတယ်။

```
1 var b = require('bonescript');
2 var inputPin = 'P8_19';
3 var outputPin = 'P8_13';
4 b.pinMode(inputPin, b.INPUT);
5 b.pinMode(outputPin, b.OUTPUT);
6 b.attachInterrupt(inputPin, true, b.CHANGE, interruptCallback);
7 setTimeout(detach, 30000);
8 function interruptCallback(x) {
9     console.log(JSON.stringify(x));
10    b.digitalWrite(outputPin,x.value);
11 }
12 function detach() {
13     b.detachInterrupt(inputPin);
14     console.log('Interrupt detached');
15 }
```

စာရင်း J.7: Interrupt ကို သုံး၍ LED ကို ထိန်းချုပ်သော ပရိုဂရမ် interrupteg.js ။

J.7 Text ဖိုင်များကိုရေးဖတ်ခြင်း

BoneScript နဲ့ text ဖိုင် တွေကို ရေးတာ၊ ဖတ်တာ လုပ်ဖို့ အတွက် `readTextFile` နဲ့ `writeTextFile` ဖန်ရှင် တွေကို သုံးနိုင် ပါတယ်။ `readTextFile` ရဲ့ ပုံစံ က အောက်က အတိုင်း ဖြစ် ပါတယ်။

```
readTextFile(filename, [callback])
```

အဲဒီမှာ `filename` က ဖတ်ချင်တဲ့ ဖိုင်ရဲ့ `path` လမ်းကြောင်း အပြည့် အစုံ ဖြစ် ပါတယ်။ `callback(x)` အတွက် `x.data` က ဖိုင် ထဲက `data` တွေ ဖြစ်ပြီး၊ `x.error` က `error message` ဖြစ် ပါတယ်။ `Return` တန်ဖိုး အနေနဲ့ ဖိုင်ထဲက ဟာတွေ အားလုံး ကို `string` တစ်ခု အနေ နဲ့ ရမှာ ဖြစ် ပါတယ်။ `writeTextFile` ရဲ့ ပုံစံ ကိုတော့ အောက်က အတိုင်း တွေ့နိုင် ပါတယ်။

```
writeTextFile(filename, data, [callback])
```

အဲဒီမှာ `filename` က ရေးချင်တဲ့ ဖိုင်ရဲ့ `path` လမ်းကြောင်း အပြည့် အစုံ ဖြစ် ပါတယ်။ `data` က ရေးဖို့ အတွက် ASCII string ဖြစ်ပြီး၊ `callback(x)` မှာ `x.err` က `error message` ဖြစ် ပါတယ်။

Text ဖိုင် တစ်ခု ကို ရေးဖတ် တဲ့ ပရိုဂရမ် နမူနာ `textfile.js` ကို စာရင်း [J.7](#) မှာ တွေ့နိုင် ပါတယ်။ စစ်ခြင်း Cloud9 IDE ကို ပဲ သုံးပြီး `mytext.txt` ဆိုတဲ့ ဖိုင် တစ်ခုကို ဖန်တီးပြီး Cloud9 အခန်း ဖြစ်တဲ့ `/var/lib/cloud9/` ထဲ မှာပဲ သိမ်းလိုက် ပါမယ်။ ပရိုဂရမ် ကို `run` လိုက်တဲ့ အခါ မူရင်း စာသား တွေနဲ့ အသစ် ရေးလိုက်တဲ့ စာသား တွေကို အောက်က `console` ထဲမှာ တွေ့ရ မှာဖြစ် ပါတယ်။

```
1 var b = require('bonescript');
2 var file='/var/lib/cloud9/mytext.txt';
3 var od = b.readTextFile(file);
4 console.log('Data = ' + od);
5 b.writeTextFile(file, 'Written content.', writeStatus);
6 b.readTextFile(file, printStatus);
7 function writeStatus(x) {
8     console.log(JSON.stringify(x));
9 }
10 function printStatus(x) {
11     console.log('x.data = ' + x.data);
12     console.log('x.err = ' + x.err);
13 }
```

စာရင်း [J.7](#): Text ဖိုင် တစ်ခု ကို ရေးဖတ် ခြင်း။

အကိုးအကားများ

J9

အကိုးအကားများ

- [Bea14] BeagleBoard. BoneScript - Physical computing library in JavaScript for Node.JS and the browser. 2014. url: <http://beagleboard.org/project/bonescript>.

J®

အခိုး J. BONESCRIPT

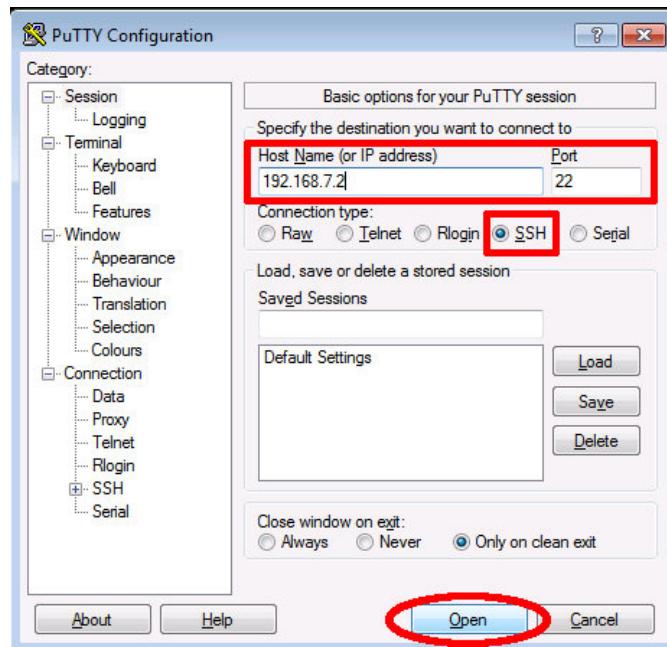
အခန်း ၃

အခြေခံလုပ်ဆောင်မှုပျား

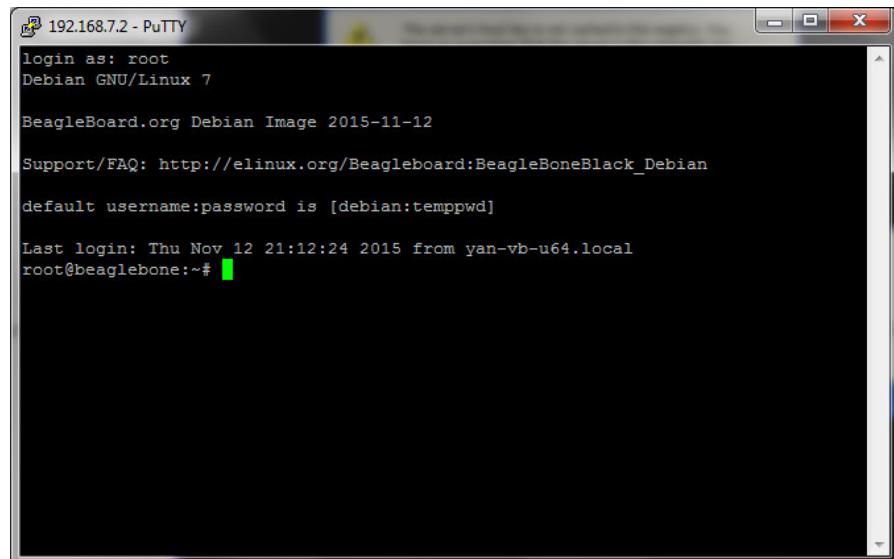
၃.၁ SSH ဆက်သွယ်အသုံးပြုခြင်း

BBB ကို SSH (Secure SHell) နဲ့ လုမ်းပြီး ဆက်သွယ် အသုံးပြု ဖို့ SSH client တစ်ခုခု ကို အသုံးပြု နိုင် ပါတယ်။ Windows အတွက် ဆိုရင် PuTTY ကို <http://www.putty.org/> ကနေ ယူပြီး တပ်ဆင် အသုံးပြု လို့ရ ပါတယ်။ ပြီးတဲ့ အခါ PuTTY ကို ဖွင့်ပြီး BBB ၏ IP address ဥပမာ USB ကြိုးနဲ့ ဆက်ထား တဲ့ အချိန် ဆို 192.168.7.2 ကို ပုံ ၃.၁ မှာ ပြထား သလို ထည့်ပြီး ဆက်သွယ် နိုင် ပါတယ် [Mon15]။

Security alert ပေါ်လာရင် Yes ကို ရွှေးလို ရပြီး၊ Login as မှာ username နဲ့ password ကို ရိုက်ထည့် လိုက်ရင် ပုံ ၃.၂ မှာလို BBB နဲ့ ဆက်သွယ် သွား တာကို တွေ့နိုင် ပါတယ်။ Debian ဗားရှင်း အဟောင်း မှာ username က root နဲ့ password အတွက် ဘာမှ မထည့်ပဲ enter နှိပ်လိုက် ရင် ရပါတယ်။ နောက်ပိုင်း firmware အသစ် တွေမှာ တော့ username က debian နဲ့ password က temppwd ဖြစ် ပါတယ်။



ပုံ ၃.၁: PuTTY အသုံးပြု၍ ဆက်သွယ်ခြင်း။



ပုံ ၃.၂: SSH လုပ်ဆောင်ပုံ။

Linux ဒါမ္မ မဟုတ် Mac ကို သုံးတယ် ဆိုရင် တော့ terminal ကို ဖွင့်ပြီး အောက်က စာရင်း ၃.၂ မှာ ပြထား တဲ့ command ကို ထည့်ပြီး ဆက်သွယ် နိုင် ပါတယ် (ပုံ ၃.၃)။

```
1 $ ssh root@192.168.7.2
```

စာရင်း ၃.၁: 'root' အတွက် SSH ကိုသုံး၍ ဆက်သွယ်ခြင်း။

```
1 $ ssh debian@192.168.7.2
```

စာရင်း ၃.၂: Username 'debian' အတွက် SSH ကိုသုံး၍ ဆက်သွယ်ခြင်း။

အဲဒီမှာ root က username အတွက် ဖြစ်ပြီး နောက်ပိုင်း အသစ်တွေ အတွက် ဆိုရင် root အစား debian ကို သုံးဖို့ လိုပြီး Are you sure you want to continue connecting (yes/no)? လို့ မေးတဲ့ အခါ yes ကိုရှိက်ထည့် နိုင် ပါတယ်။ နောက်တစ်ခါ password တောင်းတဲ့ အခါမှာ root အတွက် ဆိုရင် ဘာမှ မထည့်ပဲ enter နိုင်နိုင်ပြီး ၏ debian အတွက် ဆိုရင် temppwd ကို ထည့်ပေး နိုင် ပါတယ်။

```
yan@yanhpu:~$ ssh root@192.168.7.2
Debian GNU/Linux 7

BeagleBoard.org Debian Image 2015-11-12

Support/FAQ: http://elinux.org/Beagleboard:BeagleBoneBlack_Debian

default username:password is [debian:temppwd]

Last login: Thu Nov 12 19:07:37 2015 from yanhpulocal
root@beaglebone:~#
```

ဦး ၃.၃: Linux SSH session

BeagleBone ကို SSH နဲ့ လွမ်း ဆက်သွယ် တဲ့ အခါ password တွေ ပြန်ပြန် မထည့် ချင်ရင် စာရင်း ၃.၃ မှာ ပြထား သလို လုပ်ထား လိုရ ပါတယ်။ Passphrase တို့ ဘာတို့ တောင်းတဲ့ အခါ ဘာမှ မထည့်ပဲ enter ပဲ နိုင်ပြီး ဆက်သွား လို ရပါတယ်။

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

စာရင်း ၃.၄: SSH အတွက် ပြင်ဆင်ခြင်း။

၃.၁.၁ Password ကို ပြောင်းခြင်း

အဲဒီ default username နဲ့ password က BBB ဘူတ် အားလုံး အတူတူ ဖြစ်တာ ကြောင့် security ရှိချင်ရင် အောက်က command သုံးပြီး ပြောင်းလို့ ရပါတယ်။

```
$ sudo passwd debian
```

နောက် သတိပြု သင့်တာ တစ်ခုက <http://beaglebone.local:3000/ide.html> မှာ ရှိတဲ့ Cloud9 ရဲ့ bash မှာ root က ပွင့်နေတာ ကြောင့် root အတွက် password ကို လည်းပြောင်း၊ Cloud9 ကို လည်း ဖြုတ်နိုင် ပါတယ်။

တကယ်လို့ root user ကို enable လုပ်ချင်ရင် တော့

```
$ sudo passwd root
```

ကို ရိုက်ထည့်ပြီး 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။

၃.၁.၂ Password ကိုဖြတ်ခြင်း

User တစ်ယောက် ဥပမာ yan ရဲ့ password ကို ဖြုတ်ချင်ရင် အောက်က အတိုင်း ဖြုတ်နိုင် ပါတယ်။

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

၃.၁.၃ User အသစ်ဖန်တီးခြင်း

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

```
$ sudo adduser yan
```

၃.၁.၄ Sudoers

ပုံမှန်အားဖြင့် debian ဆိုတဲ့ user က sudoer ဖြစ်ပါတယ်။ သူက command တွေကို root အနေနဲ့ run ချင်ရင် ရွှေမှာ sudo ခံပြီး run လို ရပါတယ်။ ဖန်တီး လိုက်တဲ့ user အသစ် ကို sudoer ဖြစ်စေ ချင်ရင်

```
$ sudo visudo
```

ကို သုံးပြီး ပြင်နှင့် ပါတယ်။ အဲဒီမှာ

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

ဆိုတဲ့ လိုင်းတွေ ကို လိုက်ရှာပြီး root ရဲ့ 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
```

၃.၁.၅ User ဖျက်ခြင်း

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

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

၃.၂ ဖိုင်များပေးပို့ရယူခြင်း

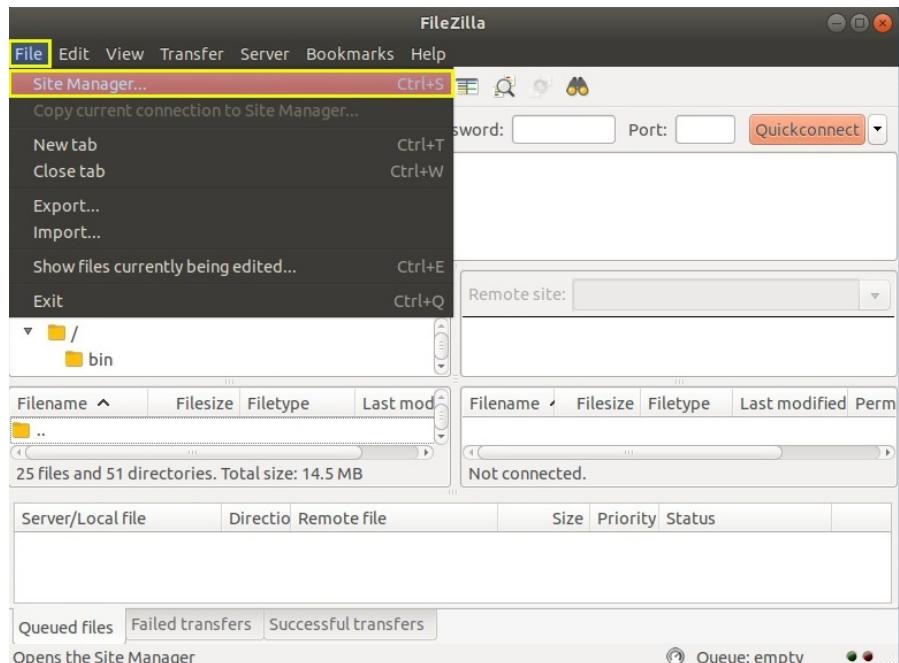
BBB နဲ့ desktop ကွန်ပျိုတာ နဲ့ ဖိုင်တွေ အပြန် အလုန် ပေးပို့ ဖလှယ် တဲ့ အကြောင်း ဆွေးနွေးချင် ပါတယ်။ နဲ့ အနေ နဲ့ scp command ကို ကွန်ပျိုတာ terminal ကနေ သုံးပြီး၊ test.cpp ဆိုတဲ့ ဖိုင်ကို BBB သိ လုမ်းပိုတဲ့ ပုံစံ တစ်ခု ကို အောက်မှာ ပြထား ပါတယ် (ပုံ ၃.၄)။

```
$ scp test.cpp debian@192.168.2.92:/home/debian/
```

```
yan@ubuntu17:~$ scp test.cpp debian@192.168.2.92:/home/debian/
debian@192.168.2.92's password:
test.cpp                                         100%   107    13.1KB/s  00:00
yan@ubuntu17:~$
```

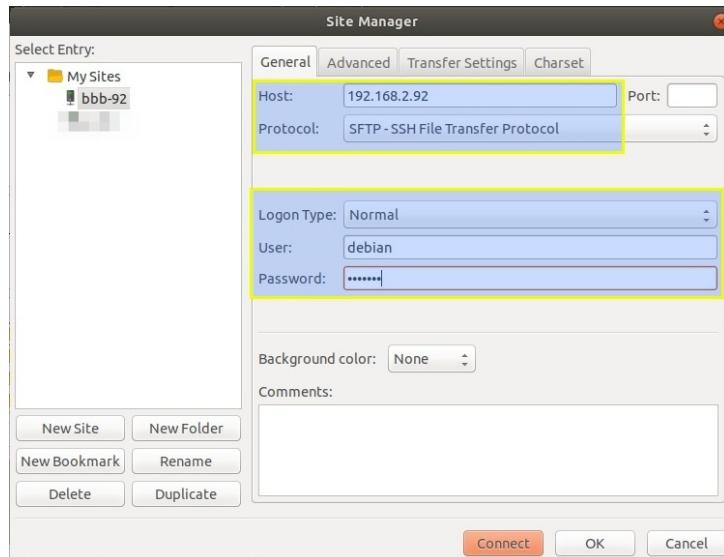
ပုံ ၃.၄: scp ဖြင့် ဖိုင်ပို့ခြင်း။

ပို့ပြီး အဆင်ပြေ တဲ့ နည်းကတော့ FileZilla စတဲ့ ftp client တွေကို သုံးတဲ့ နည်း ဖြစ်ပါ တယ်။ FileZilla ကို တပ်ဆင်ပြီး၊ ဖွင့်ပြီး တဲ့ အခါ ပုံ ၃.၅ မှာ ပြထား သလို File menu→Site Manager... ကို နှိပ် ပါမယ်။

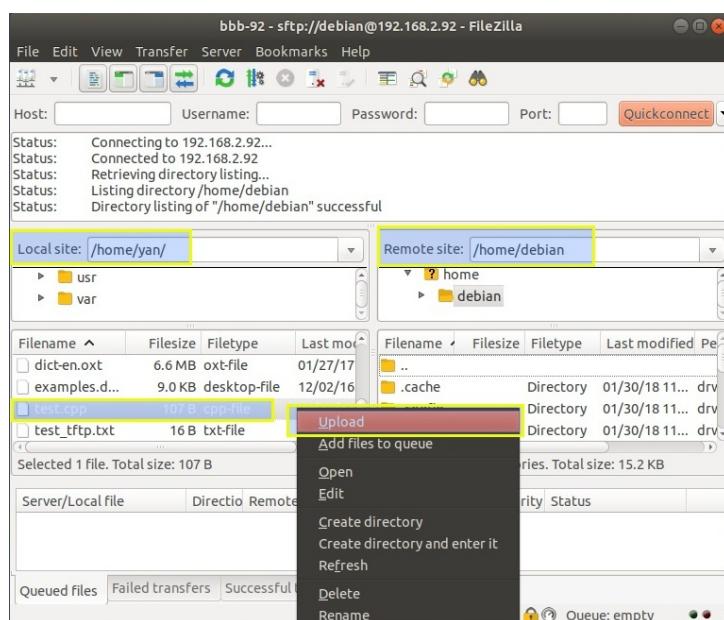


ပုံ ၃.၅: FileZilla ကို သုံးခြင်း။

Site Manager စံးပါးပေါ်လာ တဲ့ အခါ host အတွက် IP address ၊ protocol အတွက် SFTP တွေကို ရွေး၊ Logon type ကို normal နဲ့ username ၊ password တွေ ထည့်ပြီး တဲ့ အခါ connect ကို နှိပ်ပြီး ဆက် သွယ်နိုင် ပါတယ် (ပုံ ၃.၆)။



ပုံ ၃.၆: SFTP ဖြင့် ဆက်သွယ်ခြင်း။



ပုံ ၃.၇: FileZilla ဖြင့် ဖိုင်များ ပေးပို့ ရယူခြင်း။

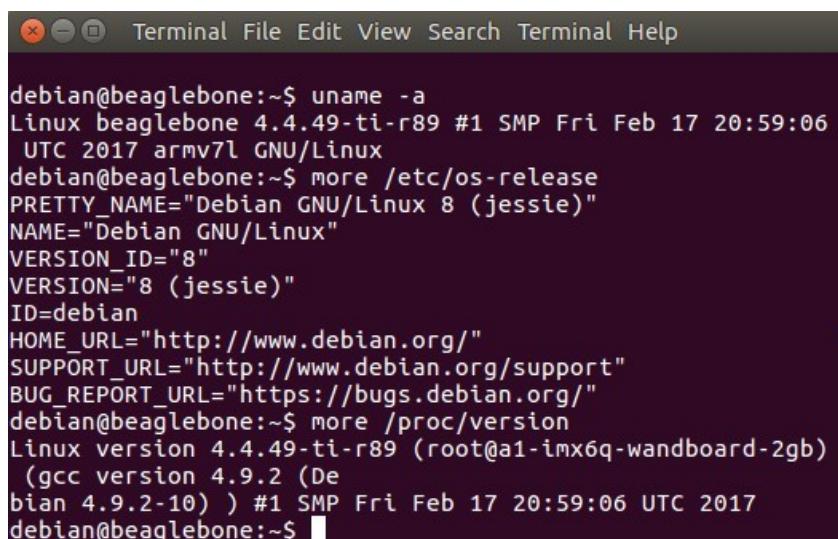
ဆက်သွယ် ပြီးတဲ့ အခါ local site နဲ့ remote site တွေကို သတ်မှတ်၊ ဖိုင်တွေ၊ အခန်း တွေ ပေါ်မှာ ညာဖက် ကလစ် နှုပ်ပြီး ပိုတာ၊ လက်ခံတာ၊ ဖျက်တာ တွေကို လွယ်လွယ် ကူကူ၊ လျင်လျင် မြန်မြန် လုပ်နိုင် ပါတယ် (ပုံ ၃.၇)။

၃.၂ ဘုတ်ကို update လုပ်ခြင်း

Beagle ဘုတ် အသစ် တစ်ခု ဝယ်လို ရလာ တဲ့အခါ ပါလာ တဲ့ software image က များသော အားဖြင့် ဗားရှင်း အဟောင်း တွေပဲ ပါလာ လေ့ရှိ ပါတယ်။ ဥပမာ ၂၀၁၇ မှာ ဝယ်တဲ့ BeagleBone Black Rev C ရဲ့ onboard 4 GB eMMC မှာ Debian Linux ကို တပ်ဆင် ထားပြီး ဘုတ်ကို ကွန်ပျူတာ နဲ့ ချိတ်ဆက် လိုက်ရင် ပေါ်လာ တဲ့ flash drive ထဲက ID.txt ဆိုတဲ့ ဖိုင် ထဲမှာ

BeagleBoard.org Debian Image 2015-11-12

လို တွေ့ရ ပါတယ်။ ရှေ့က ပုံ ၃.၂ ထဲမှာ လို ssh ဝင်လိုက် တဲ့ အခါ မှာလည်း Debian Image ရဲ့ ဗားရှင်း ကို ပြတာကို တွေ့နိုင် ပါတယ်။ ၂၀၁၇ မှာပဲ ဝယ်တဲ့ BeagleBone Blue အတွက် တော့ BeagleBoard.org Debian Image 2017-02-19 လို တွေ့ရ ပါတယ်။ System information တွေ ကြည့်ဖို့ အတွက် အောက်က command တွေကို သုံးနိုင် ပါတယ် (ပုံ ၃.၈ နှင့် စာရင်း ၃.၅)။



```

Terminal File Edit View Search Terminal Help

debian@beaglebone:~$ uname -a
Linux beaglebone 4.4.49-ti-r89 #1 SMP Fri Feb 17 20:59:06
UTC 2017 armv7l GNU/Linux
debian@beaglebone:~$ more /etc/os-release
PRETTY_NAME="Debian GNU/Linux 8 (jessie)"
NAME="Debian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=debian
HOME_URL="http://www.debian.org/"
SUPPORT_URL="http://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
debian@beaglebone:~$ more /proc/version
Linux version 4.4.49-ti-r89 (root@a1-imx6q-wandboard-2gb)
(gcc version 4.9.2 (De
bian 4.9.2-10) ) #1 SMP Fri Feb 17 20:59:06 UTC 2017
debian@beaglebone:~$ 

```

ပုံ ၃.၈: BeagleBone Blue ၏ အချက်အလက်များ။

```

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

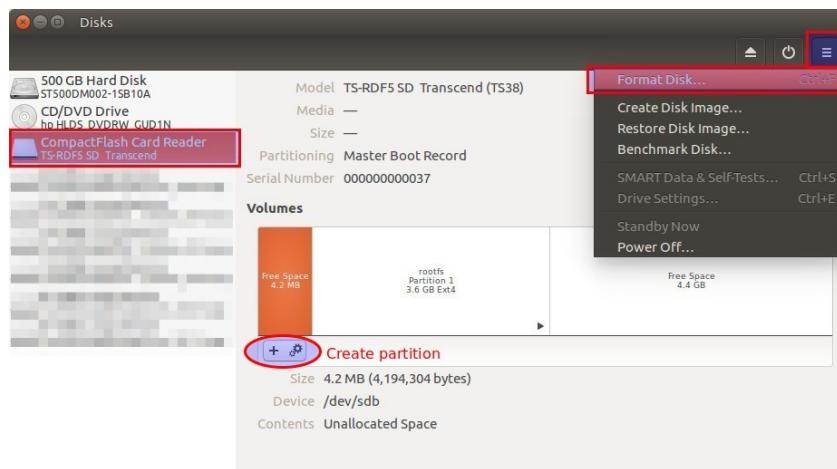
```

ঠাৰ্ড: ২.৫: System অবৃক্ষালগ্নমূলক ক্ষেত্ৰেন্স:

গুর্তকৃ ফেওক পৰি software image কৈ update লাভেন্স আটুক <http://beagleboard.org/latest-images> কৃত বুা পিয়েড [Bea17]। অৱে বৰ্তুৱা BeagleBone Black, Blue দত্ত গুর্তকৈ আটুক লগ্নৰী ফেওক পৰি image প্ৰেছ তো Debian 9.1 2017-08-31 4GB SD LXQT ক্ষেত্ৰে image কৃ download লাভেন্স পিয়েড।



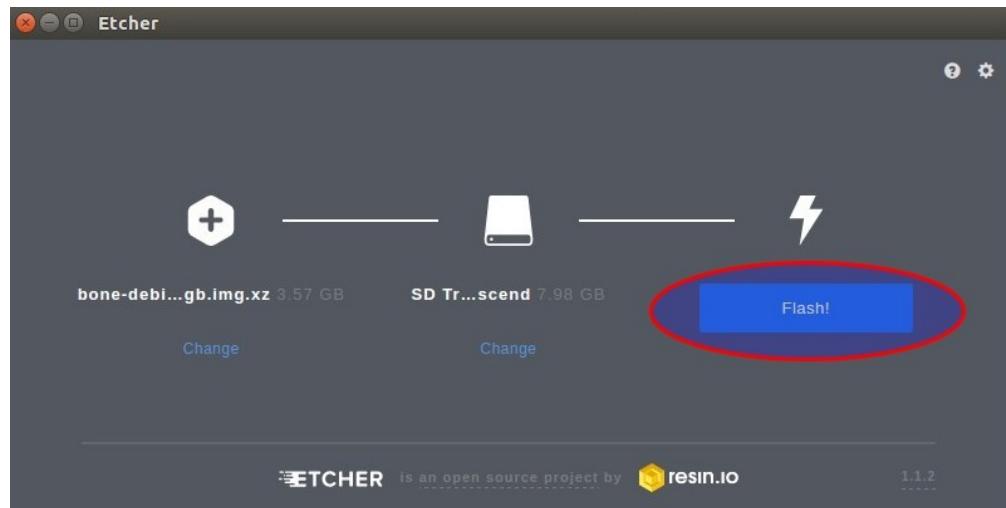
ঠি ২.৬: Disks application



ঠি ২.৭: Format লাভেন্স কৰি Partition create লাভেন্স:

နောက် တစ်ခါ <https://etcher.io/> ကို သွားပြီး Etcher ကို ရယူ တပ်ဆင် လိုက်ပါမယ်။ ပြီးတဲ့ အခါ ဗုဒ္ဓာ ကို ကွန်ပျူးတာ ရဲ့ Card Reader ဒါမဲ မဟုတ် USB adapter သုံးပြီး တပ်ဆင် လိုက် ပါမယ်။ Ubuntu Linux ရဲ့ Disks ဆိုတဲ့ application (ပုံ ၃.၉) 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။

Etcher application ကို ဖွင့်လိုက် တဲ့ အခါ ဗုဒ္ဓာ ကို အလို အလောက် ရွေးချယ် ပြုသ နေပါ လိမ့်မယ်။ လိုအပ် ရင် ရေးမယ့် ကုန် ကို ပြောင်း သတ်မှတ် ပေးနိုင် ပါတယ်။ Source file အတွက် ခုန က download လုပ် ထားတဲ့ bone-debian-9.1-lxqt-armhf-2017-08-31-4gb.img.xz ကို ရွေးပေး လိုက် ပါမယ်။ ပြီးတဲ့ အခါ ပုံ ၃.၁၂ မှာ ပြထားတဲ့ Flash ခလုတ် ကို နှိပ်ပြီး ဗုဒ္ဓာ ပေါ်ကို software image ထည့်သွင်း နိုင် ပါတယ်။



ပုံ ၃.၁၁: Software image ကို SD Card တွင် ထည့်သွင်းခြင်း။

ရလာတဲ့ ဗုဒ္ဓာ ကို BBB မှာ ထည့်ပြီး ကွန်ပျူးတာ USB နဲ့ ဆက်ပြီး power ပေးလိုက် ရင် ဗုဒ္ဓာ ပေါ်က နောက်ဆုံး software ဗားရှင်းကို သုံးပြီး စက်က တက် လာ ပါမယ်။ SSH သုံးပြီး ဆက်သွယ် ပြီးတဲ့ အခါ စာရင်း ၃.၄ က command တွေ ပြန်သုံးပြီး ကြည့်နိုင် ပါတယ်။ အဲဒီ အခါ 2017 Aug ဗားရှင်း ဖြစ်သွား တာကို တွေ့နိုင် ပါတယ်။

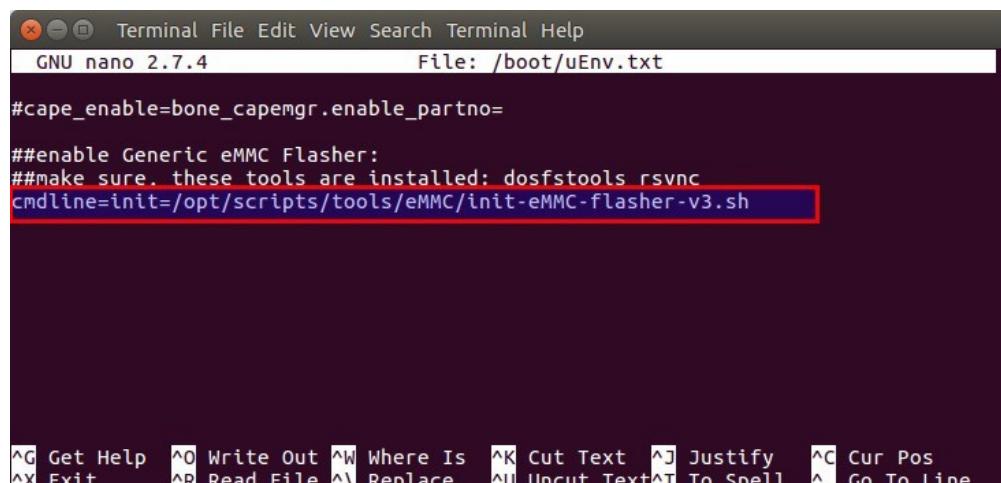
၃.၃.၁ eMMC ပေါ်သို့ ထည့်သွင်းခြင်း

အဲဒီ image ကို μSD card ပေါ်ကနေ eMMC ပေါ်ကို ပြောင်းထည့် ချင်ရင် /boot/uEnv.txt ဆိုတဲ့ ဖိုင်ကို SSH terminal မှာ အောက်က command သုံးပြီး edit လုပ် ဖို့လိုပါတယ်။

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

ပွင့်လာ တဲ့ ဖိုင်ထဲမှာ အောက်က လိုင်းတွေ ကို လိုက်ရှာပြီး cmdline = init = /opt/scripts/tools/eMMC/init-eMMC-flasher-v3.sh ဆိုတဲ့ လိုင်း ရွှေက comment လုပ်ထားတဲ့ # သကော်တာ ကို ဖြုတ်ပြီး enable လုပ်လိုက် ပါမယ်။

```
##enable Generic eMMC Flasher:  
##make sure, these tools are installed: dosfstools rsync  
cmdline=init=/opt/scripts/tools/eMMC/init-eMMC-flasher-v3.sh
```



ပုံ ၃.၁၂: eMMC ကို update လုပ်ရန် /boot/uEnv.txt ကို ပြုပြင်ခြင်း။

ပြီးတဲ့ အခါ Ctrl+O နဲ့ enter ရိုက် သိမ်းလိုက်ပြီး၊ Ctrl+X နှုပ်ပြီး ထွက် လိုက်ပါမယ်။ ဘုတ်ရဲ့ ပါဝါ ကို ပိတ်ပြီး ပြန် ဖွင့် ပေးလိုက် ရင် eMMC ပေါ်ကို image အသစ်ကို ထပ်ရေးမှာ ဖြစ်ပြီး LED မီးလုံး က ပြေးနေ တာကို တွေ့ရ ပါမယ်။ အချိန် တော်တော် ကြာပြီး တဲ့ အခါ flash လုပ်တာ ပြီးသွား ပြီးတဲ့နောက် LED မီးအား လုံး မှတ်သွား ပါလိမ့်မယ်။

μSD card ကို ပြန်ဖြုတ်ပြီး စက်ကို ပြန်ဖွင့် လိုက်ရင် eMMC ပေါ်က တက်လာတဲ့ စက်ရဲ့ ဗားရှင်း က

အသစ် ဖြစ်သွား တာကို တွေ့နှင့် ပါတယ်။ μSD card ကို မဖြတ်ရင် နောက်တစ်ခါ ထပ် flash လုပ်မှာ မို့ မမေ့အောင် သတိပြု ဖို့ လိုပါတယ်။

၃.၃.J μSD Card ၏ File System Partition ကို ချွဲခြင်း

μSD card ပေါ်မှာ BeagleBoards တွေရဲ့ image တွေ ရေးပြီးတော့၊ အဲဒီ card ကနေ boot လုပ်လိုက် တဲ့ အခါ 4 GB လောက်ပဲ သုံးလို့ ရတာ ကို တွေ့ရ မှာပါ။ ကိုယ့်ရဲ့ μSD card အရွယ် အစား က ပို့ကြီးပြီး၊ အဲဒီ storage space တွေ အကုန် သုံးချင် တယ် ဆိုရင် သူကို re-partition ပြန်လုပ် ဖို့ လိုပါတယ် [Wik17d; Ell17]။ အဲဒီ အတွက် BBB ကို SSH နဲ့ ဆက်သွယ် ပြီးတဲ့ အခါ အောက်က command နဲ့ super user ဖြောင်းလိုက် ပါမယ်။

```
$ sudo -i
```

စက်မှာ ရှိတဲ့ volumes တွေကို စစ်ကြည့် ဖို့ အတွက် အောက်က အတိုင်း list လုပ်ကြည့် တဲ့ အခါ ပုံ ၃.၁၃ မှာ ပြထား သလိုမျိုး တွေ့နှင့် ပါတယ်။

```
# ls -l /dev/mmcblk*
```

```
root@beaglebone:~# ls -l /dev/mmcblk*
brw-rw---- 1 root disk 179,  0 Aug 31 16:53 /dev/mmcblk0
brw-rw---- 1 root disk 179,  1 Aug 31 16:53 /dev/mmcblk0p1
brw-rw---- 1 root disk 179,  8 Aug 31 16:53 /dev/mmcblk1
brw-rw---- 1 root disk 179, 24 Aug 31 16:53 /dev/mmcblk1boot0
brw-rw---- 1 root disk 179, 32 Aug 31 16:53 /dev/mmcblk1boot1
brw-rw---- 1 root disk 179,  9 Aug 31 16:53 /dev/mmcblk1p1
```

ပုံ ၃.၁၃: စက်ရှိ available volume များကို စစ်ခြင်း။

အဲဒီ မှာ ပြနေတဲ့ စာရင်း ထဲက eMMC ရဲ့ အထူး boot0/1 partitions တွေကို ထည့် မတွက် ဘူး ဆိုရင် μSD card ရဲ့ mmcblk0 နဲ့ eMMC ရဲ့ mmcblk1 တို့ကို partition ကိုယ်စီ နဲ့ တွေ့နှင့် ပါတယ်။

အဲဒီနောက် μSD card ကို fdisk နဲ့ အောက်က အတိုင်း သုံးပါမယ်။ ပြီးတဲ့ အခါ ပုံ ၃.၁၄ မှာ ပြထား သလို p ကို ရိုက်ထည့် ပြီး print out ကို ကြည့်နိုင် ပါတယ်။

```
# fdisk /dev/mmcblk0
```

```
root@beaglebone:~# fdisk /dev/mmcblk0
Welcome to fdisk (util-linux 2.29.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): p
Disk /dev/mmcblk0: 7.4 GiB, 7983857664 bytes, 15593472 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x7f863e1e

Device            Boot Start    End Sectors Size Id Type
/dev/mmcblk0p1 *      8192 6963199 6955008 3.3G 83 Linux
```

ပုံ ၃.၁၄: စက်ရှု available volume များကို စစ်ခြင်း။

ထွက်လာ တဲ့ print out မှာ ကိုယ် သုံးတဲ့ μSD card ပေါ်မှုတည် ဖြီး ကွာခြားမှု တရာ့ရှိနိုင် ပါတယ်။ ဒီ နမူနာ မှာတော့ 7.4 GB disk မှာ partition အရွယ် က 3.3 GB ပဲ ရှိတာ ကို တွေ့နိုင် ပါတယ်။ အဲဒီ မှာ start sector ဖြစ်တဲ့ 8192 ကို မှတ်ထား ဖို့ အရေးကြီး ပါတယ်။

```
Command (m for help): d
Selected partition 1
Partition 1 has been deleted.

Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): ←
Using default response p.
Partition number (1-4, default 1): ←
First sector (2048-15593471, default 2048): 8192
Last sector, +sectors or +size[K,M,G,T,P] (8192-15593471, default 15593471): ←
Created a new partition 1 of type 'Linux' and of size 7.4 GiB.
Partition #1 contains a ext4 signature.

Do you want to remove the signature? [Y]es/[N]o: n
Command (m for help):
```

ပုံ ၃.၁၅: Partition ပြန်ဖန်တီးခြင်း။

အဲဒီနောက် d ကို နှိပ်ဖြီး partition ကို ဖျက်လိုက် ပါမယ်။ ဖြီးတဲ့ အခါ n ကို နှိပ်ဖြီး new partition ကို ဖန်တီး လိုက် ပါမယ်။ ဖန်တီး တဲ့ အခါ partition type အတွက် enter ပဲ နှိပ်ဖြီး default အတိုင်း primary ကို ရွေးလိုက် ပါမယ်။ နောက် တစ်ခါ partition number အတွက်လည်း default အတိုင်း 1 ကို ပဲ ထားဖို့ enter ထပ်နိုင် ပါမယ်။ First sector အတွက် ကို တော့ ခုန က မှတ်ထား တဲ့ start sector နံပါတ် ကို ပြန် ထည့် ဖို့ လို ပါတယ်။ Last sector အတွက် ကိုတော့ default အတိုင်း max possible size

ကို ရွေးနိုင် ပါတယ်။ အဲဒီ မှာ ext4 signature ကို ဖျက် မလား မေးရင် မဖျက်ဖို့ n ကို ရွေး ဖို့ လိုပါတယ် (ပုံ ၃.၁၅)။

နောက် တစ်ခါ p ကို ပြန် ထည့်ပြီး partition table အသစ် ကို ပြန် ကြည့် နိုင် ပါတယ်။ အချယ် အစား 7.4 GB ပြောင်း သွား တာကို တွေ့နိုင် ပါတယ်။ စိတ်ကြိုက် ဖြစ်ပြီ ဆိုရင် write လုပ်ဖို့ w ကို ရိုက်ထည့် ပြီး reboot လုပ်လိုက် ပါမယ် (ပုံ ၃.၁၆)။

```
Command (m for help): [p]
Disk /dev/mmcblk0: 7.4 GiB, 7983857664 bytes, 15593472 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x7f863e1e

Device      Boot Start     End Sectors [Size] Id Type
/dev/mmcblk0p1        8192 15593471 15585280 [7.4G] 83 Linux

Command (m for help): [w]
The partition table has been altered.
Calling ioctl() to re-read partition table.
Re-reading the partition table failed.: Device or resource busy

The kernel still uses the old table. The new table will be used at the next reboot
or after you run partprobe(8) or kpartx(8).

root@beaglebone:~# reboot
```

ပုံ ၃.၁၆: Partition table ကို ပြောင်းခြင်း။

```
debian@beaglebone:~$ sudo -i
[sudo] password for debian:
root@beaglebone:~# df -h .
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk0p1  3.3G  2.6G  470M  85% /
root@beaglebone:~# resize2fs /dev/mmcblk0p1
resize2fs 1.43.4 (31-Jan-2017)
Filesystem at /dev/mmcblk0p1 is mounted on /; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/mmcblk0p1 is now 1948160 (4k) blocks long.

root@beaglebone:~# df -h .
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk0p1  7.3G  2.6G  4.4G  38% /
```

ပုံ ၃.၁၇: File system ကို ချေခြင်း။

Reboot လုပ်ပြီး တဲ့ အခါ နောက်ဆုံး အဆင့် အနေနဲ့ အောက်က command တွေ သုံးပြီး file system ကို extend လုပ်နိုင် ပါတယ် (ပုံ ၃.၁၇)။

```
$ sudo -i
# df -h .
# resize2fs /dev/mmcblk0p1
```

```
# df -h .
```

၃.၄ Backup နှင့် Restore ပြလုပ်ခြင်း

ကိုယ့် BBB အတွက် ကိုယ်ဟာ ကိုယ် စိတ်တိုင်းကျ setup လုပ်ထားတဲ့ μSD Card ကို backup လုပ်ထားတာ ကောင်းပါတယ်။

၃.၄.၁ Image ဖိုင်ပွားယူ၍ကူးခြင်း

Backup လုပ်ဖို့ အတွက် Windows စက်တွေမှာ Win32 Disk Imager ကို သုံးလို့ရပြီး၊ Linux နဲ့ Mac တွေ အတွက် တော့ dd ဆိုတဲ့ command ကို သုံးနိုင် ပါတယ် [Tib17; ras17a]။ နမူနာ အနေနဲ့ Linux ပေါ်မှာ dd ကို သုံးပြီး backup လုပ်တဲ့ အကြောင်း ဖော်ပြု ပါမယ်။ စစ်ခြင်း μSD card ကို ကွန်ပူးတာ မှာ မတပ်ခင်

```
$ df -h
```

ဆိုတဲ့ command ကို ရိုက်ထည့်ပြီး ရလာတဲ့ စာရင်း ကို μSD Card တပ်ပြီး အဲဒီ command ကို ပြန်သုံးတဲ့ အခါ ရလာတဲ့ စာရင်း နဲ့ နှိမ်းယဉ် ကြည့် ပါမယ်။ ဥပမာ ကျွန်ုတ်တော့ စက်မှာ /dev/sdb1 နဲ့ /dev/sdb2 ဆိုတဲ့ partition နှစ်ခု μSD Card အတွက် ပေါ်လာ တာကို တွေ့ရ ပါတယ် (ပုံ ၃.၁၈)။

| 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 ၃.၁၈: SD Card ၏ path ကို စစ်ထည့်ခြင်း။

Disk တစ်ခု လုံး ကူးချင် တာမို့ အောက်မှာ ပြထားတဲ့ အတိုင်း dd ရဲ့ input အနေနဲ့ /dev/sdb လို့ ပေးလိုက် တဲ့ အခါ partition နှစ်ခု လုံးကို output အနေနဲ့ ပေးထားတဲ့ bbb1.img ဆိုတဲ့ ဖိုင် အနေနဲ့ backup လုပ်ပေးပါ လိမ့်မယ်။ Optional အနေနဲ့ block size ကို 1M စသည်ဖြင့် ထည့် သတ်မှတ် နိုင် ပါတယ် (ပုံ ၃.၁၉)။

```
$ sudo dd if=/dev/sdb of=/home/yan/bbb1.img bs=1M
```

```
yan@hp:~$ sudo dd if=/dev/sdb of=/home/yan/bbb1.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 ၃.၁၉: dd ကို အသုံးပြုခြင်း။

သိမ်းထားတဲ့ image ဖိုင် ကို μSD Card ပေါ်ပြန်ထည့် ချင်ရင် အဲဒီ command ကို ပဲ input နဲ့ output ကို ပြောင်းပြီး ပြန်သုံး နိုင် ပါတယ်။ နောက် တစ်နည်း အနေနဲ့ အပိုင်း ၃.၃ က အတိုင်း Etcher ဆိုတဲ့ application ကို သုံးပြီး image ဖိုင်ကို ကူးရေး နိုင် ပါတယ်။

Etcher application ကို ဖွင့်လိုက် တဲ့ အခါ μSD Card ကို အလို အလျောက် ရွေးချယ် ပြသ နေမှာ ဖြစ်ပြီး၊ လိုအပ်ရင် ရေးမယ့် ကို ပြောင်း သတ်မှတ် ပေးနိုင် ပါတယ်။ Source file အတွက် ခုန က ဖန်တီး ထားတဲ့ bbb1.img ကို ကို ရွေးပေး လိုက် ပါမယ်။ ပြီးတဲ့ Flash ခလုတ် ကို နိုပ်လိုက် ရင် ရှေ့က ပုံ ၃.၁၂ မှာ ပြထားတဲ့ နည်းတူ μSD Card ပေါ်ကို ကူးထည့် ပေး သွား မှာ ဖြစ် ပါတယ်။

၃.၅ Network ပြင်ဆင်ခြင်း

BBB board ကို Ethernet ကြိုး တပ်ဆင် အသုံးပြု ဖို့ ပြင်ဆင် ပါမယ်။ BBB ကို Ethernet network ကြိုး တပ်ပြီး တဲ့ အခါ SSH terminal မှာ ifconfig ကို ရိုက်ထည့် ပြီး လက်ရှိ network setting တွေကို ကြည့် နိုင် ပါတယ်။ eth0 က Ethernet interface အတွက် ဖြစ်ပြီး၊ usb0 က USB ပေါ်က Virtual Ethernet အတွက် ဖြစ် ပါတယ်။

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

BBB မှာ သုံးထား တဲ့ Debian Linux ရဲ့ ဗားရှင်း ပေါ်မှုတည်ပြီး ပြင်ဆင် ရတာ ကွာခြား မှ အနည်းငယ် ရှိနိုင် ပါတယ်။ BBB ရဲ့ လက်ရှိ ဗားရှင်း ကို စစ်ချင်ရင် terminal မှာ အောက်က command တွေရိုက်ပြီး ကြည့်နိုင် ပါတယ်။

```
$ more /etc/os-release
$ more /proc/version
$ uname -a
```

২.৭.১ Stretch অবগত Network প্রিংসেপ্স

Debian ভাৰুং: আবৃত্ত stretch মুা static ip লুভিৰতা লাভ কৰি পিতায়। অৱেগিমু দৈনন্দিন:

```
$ connmanctl services
```

শোকৰ কমান্ড যুৎপৰ্য়ি: লকৰণীয় network interface রে service ফাৰ্মেলকৰি গ্ৰহণ কৰি শোকৰ কৰি অৱৰ এথেন্টে_b0d5ccfd9327_cable দেখাই পিতায়। Service কৰি কৰি শোকৰ কৰি অৱেগিমু দৈনন্দিন static ip কৰি অৱেগিমু দৈনন্দিন

```
$ sudo connmanctl config <service> --ipv4 manual <ip_addr> <netmask> <gateway>
> --nameservers <dns_server>
```

শোকৰ কৰি অৱেগিমু দৈনন্দিন: ২.৭ ক কমান্ড দেখাই terminal মুা কৰি শোকৰ কৰি BBB কৰি reboot লুভি শোকৰ কৰি অৱেগিমু দৈনন্দিন রাখুৱা পিপু।

```
1 $ connmanctl services
2 $ sudo connmanctl config ethernet_b0d5ccfd9327_cable --ipv4 manual
    192.168.2.92 255.255.255.0 192.168.2.5 --nameservers 8.8.8.8
```

অৱগত: ২.৭: connmanctl যুৎপৰ্য়ি static ip অৱেগিমু দৈনন্দিন।

Dynamic ip কৰি প্ৰিংসেপ্স: অৰ্থাৎ দেখাই আৱেগিমু দৈনন্দিন অৱেগিমু দৈনন্দিন পিতায়।

```
$ sudo connmanctl config ethernet_b0d5ccfd9327_cable --ipv4 dhcp
```

၃.၅.J Wheezy/Jessie အတွက် Network ပြင်ဆင်ခြင်း

အရင် debian ဗာရှင်း အဟောင်း wheezy နဲ့ Jessie မှာ ဆိုရင် စာရင်း ၃.၆ လို command တွေကို ရုက်ပြီး /etc/network/interfaces ဆိုတဲ့ configuration ဖိုင်ကို ပြုပြင် နိုင် ပါတယ် [Mol14]။

```
1 # cd /etc/network
2 # sudo nano interfaces
```

စာရင်း ၃.၆: interfaces ကိုပြုပြင်ခြင်း။

ပြီးရင် eth0 ရဲ့ primary network interface အပိုင်းကို စာရင်း ၃.၇ မှာ ပြထား သလို ပြင်ဆင် သိမ်းဆည်း ပြီး BBB ကို reboot လုပ်လိုက် ပါမယ်။

```
1 ...
2 # The primary network interface
3 auto eth0
4 iface eth0 inet static
5 address 192.168.2.92
6 netmask 255.255.255.0
7 gateway 192.168.2.5
8 ...
```

စာရင်း ၃.၇: Static IP သတ်မှတ်ခြင်း။

DNS server ထပ်ဖြည့် သတ်မှတ် ချင်ရင်တော့ /etc/resolv.conf မှာ ထပ်ဖြည့် နိုင် ပါတယ် (စာရင်း ၃.၈) [Gio15]။ Google ရဲ့ public DNS ဖြစ်တဲ့ 8.8.8.8 က အဲဒီ ထဲမှာ သတ်မှတ် ပြီးသား ဖြစ်တာကို တွေ့နိုင် ပါတယ်။

```
1 ...
2 nameserver 165.21.83.88
3 nameserver 165.21.100.88
```

စာရင်း ၃.၈: Nameserver များ ထပ်ဖြည့်ခြင်း။

ခုနက /etc/network/interfaces ဖိုင် မှာ အောက် က အတိုင်းထပ် ဖြည့်ရင်လည်း ရ ပါတယ် [Wik17c]။

```
dns-nameservers 8.8.8.8 165.21.83.88
```

၃.၅.၂ Network ဆက်သွယ်မှုကိစစ်ခြင်း

Terminal မှာ ifconfig ဆိုတဲ့ command ကိုရိုက်ကြည့်လိုက်ရင် ဖုန်း၃၂၂၀ မှာလို သတ်မှတ် ထားတဲ့ IP address အတိုင်းဖြစ်နေ တာကို တွေ့နှင့် ပါတယ်။

```
yan@yanhpu: ~
root@beaglebone:~# cd /etc/network
root@beaglebone:/etc/network# sudo nano interfaces
root@beaglebone:/etc/network# ifconfig
eth0      Link encap:Ethernet HWaddr b0:d5:cc:fd:93:27
          inet addr:192.168.2.92 Bcast:192.168.2.255 Mask:255.255.255.0
          inet6 addr: fe80::b2d5:ccff:fed:9327/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:36281 errors:0 dropped:2 overruns:0 frame:0
            TX packets:254 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:3503883 (3.3 MiB) TX bytes:28052 (27.3 KiB)
            Interrupt:40
```

ဖုန်း၃၂၂၀: Static IP ကို စစ်ခြင်း။

BBB မှာ အင်တာနက် ဆက်သွယ်မှု အဆင်ပြေ မပြေ စစ်ကြည့် ဖို့ အတွက် အောက် ကအတိုင်း ping လုပ်ကြည့် နိုင်ပါတယ်။

```
$ ping google.com
```

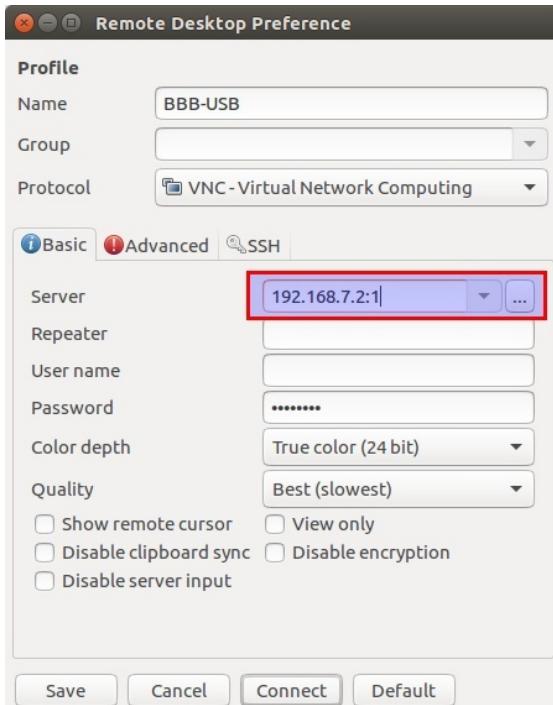
ပြီးတဲ့ အခါ ping လုပ်တာကို ရပိဖို့ အတွက် ctrl+c ကို နှိပ်နိုင် ပါတယ်။ BBB ရဲ့ network ဆက်သွယ်မှု အဆင်ပြေ တာနဲ့ USB ကြိုးကို သုံး စရာ မလိုပဲ 5 V barrel jack ကနေ ပါဝါ ပေးပြီး၊ ခန့်က သတ်မှတ် ခဲ့တဲ့ IP address ကို ဆက်သွယ် အသုံးပြု လိုရ ပါတယ်။

၃.၆ VNC ဖြင့်အသုံးပြုခြင်း

Virtual Network Computing (VNC) က တစ်ခြား ကွန်ပျူးတာ တစ်ခု ရဲ့ desktop ကို အဝေးက နေလုမ်းသုံး လို့ ရတဲ့ စနစ် တစ်ခု ပါ။ အဲဒီ အတွက် အသုံးပြုခဲ့ မယ့် ကွန်ပျူးတာ မှာ vnc server ရှိဖို့ လိုပါတယ်။ BBB မှာ tightvncserver လို့ ခေါ်တဲ့ vnc server တစ်ခါတည့် တပ်ဆင် ပြီးသား ပါ

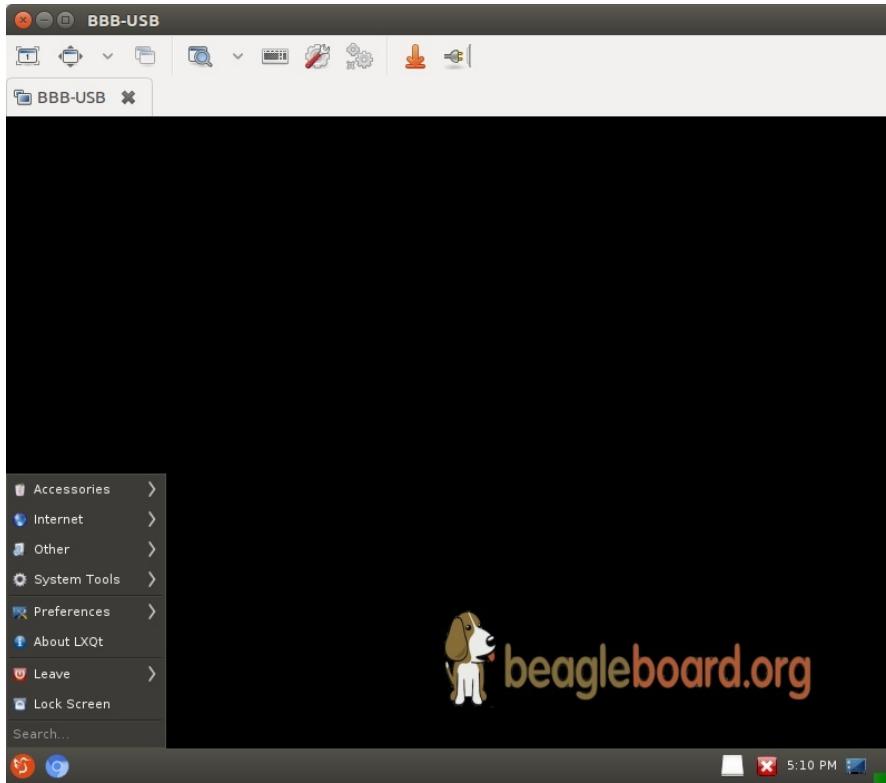
ပါတယ်။ အဲဒီ server ကို စတင် ဖို့ အတွက် SSH နဲ့ ဆက်သွယ်ပြီး tightvncserver ဆိုတဲ့ command ကို ရိုက်ထည့် ပေးနိုင် ပါတယ်။ တော်းသတ်မှတ် ချင်တဲ့ geometry စတာ တွေကို လည်း သတ်မှတ် လို့ ရပါတယ်။

```
$ tightvncserver -geometry 800x600
```



ပုံ ၃.၂၁: VNC ချိတ်ဆက်ခြင်း။

ပထမ ဆုံး အကြိမ် ဆိုရင် server ကို လုမ်းချိတ်ဆက် တဲ့ အခါ သုံးရမယ့် password ကို သတ်မှတ် ခိုင်း ပါလိမ့်မယ်။ အဲဒီ လို့ server ကို ဖွင့်ပြီး သွားရင် ကိုယ့်ရဲ့ client စက်မှာ vnc viewer တစ်ခု သုံးပြီး လုမ်း ချိတ်ဆက် အသုံးပြု လို့ ရပါပြီ။ Platform အမျိုးမျိုး အတွက် RealVNC Viewer စတာ တွေကို သုံးနိုင် ပါတယ်။ Ubuntu Linux မှာ ပါလာ ပြီးသား ဖြစ်တဲ့ Remmina Remote Desktop Client သုံးတဲ့ နမူနာ ကို ပုံ ၃.၂၁ မှာ ပြထား ပါတယ်။ တော်းclient တွေ အတွက် လည်း 192.168.7.2:1 အတိုင်း အတူတူ ဖြစ်ပြီး၊ VNC ချိတ်ဆက် ပြီးတဲ့ အခါ BBB ရဲ့ desktop ကို ကိုးဘုတ်၊ မောက်စ် တွေနဲ့ လုမ်းသုံး လို့ ရပါပြီ (ပုံ ၃.၂၂)။



ပုံ ၃.၂၂: BBB ၏ desktop ၊

စက် ပွင့်လာတိုင်း vnc server ကို အလိုလို စတင် ချင်ရင် တော့ /etc/rc.local ကို အောက်က command သုံးပြီး edit သွားလုပ် ပါမယ်။

```
$ sudo nano /etc/rc.local
```

အဲဒီ မှာ အလို အလျောက် စလုပ် စေခဲ့သူ ပရိုဂရမ်၊ command တွေ ထည့်လို ရပြီး ဒါ နမူနာ အတွက် tightvncserver ကို အောက်က စာရင်း ၃.၉ အတိုင်း ဖြစ်အောင် ထည့်ပေး ပါမယ်။ ပြီးတဲ့ အခါ ctrl+o နိုင်၊ enter နိုင် သိမ်းပြီး၊ ctrl+x နဲ့ ထွက် လိုက် ပါမယ်။

```

1 #!/bin/sh -e
2 #
3 # rc.local
4 #
5 # This script is executed at the end of each multiuser runlevel.
6 # Make sure that the script will "exit 0" on success or any other
```

```

7 # value on error.
8 # In order to enable or disable this script just change the execution
9 # bits.
10 su - debian -c '/usr/bin/tightvncserver :1 -geometry 800x600'
11 exit 0

```

စာရင်း ၃.၉: rc.local ကို ဖြေဖြင့်ခြင်း။

rc.local ဖို့ က executable ဖြစ်ဖို့လို တာမူး အောက်က အတိုင်း သတ်မှတ် ပေးနိုင် ပါတယ်။

```
$ sudo chmod +x /etc/rc.local
```

အဲဒီနောက် အောက်က command ကို ရိုက်ထည့် ပြီး run ကြည့် ပါမယ်။

```
$ sudo /etc/rc.local start
```

စက်ကို reboot လုပ်ပြီး အောက်ပါ အတိုင်း စစ်ကြည့်နိုင် ပါတယ်။ အဲဒီ မှာ Active လို တွေ့ရင် rc.local အဆင်ပြေ ကြောင်း သိနိုင် ပါတယ် (ပုံ ၃.၂၃)။

```
$ systemctl status rc-local.service
```

```
debian@beaglebone:~$ systemctl status rc-local.service
● rc-local.service - /etc/rc.local Compatibility
  Loaded: loaded (/lib/systemd/system/rc-local.service; static; vendor preset:
  Drop-In: /lib/systemd/system/rc-local.service.d
            └─debian.conf
    Active: active (exited) since Thu 2017-08-31 16:53:07 UTC; 24s ago
      Process: 749 ExecStart=/etc/rc.local start (code=exited, status=0/SUCCESS)
        Tasks: 0 (limit: 4915)
       CGroup: /system.slice/rc-local.service
```

ပုံ ၃.၂၃: rc-local.service ၏ status ကို စစ်ခြင်း။

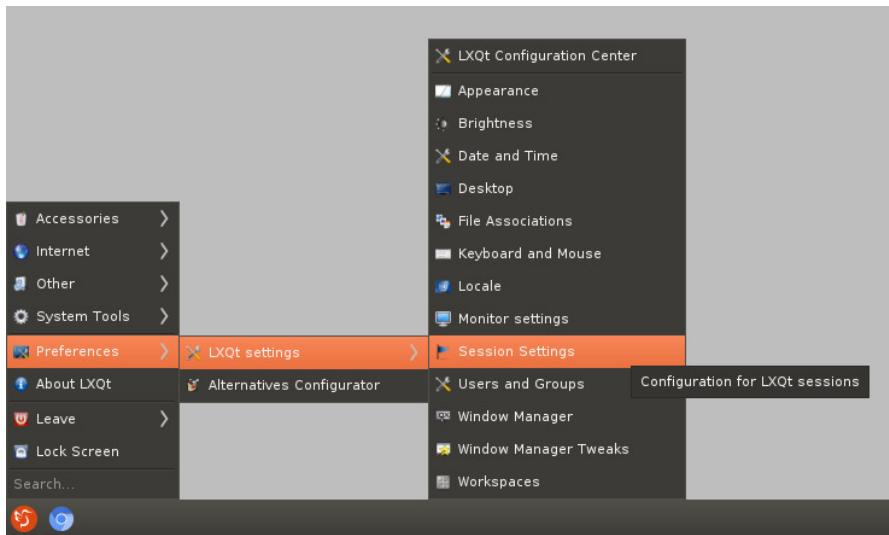
၃.၇ Autostart

Desktop application တွေ အတွက်လည်း စက် စပ်နဲ့ တာနဲ့ အလို အလျောက် စတင် အလုပ်လုပ် ဖို့ သတ်မှတ် ပေးလို ရပါတယ်။ အဲဒီ အတွက် BBB ကို VNC viewer နဲ့ လှမ်းဖွဲ့ ပြီး Start Menu → Pref-

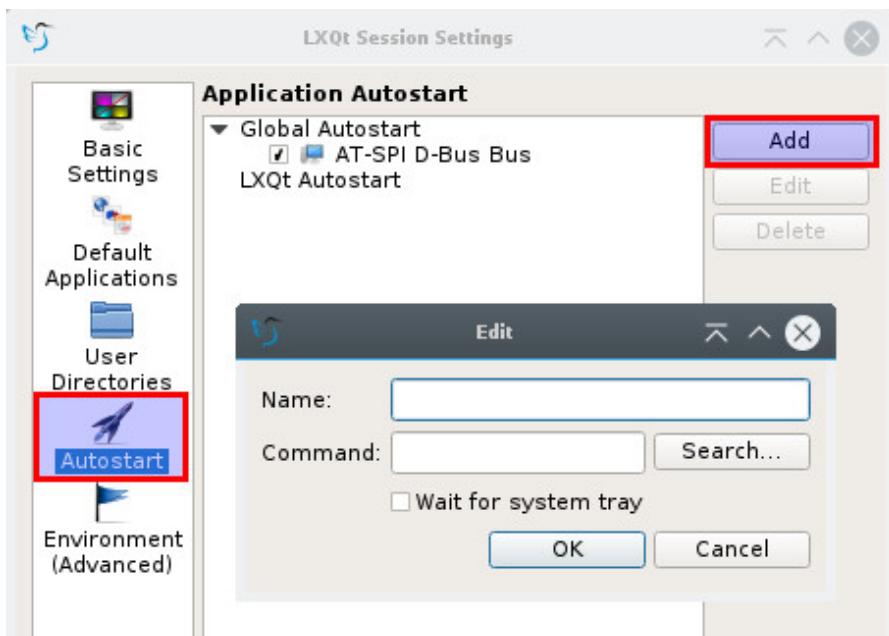
၃.၇. AUTOSTART

၂၁

ferences → LXQt settings → Session Settings ကို ဖွင့်ပါမယ် (ပဲ ၃.၂၄)။



ပဲ ၃.၂၄: LXQt settings ကို ဖွင့်ခြင်း။



ပဲ ၃.၂၅: Autostart ပရိုဂရမ် ကို သတ်မှတ်ခြင်း။

ပြီးတဲ့ အခါ ပဲ ၃.၂၅ မှာ ပြထား သလို Autostart panel မှာ Add ဆလုတ် ကို နိုပ်လိုက် ပြီး အလို

အလျောက် စချင် တဲ့ ပရိုဂရမ် အတွက် နာမည် နဲ့ command ကို သတ်မှတ် နိုင် ပါတယ်။ ဒါ နမူနာ မှာတော့ နာမည် ကို wxcvssimple နဲ့ command ကို /home/debian/code/wxcvssimple/wxcvssimple ဆိုတဲ့ ပရိုဂရမ် ရဲ့ path ကို ထည့်ပေး လိုက် ပါတယ်။ အဲဒါ ဆိုရင် home အခန်းရဲ့ .config/autostart ဆိုတဲ့ အခန်း ထဲမှာ wxcvssimple.desktop ဆိုတဲ့ ဖိုင် တစ်ခု ဖန်တီး သွားမှာ ဖြစ်ပြီး စက် စပ်င့် တာနဲ့ အဲဒီ ပရိုဂရမ် က အလိုလို တခါထဲ ပွင့်နေတာ ကို တွေ့ရ မှာပါ။

Delay ခဏ ခံတာ စတဲ့ တဗြား လုပ်ငန်း တွေနဲ့ ပါ တဲ့ လုပ်ချင်ရင် တော့ ပရိုဂရမ် အစား shell script တစ်ခု ကို ခေါ်ပြီး script ထဲက မှ ပရိုဂရမ် ကို ပြန်ဖွင့် လိုလဲ ရပါတယ်။ အလွယ် တကူ စမ်းကြည့် ဖို့ တဗြား နမူနာ အနေနဲ့ command ရဲ့ ဘေးက Search... ခလုတ် ကို နိုင်ပြီး qterminal စတာ တွေကို ရွေးပြီး လည်း စမ်းကြည့် နိုင် ပါတယ်။

အကယ်၍ အဲဒီ ပရိုဂရမ် က root privileges လိုတယ် ဆိုရင် တော့ wxcvssimple.desktop ကို သွားဖွင့်ပြီး command ရဲ့ ရွေးမှာ sudo ကို ထည့်ပေးနိုင် ပါတယ်။ GUI application ဆိုရင် တော့ sudo အစား gksudo ကို သုံးနိုင် ပါတယ် (ပုံ ၃.၂၆)။

```
nano /home/debian/.config/autostart/wxcvssimple.desktop
```

```
[Desktop Entry]
Exec=gksudo /home/debian/code/wxcvssimple/wxcvssimple
Name=wxcvssimple
Type=Application
Version=1.0
```

ပုံ ၃.၂၆: Autostart ဖိုင်ကို ပြင်ဆင်ခြင်း။

အဲဒီ နောက် application က password မထည့်ပဲ အလို အလျောက် ပွင့် လာအောင် sudoers မှာ သတ်မှတ် ဖို့ လိုပါ သေးတယ်။

```
$ sudo visudo
```

အဲဒီ command နဲ့ sudoers ကို ဖွံ့ဖြိုး၊ password မလိုပဲ ဖွင့်စေ ချင်တဲ့ ပရိုဂရမ် ကို အောက်က အတိုင်း

```
debian ALL=(ALL) NOPASSWD: /home/debian/code/wxcvssimple/wxcvssimple
```

နောက်ဆုံး မှာ သတ်မှတ် ပေးဖို့ လိုပါတယ် (ပုံ ၃.၂၇)။

```
GNU nano 2.7.4          File: /etc/sudoers.tmp          Modified
# Allow members of group sudo to execute any command
%sudo    ALL=(ALL:ALL) ALL
# See sudoers(5) for more information on "#include" directives:
#includeinclude /etc/sudoers.d
debian ALL=(ALL) NOPASSWD: /home/debian/code/wxcvssimple/wxcvssimple
```

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

ပုံ ၃.၂၇: Application တစ်ခု အတွက် password မလိုအောင် sudoers တွင် ပြင်ဆင်ခြင်း။

၃.၉ Locale ပြင်ဆင်ခြင်း

BBB ရဲ့ လက်ရှိ locale ကို သိချင်ရင် date နဲ့ locale -a စတဲ့ command ထွေ သုံးဖြီး ကြည့်နိုင် ပါတယ် (ပုံ ၃.၂၈)။

```
debian@beaglebone:~$ date
Fri Dec 22 04:59:22 UTC 2017
debian@beaglebone:~$ locale -a
C
C.UTF-8
en_US.utf8
POSIX
debian@beaglebone:~$
```

ပုံ ၃.၂၈: Locale ကိုဖြည့်ခြင်း။

ကိုယ် လိုချင် တဲ့ locale ကို ပြုလုပ် ဖို့ /etc/locale.gen ကို စာရင်း ၃.၁၀ အတိုင်း ဖွင့်ပြီး ထုတ်ချင် တဲ့ locale ရဲ့ရှေ့က # သကော်တ ကို ဖြတ်ပြီး uncomment လုပ် နိုင် ပါတယ် (ပုံ ၃.၂၉)။

```
1 $ sudo nano /etc/locale.gen
```

စာရင်း ၃.၁၀: locale.gen ကို edit လုပ်ခြင်း။

```
GNU nano 2.7.4          File: /etc/locale.gen          Modified

# mi_NZ ISO-8859-13
# mi_NZ.UTF-8 UTF-8
# mk_MK ISO-8859-5
# mk_MK.UTF-8 UTF-8
# ml_IN UTF-8
# mn_MN UTF-8
# mni_IN UTF-8
# mr_IN UTF-8
# ms_MY ISO-8859-1
# ms_MY.UTF-8 UTF-8
# mt_MT ISO-8859-3
# mt_MT.UTF-8 UTF-8
my_MM UTF-8
# nan_TW UTF-8
# nan_TW@latin UTF-8
# nb_NO ISO-8859-1
# nb_NO.UTF-8 UTF-8
# nds_DE UTF-8
# nds_NL UTF-8

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

ပုံ ၃.၂၉: Locale ကို ထပ်ဖြည့်ခြင်း။

Edit လုပ်ပြီး တဲ့ အခါ အောက်ပါ command နဲ့ locale တွေကို ထုတ်နိုင် ပါတယ်။

```
$ sudo locale-gen
```

အဲဒီ နောက် /etc/default/locale ကိုဖွင့်ပြီး (စာရင်း ၃.၁၁) အဲဒီ ထဲမှာ သတ်မှတ် ချင်တဲ့ language ကို သတ်မှတ် ပြီး တဲ့ အခါ ctrl+o နှိပ်ပြီး သိမ်းနိုင် ပါတယ် (စာရင်း ၃.၁၂)။ အဲဒီနောက် BBB ကို reboot

လုပ်ပြီး locale ဆိုတဲ့ command ရိုက်ပြီး ပြန်ကြည့် နှင့် ပါတယ်။

```
1 $ sudo nano /etc/default/locale
```

စာရင်း ၃.၁၁: Default locale ကို edit လုပ်ခြင်း။

```
1 LANG = en_US.UTF-8
2 LANGUAGE=en_US
3 LC_ALL = en_US.UTF-8
```

စာရင်း ၃.၁၂: Locale သတ်မှတ်ခြင်း။

Time zone သတ်မှတ် ဖို့ အတွက်တော့ စာရင်း ၃.၁၃ က command တွေကို သုံးပြီး သတ်မှတ်နိုင် ပါတယ်။

```
1 $ timedatectl list-timezones
2 $ sudo timedatectl set-timezone Asia/Yangon
```

စာရင်း ၃.၁၃: Time zone သတ်မှတ်ခြင်း။

၃.၉ C/C++ ပရိုဂရမ်ရေးသားခြင်း

BBB ကို SSH နဲ့ ဆက်ပြီး တဲ့ အခါ ရိုးရှင်း တဲ့ C ပရိုဂရမ် နမူနာ လေး တစ်ခု ရေးကြည့် ပါမယ်။ ဘုတ်ပေါ်က URS3 LED လေးကို ဆယ်ခါ အဖွင့် အပိတ် လုပ်မယ့် ပရိုဂရမ် လေး ကို nano သုံးပြီး ရေးဖို့ SSH terminal မှာ စာရင်း ၃.၁၄ က command ကို ရိုက်ထည့် လိုက်ပါမယ် [Eli13]။

```
1 $ nano egblink.cpp
```

စာရင်း ၃.၁၄: Nano သုံး၍ C ပရိုဂရမ် ရေးသားခြင်း။

ပြီးတဲ့ အခါ စာရင်း ၃.၁၅ မှာ ပြထားတဲ့ နမူနာ C ပရိုဂရမ် ကို ရေးထည့် ပြီး၊ Ctrl+X နဲ့ ပိတ်၊ သိမ်းဖို့ အတွက် Y ကို နှိပ်ပြီး တဲ့ အခါ လက်ရှိ ဖိုင် နာမည် နဲ့ပဲ သိမ်းဖို့ Enter ကို နှိပ်နိုင် ပါတယ်။

```

1 #include<stdio.h>
2 #include<unistd.h>
3 using namespace std;
4 int main(){
5     printf("Starting LED blink.\n");
6     FILE *f=NULL;
7     const char *path="/sys/class/leds/beaglebone:green:usr3/brightness";
8     for(int i=0;i<10;i++){
9         if((f=fopen(path,"r+"))!=NULL){
10             fwrite("1",sizeof(char),1,f);
11             fclose(f);
12         }
13         usleep(1000000);
14         if((f=fopen(path,"r+"))!=NULL){
15             fwrite("0",sizeof(char),1,f);
16             fclose(f);
17         }
18         usleep(1000000);
19     }
20     printf("Ending LED blink.\n");
21     return 0;
22 }
```

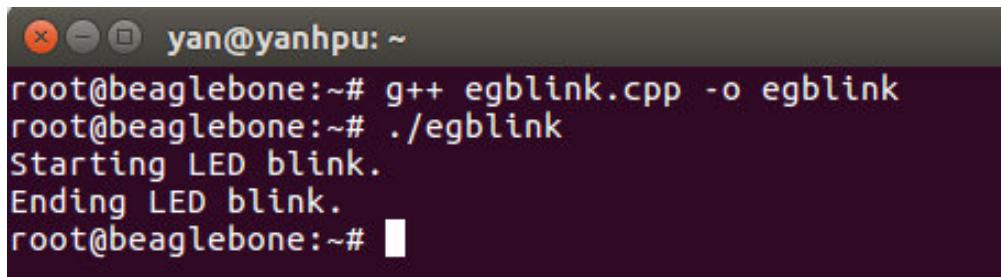
စာရင်း ၃.၁၅: LED အဖွင့် အပိတ် နှမူနာ C ပရိုဂရမဲ့

အဲဒီနောက် ပရိုဂရမဲ့ ကို build လုပ်ပြီး run ဖို့ စာရင်း ၃.၁၆ က command တွက် terminal မှာ ရိုက်ထည့် ပါမယ်။ ပရိုဂရမဲ့ run လိုက်တဲ့ အခါ USR3 LED မိုတ်တုတ် မိုတ်တုတ် ဆယ်ခါ ဖြစ်သွား ပြီး terminal မှာ message တွေပါ ပေါ်လာ တာကို ပုံ ၃.၃၀ မှာ ပြထား သလို တွေ့နိုင် ပါတယ်။

```

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

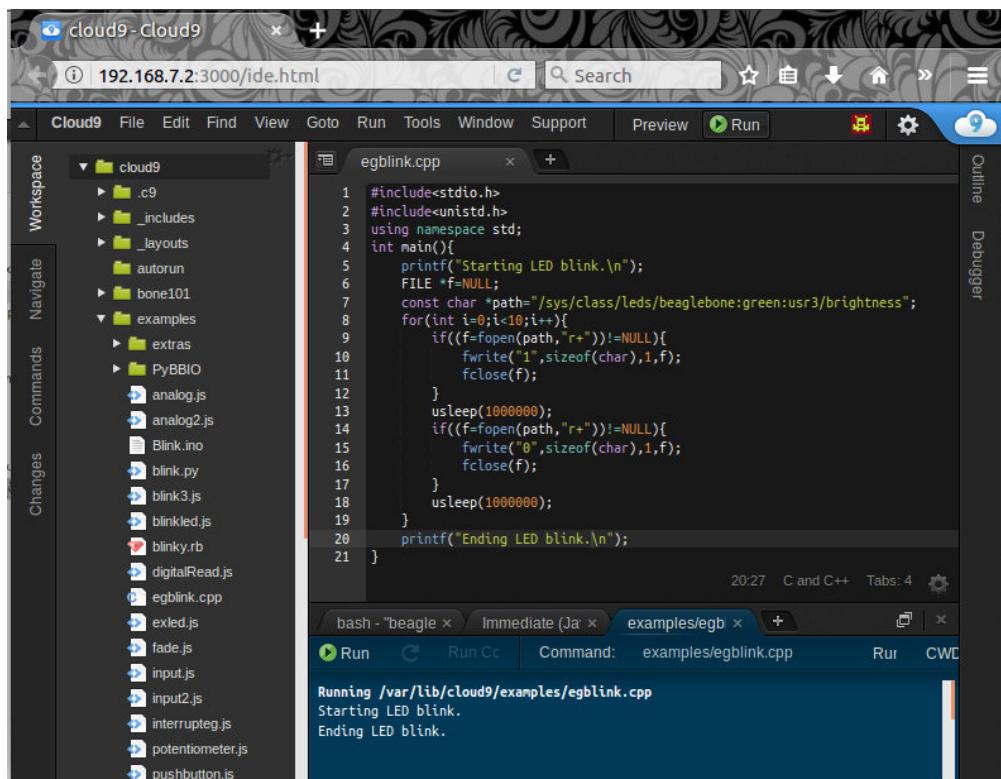
စာရင်း ၃.၁၆: C ပရိုဂရမဲ့ build လုပ်၍ run ခြင်း။



```
yan@yanhpu: ~
root@beaglebone:~# g++ egblink.cpp -o egblink
root@beaglebone:~# ./egblink
Starting LED blink.
Ending LED blink.
root@beaglebone:~#
```

ঃ ২.২০: LED blink C পর্যাগৰণ টি output কি দেখুন।

SSH ৰেখাৰে Cloud9 IDE কি <http://192.168.7.2:3000/ide.html> মোড়ে আছে এবং এটি সহজেই পৰিবহন কৰিব।

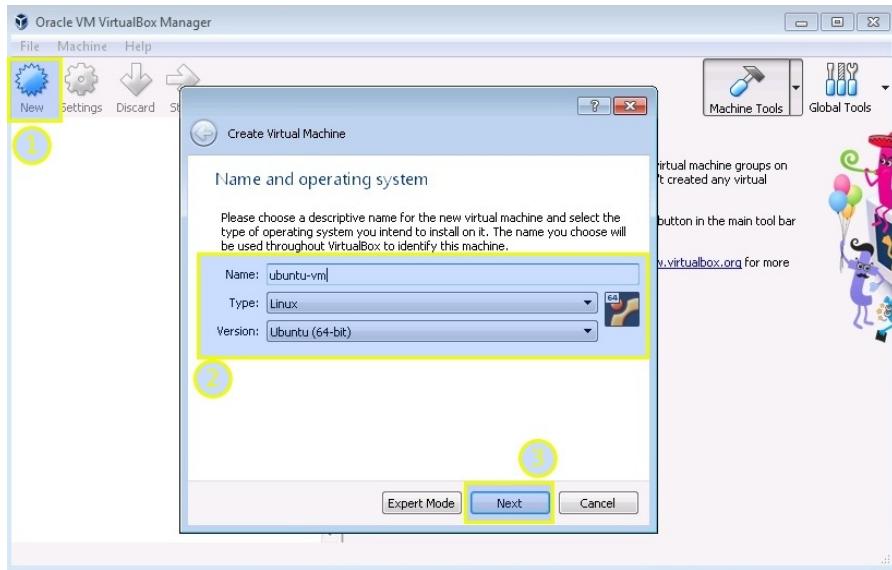


ঃ ২.২১: Cloud9 IDE ৰেখাৰে একটি পৰিবহন কৰিব।

၃.၁၀ Linux Virtual Machine တစ်ခု တပ်ဆင်ခြင်း

BBB က Debian Linux ကို သုံးထားတာ 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 အပိုင်းမှာ ဆွေးနွေး ထား သလို cross compilation tool chain တွေ၊ emulator တွေ သုံးတာ မျိုး အတွက် ပိုမြို့း 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) ကို သုံးပြီး Ubuntu (ubuntu.com) ကို တပ်ဆင် အသုံးပြု ပါမယ်။ Windows hosts အတွက် VirtualBox နောက်ဆုံး ဗားရှင်း ကို download လုပ်ပြီး၊ တပ်ဆင် ပါမယ်။ ပြီးတဲ့ အခါ ပုံ ၃.၂၂၂ 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 ၃.၂၂၂: Virtual machine တစ်ခု ဖန်တီးခြင်း။

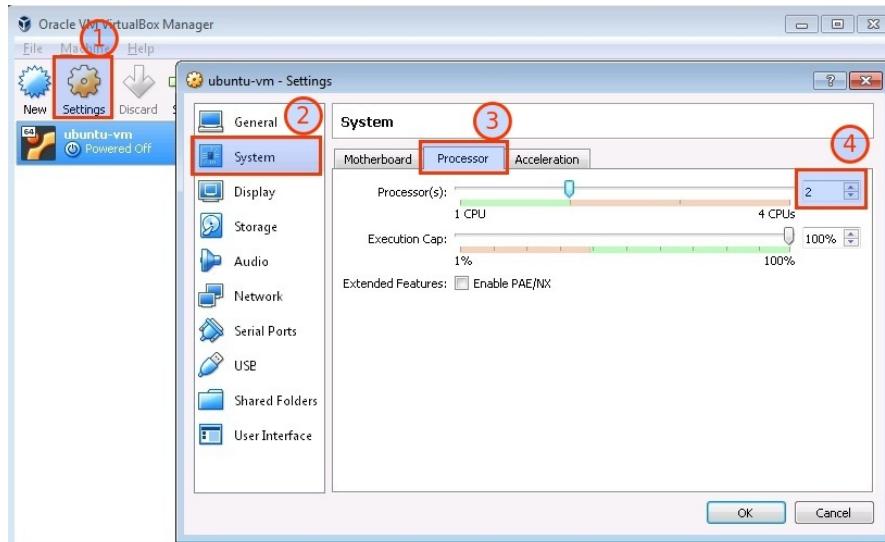
Memory အချယ် အစား ကို လက်ရှိ နောက်ဆုံး Ubuntu 17 အတွက် ရဲ့ အနည်းဆုံး လိုအပ်ချက် ဖြစ်တဲ့ 2 GB ရွေးပြီး၊ Create a virtual hard disk ကို ရွေးပြီး virtual hard disk တစ်ခု ဖန်တီး။

၃.၁၀. LINUX VIRTUAL MACHINE တစ်ခု တပ်ဆင်ခြင်း

၅၉

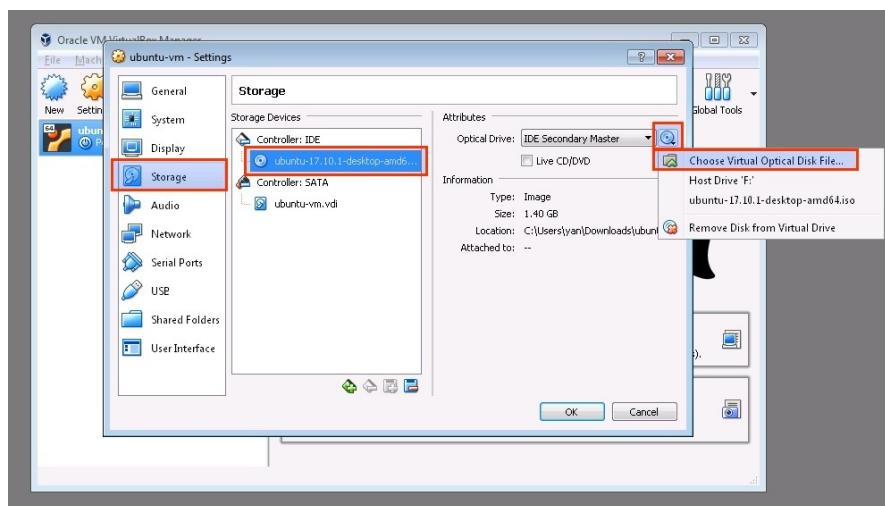
အမျိုးအစား နဲ့ အချယ် (အနည်းဆုံး 25 GB) ကို သင့်တော် သလို ရွေးနိုင် ပါတယ်။

ဖန်တီးလိုက်တဲ့ 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 ၃.၂၃)။



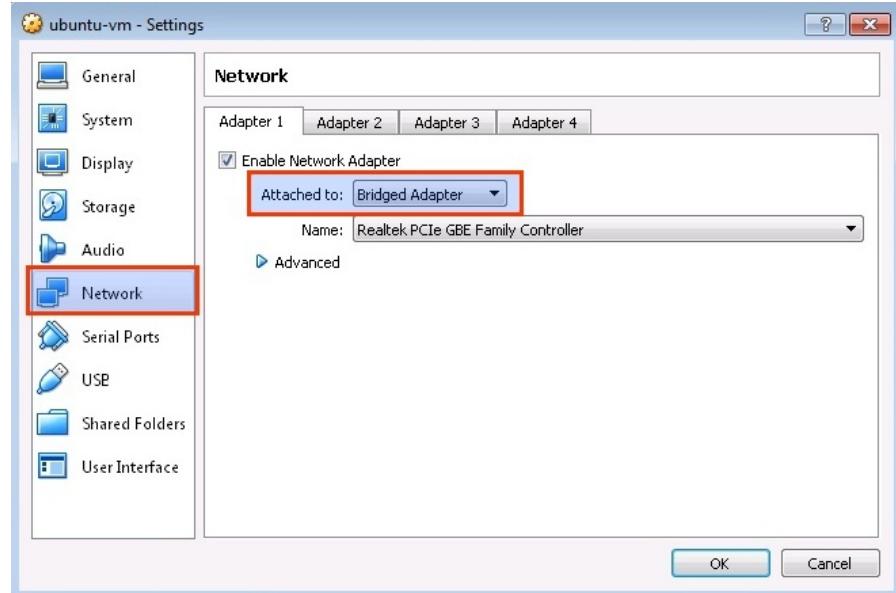
ပုံ ၃.၂၃: Virtual machine တစ်ခု ဖန်တီးခြင်း။

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

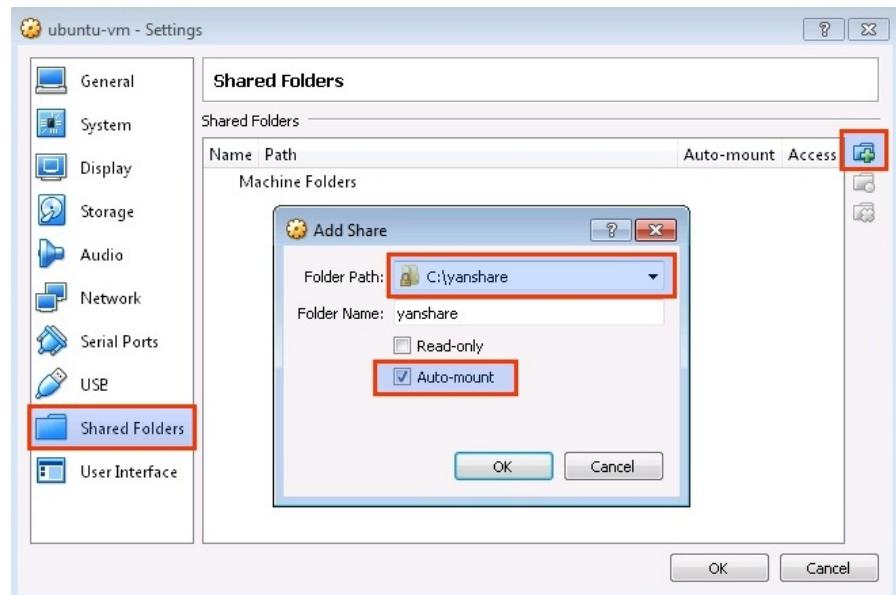


ပုံ ၃.၂၄: တပ်ဆင်မည့် OS ၏ image ကို သတ်မှတ်ခြင်း။

Virtur machine အတွက် network adapter မှာ Bridge Adapter ကို ရွေးပေး မယ်ဆို ရင် host machine တို့ BBB တို့နဲ့ network တစ်ခု ထဲမှာ အတူ ရှိသွား ပါမယ် (ပုံ ၃.၃၅)။

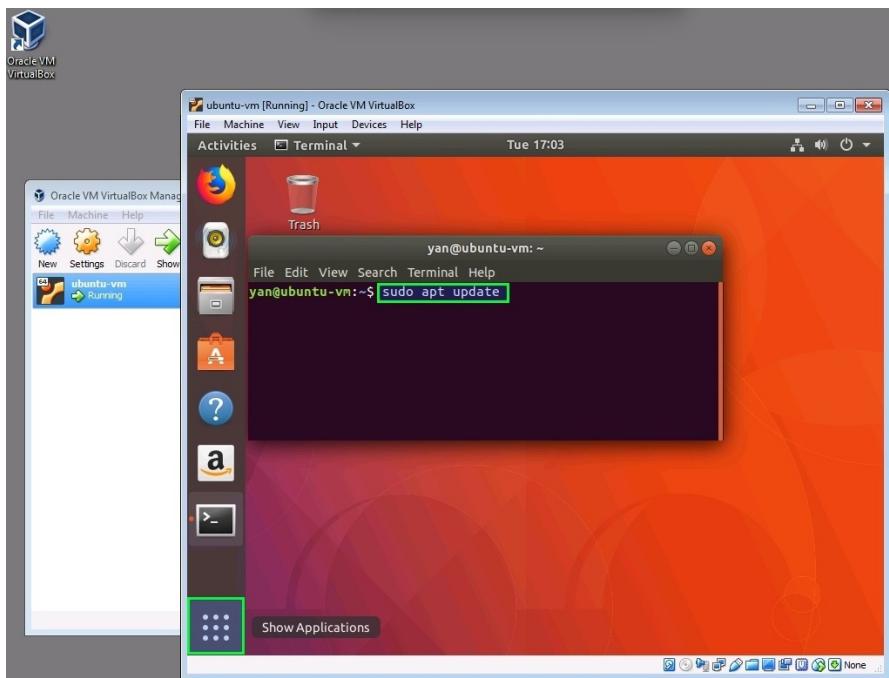


ပုံ ၃.၃၅: Network adapter ကိုသတ်မှတ်ခြင်း။



ပုံ ၃.၃၆: Share folder ကိုသတ်မှတ်ခြင်း။

Host machine နဲ့ ဖိုင်တွေ အပြန် အလှန် share လုပ်ဖို့ share folder တစ်ခု ကို သတ်မှတ် ပါမယ်။ အဲဒီ အတွက် ပုံ ၃.၂၆ 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 နှိပ်၊ Start ကို နှိပ်ပြီး virtual machine ကို စတင် နိုင် ပါတယ်။

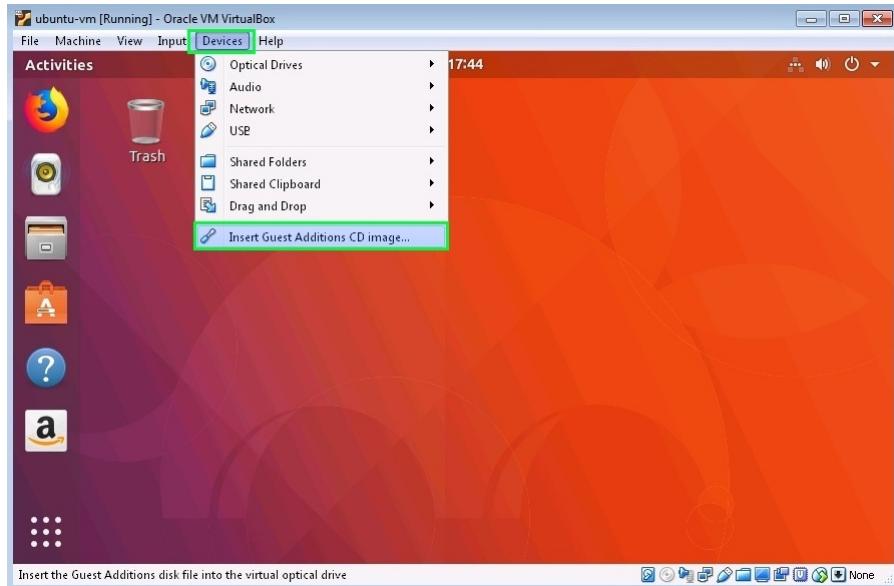


ပုံ ၃.၂၇: လိုအပ်သော software packages များတပ်ဆင်ခြင်း။

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 ၃.၂၇ 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 ၃.၂၈ မှာ ပြထား သလို Virtual machine ဝင်းမှာ Devices menu ကို နှိပ်ပြီး၊ Insert Guest Additions CD image... ကို နှိပ်ပြီး တပ်ဆင် လိုက် ပါမယ်။



ပုံ ၃.၃၈: Guest additions ကို တပ်ဆင်ခြင်း။

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

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

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

၃.၁၁ Cross Compilation Tool Chain

အပိုင်း ၃.၉ က နှမူနာ C++ ပရိုဂရမ် မှာ code ကို ရေးတာရော၊ execute လုပ်တာရော က BeagleBone ပေါ်မှာ ပဲ လုပ်တာပါ။ အဲဒါက ပရိုဂရမ် ကြီးလာ တဲ့အခါ ရေးရာ ပြန်စစ်ရ တာ သိပ် အဆင် မပေါ် ပါဘူး။ ဒါကြောင့် desktop computer ပေါ်မှာ ပဲ ပရိုဂရမ် ကို ရေးပါ မယ်။ compile လုပ်ရင် လည်း code ရေးတဲ့ desktop ပေါ်မှာ တစ်ခါထည်း လုပ်တာ က ပိုပြီး အဆင်ပြော မြန်ဆန် ပါတယ်။ အဲဒါ 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 တွေရဲ့လိုင်ဘရီ တွေကို စက်တခု ထဲမှာ တပ်ဆင် ဖိုအတွက် MultiArch [Wik17b] ကို သုံးပါမယ်။ လက်ရှိ စက်ရဲ့ အာခီ တက်ချာ ကို အောက်က command သုံးပြီး ကြည့်နိုင် ပါတယ်။

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

Floating point နံပါတ် တွေကို တွက်ချက်ဖို hardware အထောက် အပံ့ ပါတဲ့ armhf အာခီ တက်ချာ ကိုထည့်ဖို အတွက် အောက်က ပထမ command သုံး ထည့်နိုင်ပြီး၊ စက်ထဲမှာ ရှိတဲ့ တွေ့ဗ္ဗား အာခီ 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
```

ထည့်သွင်း တပ်ဆင် ပြီးတဲ့ အခါ terminal မှာ အောက်အတိုင်း ရှိက်ထည့်ပြီး စစ်ကြည့်နိုင်ပါတယ် (ပုံ ၃.၂၉)။

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

၆၅

အခန်း ၃. အခြေခံလုပ်ဆောင်မှုများ

```
yan@yan-u16043: /usr/bin
arm-linux-gnueabihf-gcov-tool-5      x86_64-linux-gnu-gcov-tool
arm-linux-gnueabihf-gprof            x86_64-linux-gnu-gcov-tool-5
arm-linux-gnueabihf-ld                x86_64-linux-gnu-gprof
arm-linux-gnueabihf-ld.bfd           x86_64-linux-gnu-ld
arm-linux-gnueabihf-ld.gold          x86_64-linux-gnu-ld.bfd
arm-linux-gnueabihf-nm                x86_64-linux-gnu-ld.gold
arm-linux-gnueabihf-objcopy          x86_64-linux-gnu-nm
arm-linux-gnueabihf-objdump          x86_64-linux-gnu-objcopy
arm-linux-gnueabihf-pkg-config       x86_64-linux-gnu-objdump
arm-linux-gnueabihf-ranlib           x86_64-linux-gnu-pkg-config
arm-linux-gnueabihf-readelf         x86_64-linux-gnu-ranlib
arm-linux-gnueabihf-size             x86_64-linux-gnu-readelf
arm-linux-gnueabihf-strings          x86_64-linux-gnu-size
arm-linux-gnueabihf-strip            x86_64-linux-gnu-strings
cpans5.22-x86_64-linux-gnu          x86_64-linux-gnu-strip
i686-linux-gnu-pkg-config           x86_64-pc-linux-gnu-pkg-config
perl5.22-x86_64-linux-gnu          yan@yan-u16043: /usr/bin$ arm-linux-gnueabihf-g++ --version
arm-linux-gnueabihf-g++ (Ubuntu/Linaro 5.4.0-6ubuntu1-16.04.4) 5.4.0 20160609
Copyright (C) 2015 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@yan-u16043: /usr/bin$
```

ပုံ ၃.၃၉: Cross compiler တပ်ဆင် ထားမှု။

```
yan@yan-u16043: ~/source
yan@yan-u16043:~$ mkdir source
yan@yan-u16043:~$ cd source
yan@yan-u16043:~/source$ more cctest.cpp
#include<stdio.h>
int main()
{
    printf("Cross compiled test!\n");
    return 0;
}
yan@yan-u16043:~/source$ arm-linux-gnueabihf-g++ cctest.cpp -o cctest
yan@yan-u16043:~/source$ ls
cctest  cctest.cpp
yan@yan-u16043:~/source$ ./cctest
bash: ./cctest: cannot execute binary file: Exec format error
yan@yan-u16043:~/source$
```

ပုံ ၃.၄၀: Cross compiler ကိစစ်းကြည့်ခြင်း။

ထည့်လိုက် တဲ့ cross compiler ကို စမ်းကြည့် ဖို့ အတွက် ပုံ ၃.၄၀ မှာ ပြထား တဲ့ အတိုင်း cctest.cpp ဆိုတဲ့ နမူနာ ပရိုကရမဲ့ လေး ရေး ကြည့်ပြီး၊ အောက် က command နဲ့ Ubuntu desktop ကွန်ပျိုတာ ရဲ့ terminal မှာ compile လုပ်ကြည့် လို့ ရတာ ကို တွေ့ရ ပါမယ်။ သူကို run ကြည့် လိုက်တဲ့ အခါ မှာတော့ လက်ရှိ စက်က amd64 အာနီ တက်ချာ မို့လို့ မရ တာကို တွေ့ရ မှာပါ။

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

အဲဒီ ပရိုဂရမ် ကိုပဲ စာရင်း ၃.၁၇ မှာ ပြထားတဲ့ အတိုင်း BeagleBone ပေါ်ကို ကူးထည့်ဖြီး၊ run ကြည့်တဲ့ အခါ စက်နဲ့ အာခီ တက်ချာ ကိုက်ညီ တဲ့ အတွက် အလုပ် လုပ်တာကို တွေ့ရ ပါမယ် (ပုံ ၃.၄၁)။

```
1 $ scp cctest root@192.168.2.92:/root/
2 $ ssh root@192.168.2.92
3 $ ./cctest
```

စာရင်း ၃.၁၇: Cross compile လုပ်ထား သည့် ပရိုဂရမ် ကို run ခြင်း။

```
yan@yan-u16043:~/source$ scp cctest root@192.168.2.92:/root/
Debian GNU/Linux 7

BeagleBoard.org Debian Image 2015-11-12
Support/FAQ: http://elinux.org/Beagleboard:BeagleBoneBlack_Debian
default username:password is [debian:temppwd]

root@192.168.2.92's password:
cctest                                100% 8312      8.1KB/s   00:00

yan@yan-u16043:~/source$ ssh root@192.168.2.92
Debian GNU/Linux 7

BeagleBoard.org Debian Image 2015-11-12
Support/FAQ: http://elinux.org/Beagleboard:BeagleBoneBlack_Debian
default username:password is [debian:temppwd]

root@192.168.2.92's password:
Last login: Thu Sep 14 09:27:52 2017 from yanhpulocal
root@beaglebone:~# ./cctest
Cross compiled test!
root@beaglebone:~#
```

ပုံ ၃.၄၁: Cross compiled test ကို BeagleBone ပေါ်တွင် run ခြင်း။

၃.၁၂ QEMU – Quick EMUlator

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

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

QEMU ထည့်ပြီး တဲ့ အခါ အောက်က အတိုင်း -static အနေနဲ့ compile လုပ်ပြီး တဲ့အခါ desktop ကွန်ပျူးတာ ပေါ်မှာ လည်း run ကြည့်လို ရသွား တာကို တွေ့ရ ပါမယ် (ပုံ ၃.၄၂)။

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

```
yan@yan-u16043:~/source$ arm-linux-gnueabihf-g++ cctest.cpp -o cctest -static
yan@yan-u16043:~/source$ ./cctest
Cross compiled test!
```

ပုံ ၃.၄၂: Emulator ဖြင့် run ခြင်း။

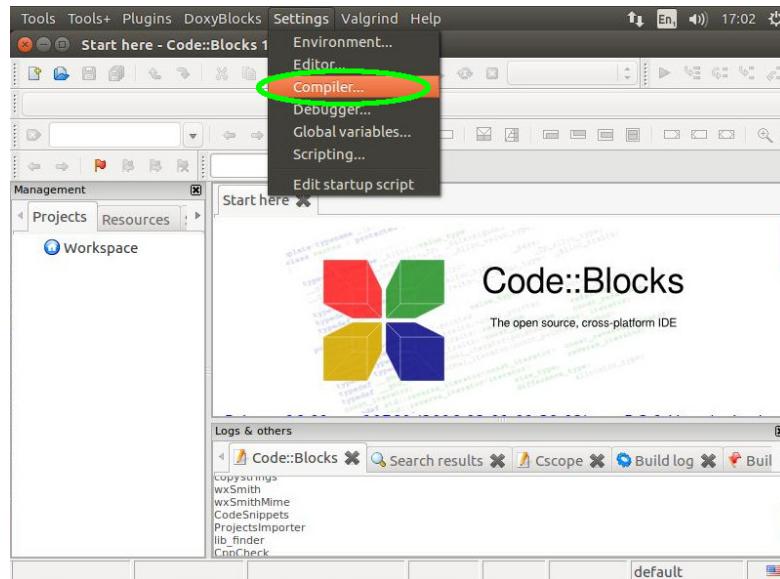
၃.၁၃ Code::Blocks IDE တပ်ဆင်ခြင်း

C/C++ ပရိုဂရမ် တွေ ရေးသားဖို့ အတွက် Code::Blocks ဆိုတဲ့ open source နဲ့ free ဖြစ်တဲ့ IDE ကို သုံး ပါမယ်။ သူက extension တွေလည်းများ၊ စိတ်ကြိုက် configure လည်း လုပ်နိုင်ပြီး တော်တော် လည်း ဆောင်စား 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 ၃.၁၈ အတိုင်း ရိုက်ထည့် နိုင် ပါတယ်။

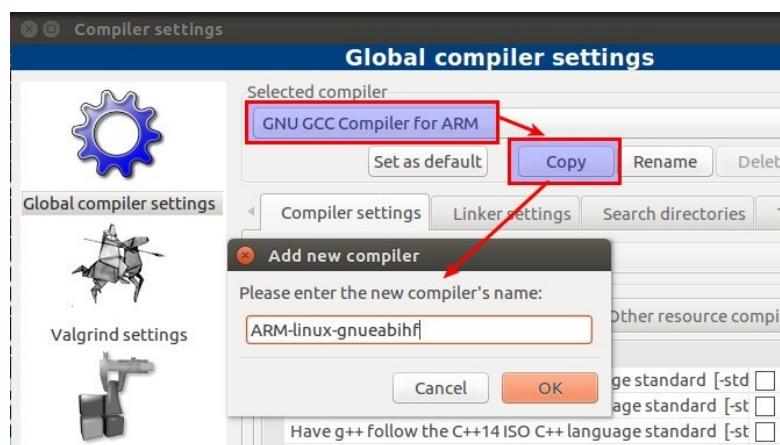
```
1 $ sudo apt-get install build-essential
2 $ sudo apt-get install gdb
3 $ sudo apt-get install libwxgtk3.0
4 $ sudo add-apt-repository ppa:damien-moore/codeblocks-stable
5 $ sudo apt-get update
6 $ sudo apt-get install codeblocks codeblocks-contrib
```

စာရင်း ၃.၁၈: Code Blocks တပ်ဆင်ခြင်း။

Code::Blocks ကို တပ်ဆင် ပြီးတဲ့ အခါ သူကို ဖွင့်လိုက်ပြီး ပုံ ၃.၄၃ မှာ ပြထား သလို Settings menu → Compiler... ကို ဖွင့် လိုက် ပါမယ်။



ပုံ ၃.၄၃: Code::Blocks IDE တပ်ဆင် ပြင်ဆင်မှု။



ပုံ ၃.၄၄: Code::Blocks တွင် Compiler အတွက် ပြင်ဆင်ခြင်း။

Compiler settings ဝင်းရှိုးပေါ်လာ တဲ့အခါ Global Compiler Settings ရဲ့ selected compiler မှာ GNU GCC Compiler for ARM ကို ရွေးလိုက် ပြီး၊ copy ကို နှိပ်လိုက် ပါမယ်။ Compiler အသစ် ရဲ့ နာမည်ကို ARM-linux-gnueabihf အစ ရှိတဲ့ နာမည် တစ်ခု ပေးနိုင် ပါတယ် (ပုံ ၃.၄၄)။

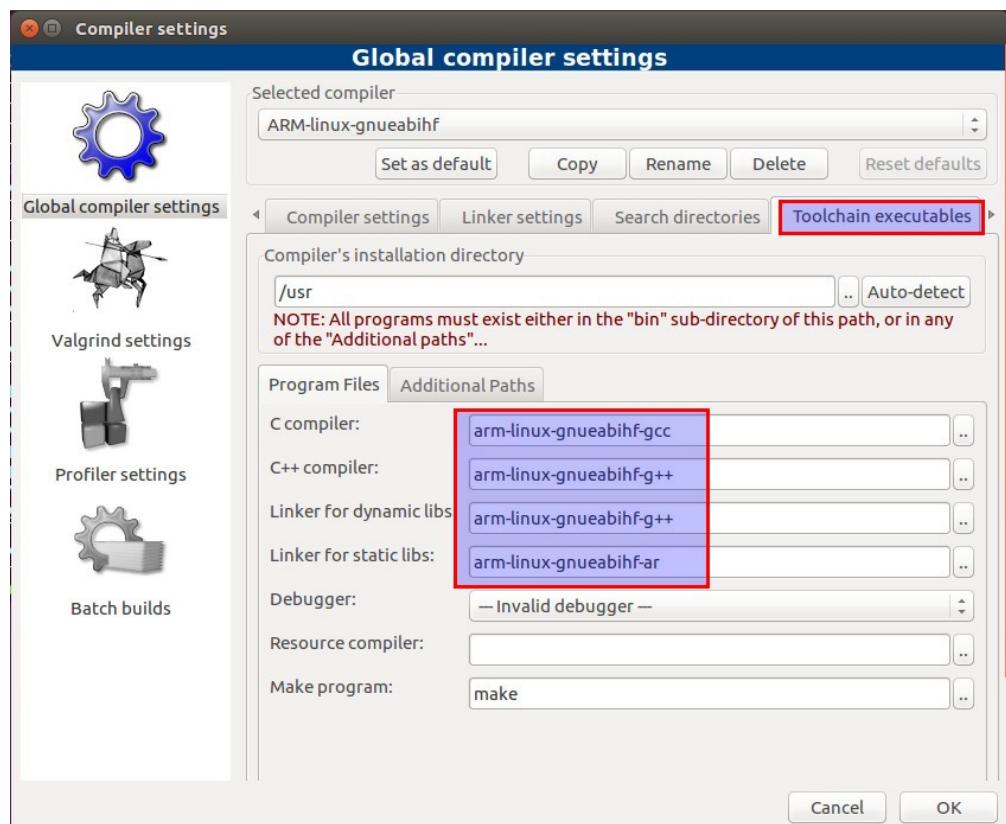
ပြီးတဲ့ အခါ Toolchain executables ဆိုတဲ့ tab ကို သွားပြီး၊ Program Files ဆိုတဲ့ tab မှာ အောက်က စာရင်း ၃.၁၉ နဲ့ ပုံ ၃.၄၅ မှာ ပြထား သလို အသီးသီး ဖြည့်နိုင် ပါတယ်။

```

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

```

စာရင်း ၃.၁၉: Code Blocks တွင် compiler သတ်မှတ်ခြင်း။

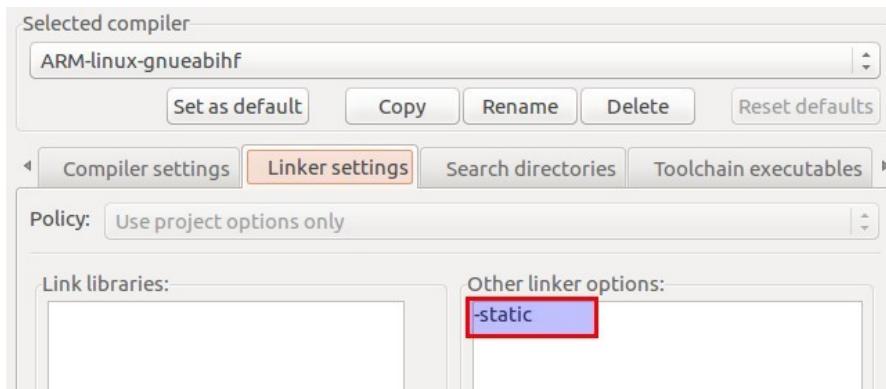


ပုံ ၃.၄၅: Code::Blocks တွင် Compiler program ဖိုင်များ သတ်မှတ်ခြင်း။

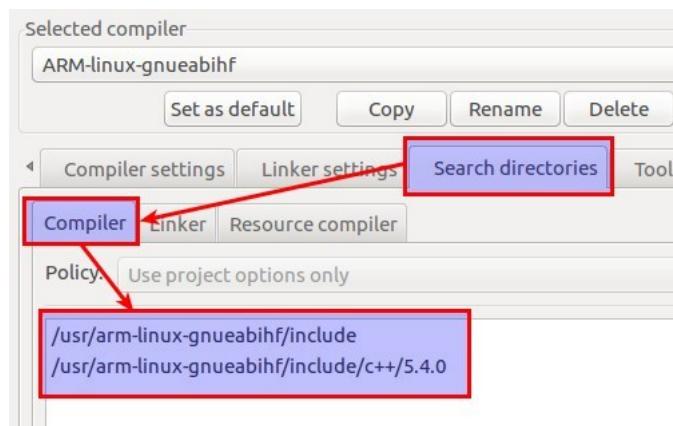
၃.၁၃. CODE::BLOCKS IDE တပ်ဆင်ခြင်း

၆၉

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



ပုံ ၃.၄၆: Code::Blocks တွင် linker option သတ်မှတ်ခြင်း။



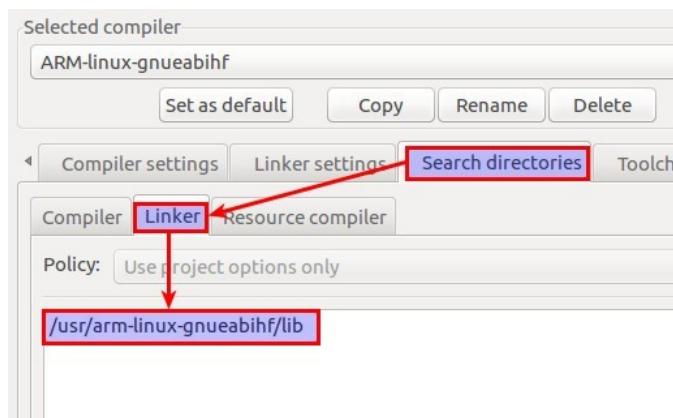
ပုံ ၃.၄၇: Code::Blocks တွင် include paths များ သတ်မှတ်ခြင်း။

နောက် တစ်ခါ ပုံ ၃.၄၇ မှာ ပြထား သလို Search directories ဆိုတဲ့ tab ထဲက compiler မှာ

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

ထိုကို ဖြည့်လိုက် ပါမယ်။

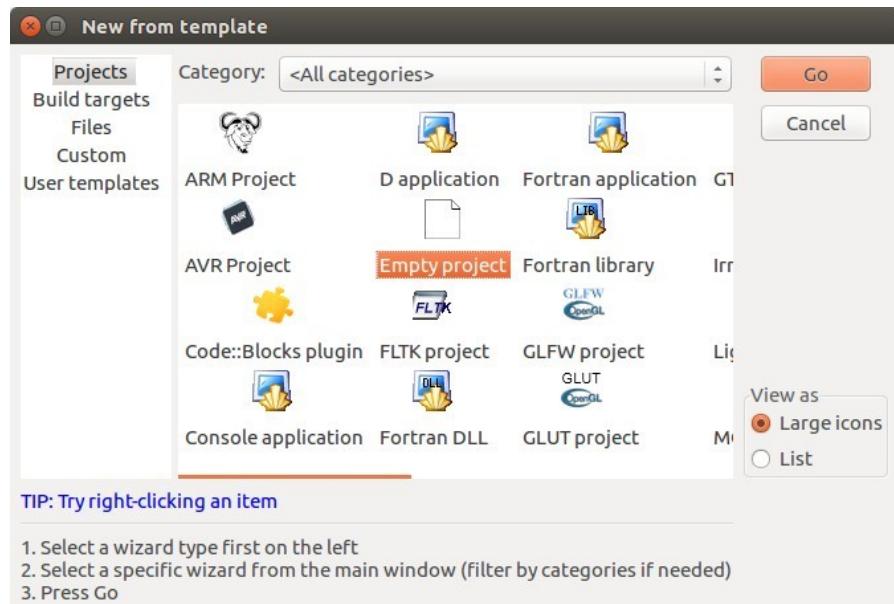
ခုနက လိုပဲ ပုံ ၃.၄၈ မှာ ပြထား သလို Search directories ဆိုတဲ့ tab ထဲက linker မှာ /usr/arm-linux-gnueabihf/lib ကို ဖြည့်လိုက် ပါမယ်။



ပုံ ၃.၄၈: Code::Blocks တွင် lib path သတ်မှတ်ခြင်း။

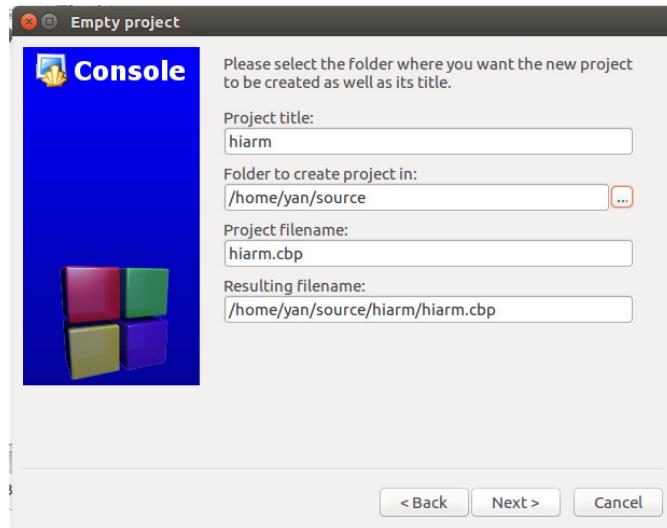
၃.၁၂.၁ Code::Blocks Project နမူနာ

Code::Blocks ထဲမှာ File menu → New → Project... ကို နှိပ်ပြီး project အသစ် တစ်ခု ဖန်တီးလိုက် ပါမယ်။ ပေါ်လာတဲ့ ဝင်းဒီးမှာ Empty Project ကို ရွေးပြီး Go ကို နှိပ်ပြီးရင် Next ကို နှိပ်ပြီးရှေ့ဆက် သွား ပါမယ် (ပုံ ၃.၄၉)။



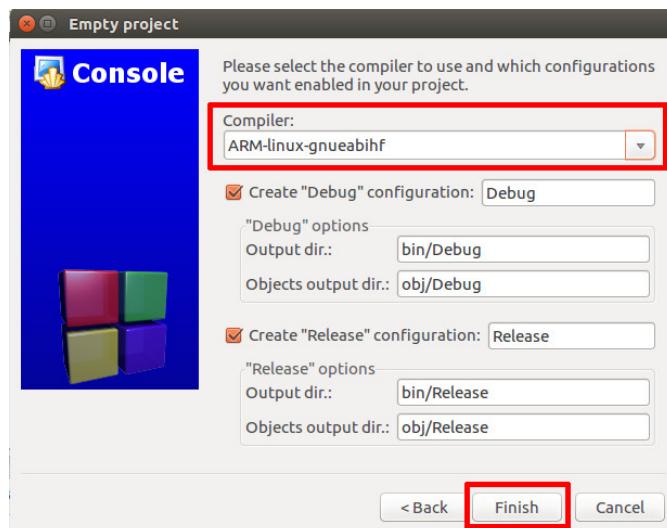
ပုံ ၃.၄၉: Code::Blocks တွင် Project တစ်ခု ဖန်တီးခြင်း။

ပြီးတဲ့ အခါ ပရောဂျက် နာမည်၊ နေရာ တွေကို သတ်မှတ် ပြီးရင် Next ကို ထပ်နိုင် ပါမယ် (ပုံ ၃.၅၀)။



ပုံ ၃.၅၀: Project နာမည် နှင့် နေရာ များ သတ်မှတ် ခြင်း။

နောက် အဆင့် အနေနဲ့ compiler ကို ရွေးတဲ့ အခါ ပုံ ၃.၅၁ မှာ ပြထား သလို ခုနက အပိုင်းမှာ ဖန်တီး ခဲ့တဲ့ ARM-linux-gnueabihf ကို ရွေးလိုက် ပြီး Finish ခလုတ် ကို နှိပ်နိုင် ပါတယ်။

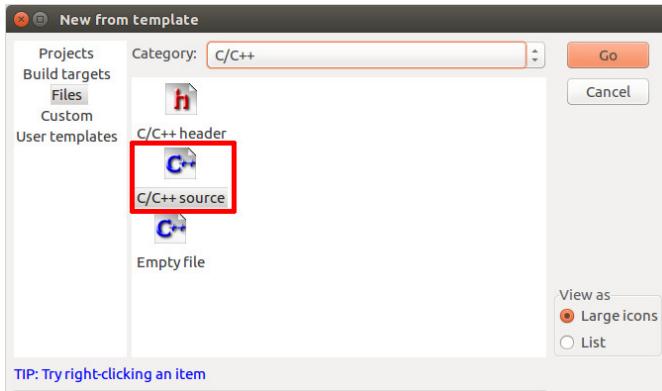


ပုံ ၃.၅၁: Project နာမည် နှင့် နေရာ များ သတ်မှတ် ခြင်း။

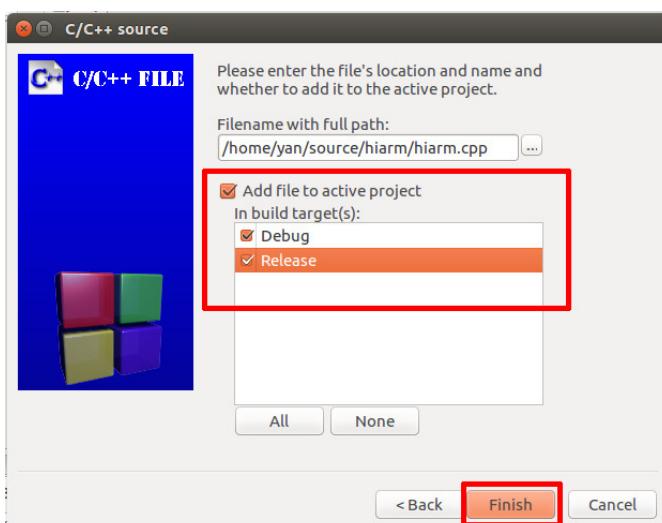
ပရောဂျက် ဖန်တီး လို ရလာ ပြီးတဲ့ အခါ File menu → New → File... ကို နှိပ်ပြီး ဖိုင် အသစ် တစ်ခု

အခန်း ၃. အခြေခံလုပ်ဆောင်မှုများ

ဖန်တီးလိုက်ပြီး ပုံ ၃.၅၂ နဲ့ ပုံ ၃.၅၃ မှာ အဆင့်ဆင့် ပြထားသလို Active project ရဲ့ target တွေထဲ ကို ထည့်လိုက်ပါမယ်။



ပုံ ၃.၅၂: C++ ဖိုင် အသစ် ဖန်တီးခြင်း။



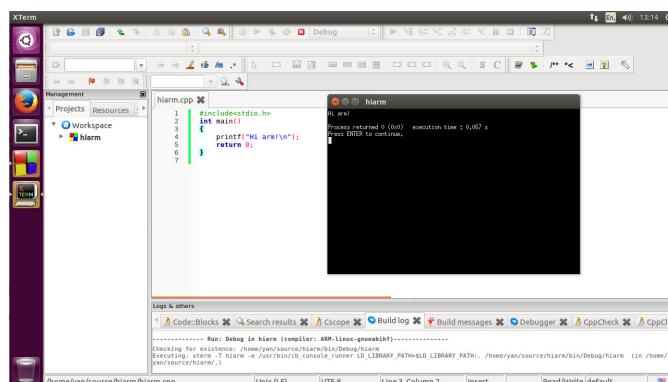
ပုံ ၃.၅၃: Active project နဲ့ target များ တွင်ထည့်ခြင်း။

အဲဒီနောက် မှာ စာရင်း ၃.၂၀ မှာ ပြထားတဲ့ စမ်းသပ် ကြည့်ဖို့ နမူနာ C/C++ ပရိုဂရမ် လေး ရေးလိုက်ပါမယ်။ ပြီးတဲ့ အခါ F9 ခလုပ် နိုပ်ပြီး Build and run လုပ်လိုက်ရင် ပုံ ၃.၅၄ မှာ ပြထားတဲ့ အတိုင်း ပရိုဂရမ် ရဲ့ output ကို တွေ့နိုင်ပါတယ်။ နောက် တဆင့် အနေနဲ့ ပရောဂျက် အခန်းထဲက build လုပ်လိုက် တဲ့ configuration ပေါ်မှုတည် ပြီး bin/Debug ဒါမူ မဟုတ် bin/Release အခန်းထဲက executable application ကို BeagleBone ပေါ်ကူးထည့် ပြီး run ကြည့်တဲ့ အခါ မှာလည်း

ရလဒ် တူတာ ကို တွေ့နှင့် ပါတယ်။

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

ତାରିଖ: ୨୦୧୦: Code::Blocks ଟ୍ୱେଳି ପରିପରାମରଣ କରିବାର ଏକ ପରିଯାଳିତ ପରିକଳ୍ପନା

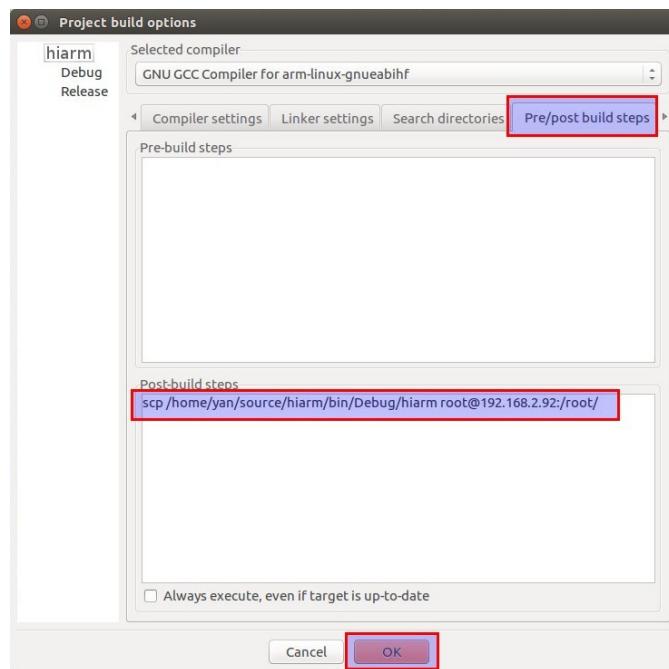


ঃ ২.৭৪: Build and Run প্রিলাউন্ডিংস:

Program ကို Build လုပ်ဖြီး တိုင်း BeagleBone ပေါ်ကို အလို အလျောက် ကူးပေး စေခဲ့ရင် Project menu → Bulid Options... ကို နှိပ်ဖြီး Pre/post build steps tab အောက် က Post-build steps မှာ

```
scp /home/yan/source/hiarm/bin/Debug/hiarm root@192.168.2.92:/root
```

စတဲ့ command တွေကို ဖြည့်ထားနိုင်ပါတယ် (ပဲ ၃:၅၅)။



ပုံ ၃.၅၅: Post-build step များ သတ်မှတ်ခြင်း။

အကိုးအကားများ

- [Bea17] BeagleBoard. BeagleBone - Getting Started. 2017. url: <https://beagleboard.org/getting-started#update>.
- [Eli13] Elinux. Beagleboard: C/C++ Programming. 2013. url: http://elinux.org/Beagleboard:C/C%2B%2B_Programming.
- [Ell17] Duane Ellis. OpenThread Border Router (OTBR) for the BeagleBone Black (BBB) platform. 2017. url: https://openthread.io/guides/border_router/beaglebone_black.
- [Gio15] Rodolfo Giometti. BeagleBone Essentials. 1st ed. Packt Publishing, May 2015. isbn: 978-1-78439-352-6.

- [Mol14] Derek Molloy. Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. Wiley, 2014. isbn: 1118935128. url: <http://www.exploringbeaglebone.com/>.
- [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>.
- [ras17a] raspberrypi.org. Backups. 2017. url: <https://www.raspberrypi.org/documentation/linux/filesystem/backup.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>.
- [Wik17b] Bebian Wiki. Multiarch HowTo. 2017. url: <https://wiki.debian.org/Multiarch/HOWTO>.
- [Wik17c] Debian Wiki. Network Configuration. 2017. url: <https://wiki.debian.org/NetworkConfiguration>.
- [Wik17d] Embedded Linux Wiki. Beagleboard:Expanding File System Partition On A microSD. 2017. url: https://elinux.org/Beagleboard:Expanding_File_System_Partition_On_A_microSD.

အခန်း ၃. အခြေခံလုပ်ဆောင်မှုများ

၇၆

အခန်း ၄

Input/Output များအသုံးပြုခြင်း

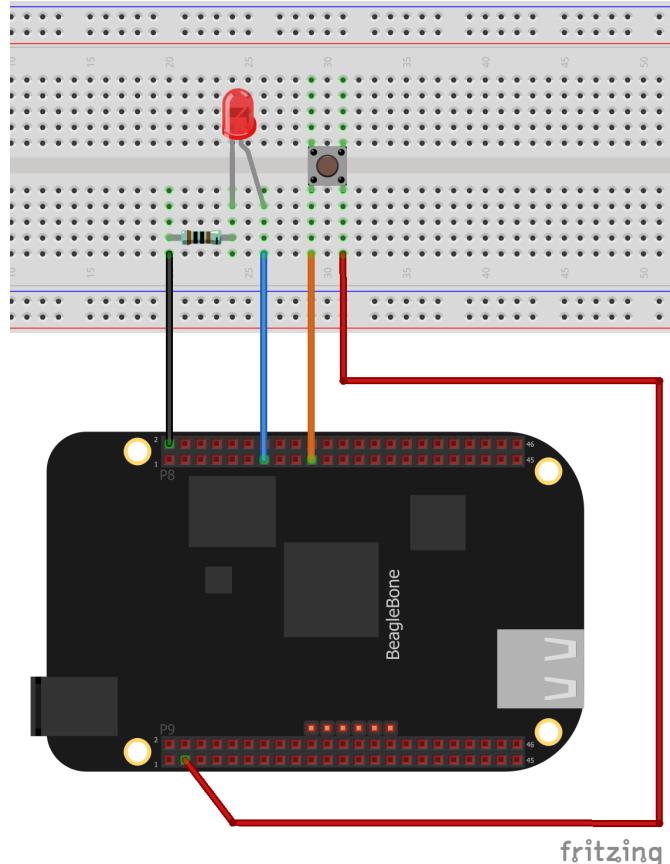
၄.၁ Digital IO

GPIO တွေကို ထိန်းချုပ် တဲ့ နမူနာ အနေနဲ့ အပိုင်း [J.၃](#) က အတိုင်း P8 ခေါင်းရဲ့ pin 13 ကို အသုံးပြု ပါမယ်။ အသုံးပြုဖို့ အဆင့် သင့်ရှိတဲ့ pin တွေကို စစ်ကြည့်ဖို့ အတွက် BeagleBone ရဲ့ /sys/class/gpio အခန်း ထဲကို သွားကြည့် နိုင် ပါတယ်။ ပုံ [၄.၁](#) မှာ BeagleBone ရဲ့ GPIO တွေကို ပြန်ဖော်ပြ ထားပြီး P8_13 က GPIO_23 ဖြစ် တာကို တွေ့ရ မှာ ဖြစ် ပါတယ်။

| P9 | | | P8 | | |
|----------|----|----|------------|--|--|
| DGND | 1 | 2 | DGND | | |
| VDD_3V3 | 3 | 4 | VDD_3V3 | | |
| VDD_5V | 5 | 6 | VDD_5V | | |
| SYS_5V | 7 | 8 | SYS_5V | | |
| PWR_BUT | 9 | 10 | SYS_RESETN | | |
| GPIO_30 | 11 | 12 | GPIO_60 | | |
| GPIO_31 | 13 | 14 | GPIO_50 | | |
| GPIO_48 | 15 | 16 | GPIO_51 | | |
| GPIO_5 | 17 | 18 | GPIO_4 | | |
| I2C2_SCL | 19 | 20 | I2C2_SDA | | |
| GPIO_3 | 21 | 22 | GPIO_2 | | |
| GPIO_49 | 23 | 24 | GPIO_15 | | |
| GPIO_117 | 25 | 26 | GPIO_14 | | |
| GPIO_115 | 27 | 28 | GPIO_113 | | |
| GPIO_111 | 29 | 30 | GPIO_112 | | |
| GPIO_110 | 31 | 32 | VDD_ADC | | |
| AIN4 | 33 | 34 | GNDA_ADC | | |
| AIN6 | 35 | 36 | AIN5 | | |
| AIN2 | 37 | 38 | AIN3 | | |
| AIN0 | 39 | 40 | AIN1 | | |
| GPIO_20 | 41 | 42 | GPIO_7 | | |
| DGND | 43 | 44 | DGND | | |
| DGND | 45 | 46 | DGND | | |

ပုံ [၄.၁](#): BBB ၏ GPIO pin များ။

Digital input နဲ့ output တွေကို စမ်းသပ် အသုံးပြု ကြည့်ဖို့ အတွက် LED မီးလုံး တစ်လုံး နဲ့ push button တစ်ခု ကို အပိုင်း J.၄ က အတိုင်း ပုံ ၄.J မှာ ပြထားသလို ဆက်သွယ် အသုံးပြု ပါမယ်။



ပုံ ၄.J: Digital input/output များ စမ်းသပ်ရန် အပိုင်း J.၄ မှ ဆက်သွယ်ပုံ နမူနာ ကို ပြန်လည်ဖော်ပြခြင်း။

ပုံမှန် အားဖြင့် GPIO တွေက disable ဖြစ်နေ တာမို့ သူတို့ ကို စာရင်း ၄.၁ က အတိုင်း export လုပ်ပြီး enable လုပ်ဖို့ လို ပါတယ်။ ဥပမာ GPIO_23 ကို enable လုပ်ချင် ရင် /sys/class/gpio အခန်း ထဲက export ဆိုတဲ့ ဖိုင်ထဲကို 23 ရေးထည့် လိုက်တာပါ။

```
$ cd /sys/class/gpio
$ echo 23 > export
```

Enable လုပ်ပြီး တဲ့ အခါ /sys/class/gpio အခန်းထဲမှာ enable လုပ်လိုက်တဲ့ gpio ရဲ့ နာမည်နဲ့ အခန်း

တစ်ခု ပေါ်လာတာ တွေ့နှင့် ပါတယ်။ ပြီးတဲ့ အခါ GPIO တွေရဲ့ direction ကို အဝင် သိမဟုတ် အထွက် စာဖြင့် သတ်မှတ် နိုင်ပါ တယ်။ အဲဒီ အတွက် သူရဲ့ အခန်းထဲက direction ဆိုတဲ့ ဖိုင်ထဲကို in သိမဟုတ် out ကို ရေးနိုင် ပါတယ်။ ဥပမာ GPIO_23 ကို output သတ်မှတ် မယ် ဆိုရင် gpio23 အခန်းထဲက direction ဆိုတဲ့ ဖိုင်မှာ out ကို ရေးနိုင် ပါတယ်။

```
$ cd gpio23
$ echo out > direction
```

အဲဒီ နောက်မှာ အဝင် ဆိုရင် တန်ဖိုးကို ဖတ်၊ အထွက် ဆိုရင် တန်ဖိုး တွေကို 0 သိမဟုတ် 1 ထဲတဲ့ ပေးနိုင် ပါတယ်။ စာရင်း ၄.၁ က command တွေကို ပုံ ၄.၃ က အတိုင်း ထည့်ပြီး စမ်းသပ် ကြည့်တဲ့ အခါ LED မီးလုံး အဖွင့် အပိတ် ဖြစ်သွား တာကို တွေ့နှင့် ပါတယ်။

```
1 $ cd /sys/class/gpio
2 $ ls
3 $ echo 23 > export
4 $ ls
5 $ cd gpio23
6 $ ls
7 $ more direction
8 $ echo out > direction
9 $ more direction
10 $ more value
11 $ echo 1 > value
12 $ echo 0 > value
```

စာရင်း ၄.၁: GPIO များကို ကြည့်ခြင်း။

```

root@beaglebone:~# cd /sys/class/gpio
root@beaglebone:/sys/class/gpio# ls
export gpiochip0 gpiochip32 gpiochip64 gpiochip96 unexport
root@beaglebone:/sys/class/gpio# echo 23 > export
root@beaglebone:/sys/class/gpio# ls
export gpio23 gpiochip0 gpiochip32 gpiochip64 gpiochip96 unexport
root@beaglebone:/sys/class/gpio# cd gpio23
root@beaglebone:/sys/class/gpio/gpio23# ls
active_low direction edge power subsystem uevent value
root@beaglebone:/sys/class/gpio/gpio23# more direction
in
root@beaglebone:/sys/class/gpio/gpio23# echo out > direction
root@beaglebone:/sys/class/gpio/gpio23# more direction
out
root@beaglebone:/sys/class/gpio/gpio23# more value
0
root@beaglebone:/sys/class/gpio/gpio23# echo 1 > value
root@beaglebone:/sys/class/gpio/gpio23# echo 0 > value

```

ပုံ ၄.၃: GPIO pin ကို Linux shell မှ ကြည့်ရှု ထိန်းချုပ်ခြင်း။

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

```
echo 23 > unexport
```

၄.၁.၁ C++ ဖြော်သုံးခြင်း

C++ နဲ့ GPIO တွေကို ထိန်းချုပ် ဖို့ အတွက် ရိုးရှင်းတဲ့ နမူနာ class လေး တစ်ခု ကို [ce_io.h](#) (စာရင်း ၄.၂) နဲ့ [ce_io.cpp](#) (စာရင်း ၄.၃) မှာ ပြထား ပါတယ်။ စာရင်း ၄.၁ 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.sg
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 #ifndef CE_IO_H
8 #define CE_IO_H
9

```

```

10 #include <iostream>
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 #endif

```

စာရင်း၏ ce_io.h

```

1 //File: ce_io.cpp
2 //Description: Controlling GPIO pins
3 //WebSite: http://cool-emerald.blogspot.sg
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 #include "ce_io.h"

```

```
8 template <typename T>
9 string CE_IO::ToString(T a)
10 {
11     ostringstream ss;
12     ss << a;
13     return ss.str();
14 }
15
16 CE_IO::CE_IO()
17 {
18
19 }
20
21 CE_IO::CE_IO(int gpono,iodir mode)
22 {
23     Begin(gpono,mode);
24 }
25
26 int CE_IO::Begin(int gpono,iodir mode)
27 {
28     gpioNumber=gpono;
29     fpath = IOPATH;
30     fpath+="gpio"+ToString(gpioNumber)+"/value";
31     if(Export()<0) return -1;
32     if(SetDirection(mode)<0) return -1;
33     return 0;
34 }
35
36 int CE_IO::Export()
37 {
38     ofstream wfile;
39     int r=-1;
40     string epath=IOPATH;
41     epath+="export";
42     wfile.open(epath.c_str());
43     if (wfile.is_open()) {
```

```
44     wfile << gpioNumber;
45     r=0;
46 }
47 wfile.close();
48 return r;
49 }

50

51 int CE_IO::SetDirection(iodir mode)
52 {
53     ofstream wfile;
54     int r=-1;
55     string dirpath=IOPATH;
56     dirpath+="gpio"+ToString(gpioNumber)+"/direction";
57     wfile.open(dirpath.c_str());
58     string modestr[2]={"in","out"};
59     if (wfile.is_open()) {
60         wfile << modestr[mode];
61         r=0;
62     }
63     wfile.close();
64     return r;
65 }

66

67

68 CE_IO::~CE_IO()
69 {
70     ofstream wfile;
71     string epath=IOPATH;
72     epath+="unexport";
73     //unexport the gpio
74     wfile.open(epath.c_str());
75     if (wfile.is_open()) {wfile << gpioNumber;}
76     wfile.close();
77 }
78

79 digitalval CE_IO::Read()
```

```

80 {
81     ifstream rfile;
82     string str;
83     digitalval rdv=LOW;
84     rfile.open(fpath.c_str());
85     if (rfile.is_open()) {
86         getline(rfile,str);
87     }
88     else {
89         str="";
90     }
91     rfile.close();
92     if (str=="1") rdv=HIGH;
93     return rdv;
94 }
95
96 int CE_IO::Write(digitalval b)
97 {
98     ofstream wfile;
99     int r=-1;
100    wfile.open(fpath.c_str());
101    if (wfile.is_open()) {
102        wfile << b;
103        r=0;
104    }
105    wfile.close();
106    return r;
107 }
```

စာရင်း ၄.၃: ce_io.cpp

အဲဒီ class ကို အသုံးပြုတဲ့ C++ ပရိုဂရမ် နမူနာ `bbgpio.cpp` ကို စာရင်း ၄.၄ မှာ ပြထား ပါတယ်။ ပုံ ၄.၂ က ဆက်သွယ်မှု ပုံစံ အတိုင်း ၁ စက္ကန် တစ်ခါ P8_19 ရဲ့ အဝင်ကို ဖတ်ပေး ပြီး၊ P8_13 ရဲ့ အထွက် LED မီးလုံးကို မိုတ်တူတ် မိုတ်တူတ် ပြုလုပ် ပေးပါမယ်။

```
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 dout(23, OUTPUT);
9     CE_IO din(22, INPUT);
10
11    for(int i=0;i<5;i++){
12        printf("i=%d\n",i);
13
14        //Read input
15        printf("Input= %d \n",din.Read());
16
17        //Write output
18        dout.Write(HIGH);
19        printf("Output = 1");
20        usleep(500000);
21        dout.Write(LOW);
22        printf(" > 0\n");
23        usleep(500000);
24    }
25    return 0;
26 }
```

စာရင်း ၄.၄: bbgpio.cpp

ပရိုဂရမ် ကို စာရင်း ၄.၅ အတိုင်း Build လုပ်ပြီး Run နိုင် ပါတယ်။ ပရိုဂရမ် ခဲ့ output ကို ပုံ ၄.၆ မှာ တွေ့နိုင် ပါတယ်။

```

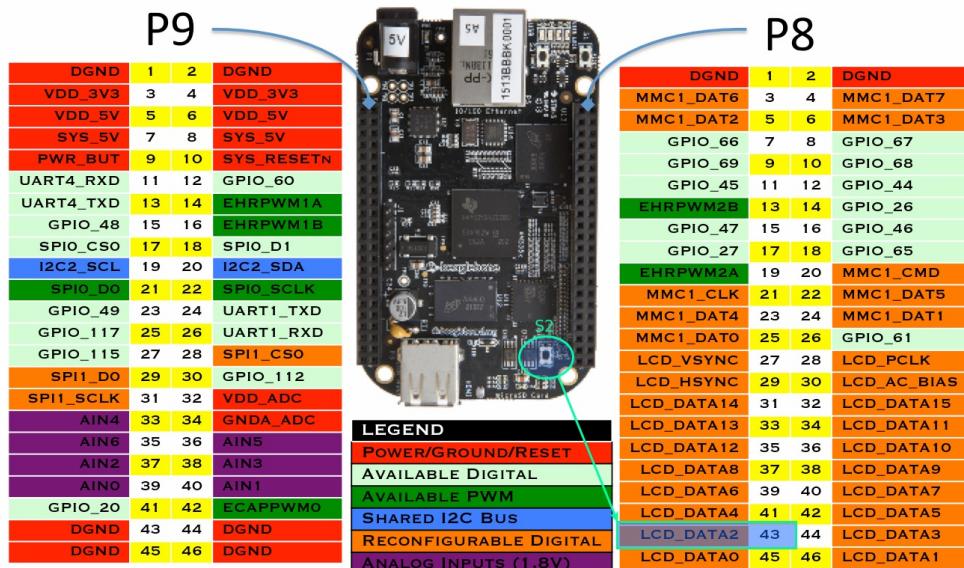
1 $ g++ bbgpio.cpp ce_io.cpp -o bbgpio
2 $ sudo ./bbgpio
```

စာရင်း ၄.၅: GPIO များကို C++ ဖြင့် ထိန်းချုပ် သည့် ပရိုဂရမ် ကို run ခြင်း။

```
debian@beaglebone:~/bbgpio$ g++ bbgpio.cpp ce_io.cpp -o bbgpio
debian@beaglebone:~/bbgpio$ sudo ./bbgpio
i=0
Input= 0
Output = 1 > 0
i=1
Input= 0
Output = 1 > 0
i=2
Input= 1
Output = 1 > 0
i=3
Input= 1
Output = 1 > 0
i=4
Input= 0
Output = 1 > 0
debian@beaglebone:~/bbgpio$
```

ပုံ ၄.၄: GPIO များကို ထိန်းချုပ်သည့် C++ ပရီဂရမ်။

BBB မှာ ရှိတဲ့ pin တွေက လုပ်ငန်း တစ်ခု မက လုပ်နိုင်ကြ ဖြီးပုံ ၄.၅ မှာ ပြထားတဲ့ လိမ္မာ်ရောင် pin တွေက reconfigurable digital pin တွေ ဖြစ်ကြ ပါတယ်။ ဥုမှာ BBB မှာ ပါတဲ့ S2 push button လေးက P8_43 နဲ့ ဆက်ထား ပေမယ့် သူကို digital input အနေနဲ့ ဒါ အတိုင်း တန်းသုံးလို့ မရ ပါဘူး။ အဲဒီ pin က LCD_DATA2 အနေနဲ့ သုံးနေတဲ့ အတွက် ကြောင့်ပါ။



ပုံ ၄.၅: လိမ္မာ်ရောင် Reconfigurable digital pin များ နှင့် S2 switch ၏ header pin ။

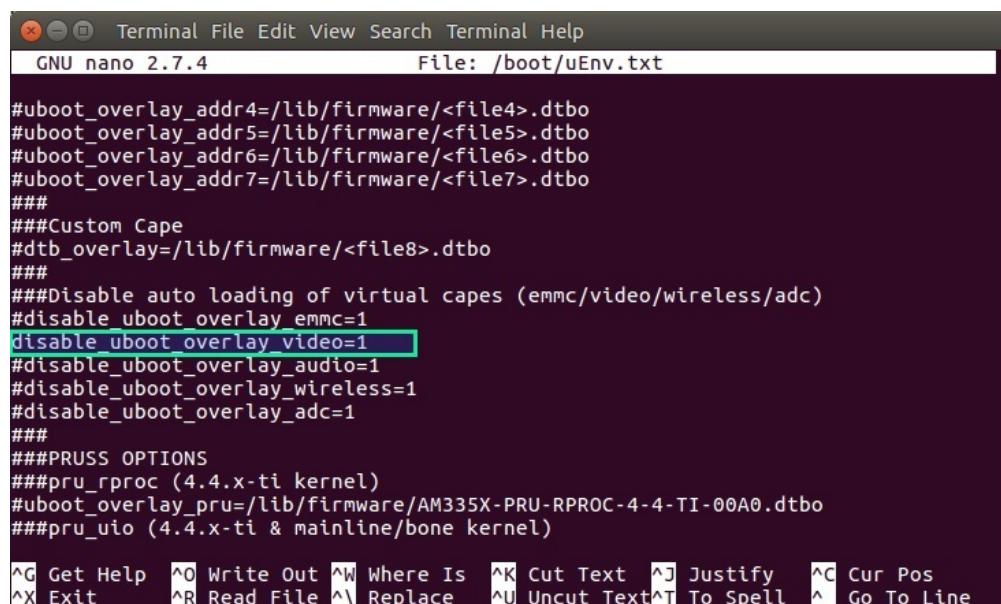
ဒါကြား P8_43 ပဲ GPIO_72 ကို သုံးချင်ရင် /boot/uEnv.txt ကို အောက်က အတိုင်း ဖွင့်ပြီး edit လုပ်ဖို့ လိပါတယ်။ အရင် firmware ဟူငြင်း အဟောင်းတွေ အတွက် တော့ bone_capemgr နဲ့ နောက်ဆက်တဲ့ A အပိုင်း ၁.၂ မှာ ဖော်ပြ ထားသလို ပြင်နိုင် ပါတယ်။

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

အဲဒီ ဖိုင် ထဲမှာ

```
disable_uboot_overlay_video=1
```

ဆိုတဲ့ စာကြားငဲ့ ကို ပဲ ၄.၆ မှာ ပြထားသလို uncomment လုပ်ပြီး ထည့်လိုက် ပါမယ်။ ပြီးတဲ့ အခါ ctrl+o နဲ့ သိမ်း၊ ctrl+x နဲ့ ထွက်နိုင် ပါတယ်။



```
GNU nano 2.7.4          File: /boot/uEnv.txt

#uboot_overlay_addr4=/lib/firmware/<file4>.dtbo
#uboot_overlay_addr5=/lib/firmware/<file5>.dtbo
#uboot_overlay_addr6=/lib/firmware/<file6>.dtbo
#uboot_overlay_addr7=/lib/firmware/<file7>.dtbo
###  

###Custom Cape
#dtb_overlay=/lib/firmware/<file8>.dtbo
###  

###Disable auto loading of virtual capes (emmc/video/wireless/adc)
#disable_uboot_overlay_emmc=1
disable_uboot_overlay_video=1  

#disable_uboot_overlay_audio=1
#disable_uboot_overlay_wireless=1
#disable_uboot_overlay_adc=1
###  

###PRUSS OPTIONS
###pru_rproc (4.4.x-ti kernel)
#uboot_overlay_pru=/lib/firmware/AM335X-PRU-RPROC-4-4-TI-00A0.dtbo
###pru_uio (4.4.x-ti & mainline/bone kernel)

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

ပဲ ၄.၆: Video overlay ကို disable လုပ်ခြင်း။

အဲဒီ နောက် BBB ကို reboot လုပ်ပြီး တဲ့ အခါ LCD_DATA pin တွေကို မသုံးတော့ တဲ့ အတွက် GPIO_72 ကို S2 အတွက် input အနေနဲ့ သုံးလို့ ရသွား တာကို တွေ့နိုင် ပါတယ်။ အဲဒီ နည်းတူပဲ MMC pin တွေကို လည်း reconfigure ပြန်လုပ် နိုင် ပါတယ်။

၄.၁.၂ 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။

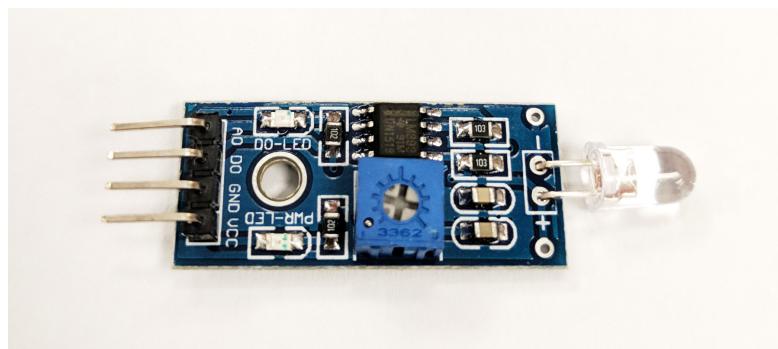
အဲဒီ module က 3.3 V နဲ့ ရော 5 V နဲ့ပါ သုံးလို့ ရပါတယ်။ သူရဲ့ pin လေးခုက အောက်ပါ အတိုင်း ဖြစ်ပါတယ်။

Vcc: Power supply pin 3.3 V to 5 V.

Gnd: Ground.

Do: Digital output.

Ao: Analog output.



ပုံ ၄.၇: 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 ၄.၇ 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 digital output ကိုပဲ သုံးမှာ ဖြစ်တဲ့ အတွက် သူကို BBB ရဲ့ P9_15 (GPIO_48) နဲ့ ဆက်သွယ် လိုက် ပါမယ်။

၄.၁.၃ 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 သုံးခဲ့ပါပြီး၊ 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 ၄.၈ 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 မီးချောင်း စတဲ့ လျှပ်စစ် ပစ္စည်း တွေ အတွက် ခလုတ် အနေနဲ့ ထိန်းချုပ် နိုင် ပါတယ်။



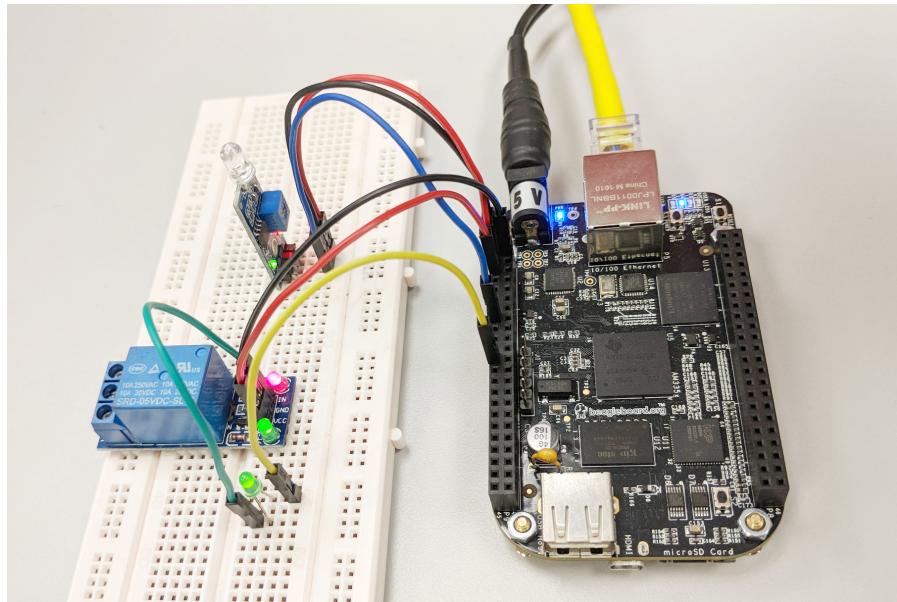
ပုံ ၄.၈: Relay

Relay ကို GND နဲ့ 5 V ပါဝါ ပေးပြီး တဲ့အခါ BBB ရဲ့ P9_23 (GPIO_49) နဲ့ ဆက်သွယ် လိုက် ပါမယ်။ သုံးထား တဲ့ relay က 5 V ဖြစ်နေတဲ့ အတွက် ဆက်သွယ် တဲ့ အခါ BBB ရဲ့ IO မို့ 3.3 V နဲ့ အဆင်

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

၄.၁.၅ Light Activated Switch

ခုနေက light sensor နဲ့ relay ကို သုံးပြီး ညာမောင် လာရင် အလိုအလောက် မီးဖွင့် ပေးမယ့် light activated switch လေး လုပ်ဖို့ အတွက် BBB နဲ့ ဆက်သွယ်ပုံ ကို ပုံ ၄.၆ မှာ ပြထား ပါတယ်။



ပုံ ၄.၆: Light activated switch အတွက် light sensor နှင့် relay ကို BBB ပြင် ဆက်သွယ်ပုံ။

အဲဒီ အတွက် နမူနာ ပရိုဂရမ် အနေနဲ့ [lightswitch.cpp](#) ကို စာရင်း ၄.၆ မှာ ပြထား ပါတယ်။ ရှောက နမူနာ အတိုင်း CE_IO class ကို သုံးပြီး GPIO_48 ကို light sensor input အနေ နဲ့ GPIO_49 ကို relay အတွက် output အနေ နဲ့ သုံးလိုက် ပါတယ်။ ပြီးတဲ့ အခါ while loop ကို အဆုံး မရှိ ပတ်နေပြီး light sensor က ဖတ်လို ရတဲ့ တန်ဖိုး ကို relay အတွက် ထူတ်ပေး 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 {

```

၄.J. ANALOG INPUT များကိုဖတ်ခြင်း

၉၁

```
8     CE_IO light(48, INPUT);
9     CE_IO relay(49, 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 }
```

စာရင်း ၄.၆: lightswitch.cpp

ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး run နိုင် ပါတယ်။ ပရိုဂရမ် ကို အဆုံးသတ် ဖို့ အတွက် ctrl+c ကို နှိပ်ပြီး ရပ်နှင့် ပါတယ်။

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

၄.J Analog input များကိုဖတ်ခြင်း

BBB မှာ analog input တွေက အစ ကတည်းက enable ဖြစ်နေ ပြီး တန်းသုံးလို့ ရပါတယ်။ အရင် firmware ဗားရှင်း အဟောင်း တွေမှာ တော့ နောက်ဆက်တွဲ A အပိုင်း ၁.၃ မှာ ပြထား သလို့ enable လုပ်ဖို့ လို ပါတယ်။ အဲဒါ ကို စစ်ကြည့်ဖို့ စာရင်း ၄.၇ က command တွေကို သုံးနိုင် ပါတယ်။

```
1 $ cd /sys/bus/iio/devices
2 $ ls
```

စာရင်း ၄.၇: Analog input device ကို စစ်ခြင်း။

အဲဒီ အခါ iio:device0 ဆိုတဲ့ device ကို တွေ့ရမှာ ဖြစ်ပါတယ်။ အဲဒီ အခန်းထဲ ဝင်ပြီး list လုပ် ကြည့်လို တွေ့ရတဲ့ analog pin တွေကို စာရင်း ၄.၈ ထဲကလို ဖတ်ကြည့်နိုင် ပါတယ်။

```
1 $ cd iio:device0
2 $ ls
3 $ more in_voltage0_raw
```

စာရင်း ၄.၈: Analog input များကို ဖတ်ခြင်း။

ဖတ်လို ရတဲ့ တန်ဖိုး တစ်ခုကို ပုံ ၄.၁၀ မှာ ပြထား ပါတယ်။ တကယ်လို အဝင် မှာ ဘာမှ ဆက်မထားရင် noise ကြောင့် တစ်ခါ နဲ့ တစ်ခါ ဖတ်လို ရတဲ့ တန်ဖိုး တွေ အပြောင်း အလဲ ရှိနိုင် ပါတယ်။

```
debian@beaglebone:/sys/bus/iio/devices$ ls
iio:device0
debian@beaglebone:/sys/bus/iio/devices$ cd iio:device0
debian@beaglebone:/sys/bus/iio/devices/iio:device0$ ls
buffer      in_voltage1_raw  in_voltage4_raw  name      scan_elements
dev          in_voltage2_raw  in_voltage5_raw  of_node   subsystem
in_voltage0_raw  in_voltage3_raw  in_voltage6_raw  power    uevent
debian@beaglebone:/sys/bus/iio/devices/iio:device0$ more in_voltage0_raw
3912
debian@beaglebone:/sys/bus/iio/devices/iio:device0$ more in_voltage0_raw
3911
debian@beaglebone:/sys/bus/iio/devices/iio:device0$ more in_voltage0_raw
3908
debian@beaglebone:/sys/bus/iio/devices/iio:device0$
```

ပုံ ၄.၁၀: Analog အဝင် များကို ဖတ်ခြင်း။

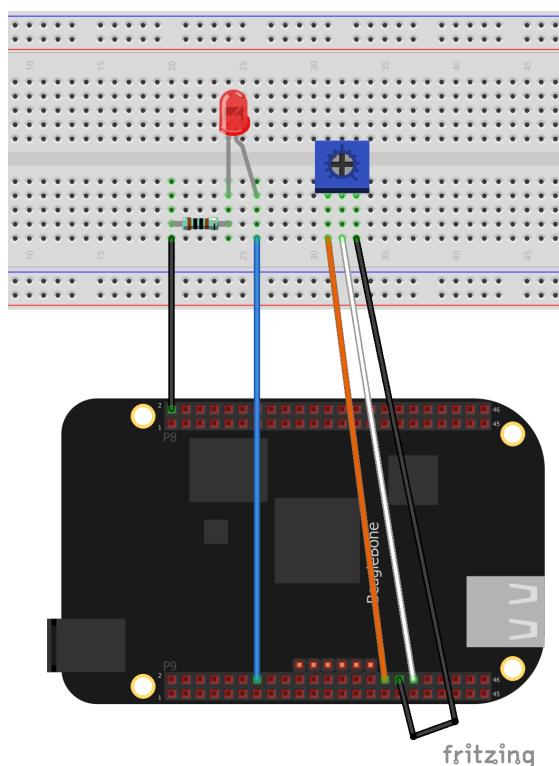
| P9 | | | P8 | | |
|----------|----|----|------------|--|--|
| DGND | 1 | 2 | DGND | | |
| VDD_3V3 | 3 | 4 | VDD_3V3 | | |
| VDD_5V | 5 | 6 | VDD_5V | | |
| SYS_5V | 7 | 8 | SYS_5V | | |
| PWR_BUT | 9 | 10 | SYS_RESETN | | |
| GPIO_30 | 11 | 12 | GPIO_60 | | |
| GPIO_31 | 13 | 14 | GPIO_50 | | |
| GPIO_48 | 15 | 16 | GPIO_51 | | |
| GPIO_5 | 17 | 18 | GPIO_4 | | |
| I2C2_SCL | 19 | 20 | I2C2_SDA | | |
| GPIO_3 | 21 | 22 | GPIO_2 | | |
| GPIO_49 | 23 | 24 | GPIO_15 | | |
| GPIO_117 | 25 | 26 | GPIO_14 | | |
| GPIO_115 | 27 | 28 | GPIO_113 | | |
| GPIO_111 | 29 | 30 | GPIO_112 | | |
| GPIO_110 | 31 | 32 | VDD_ADC | | |
| AIN4 | 33 | 34 | GNDA_ADC | | |
| AIN6 | 35 | 36 | AIN5 | | |
| AIN2 | 37 | 38 | AIN3 | | |
| AIN0 | 39 | 40 | AIN1 | | |
| GPIO_20 | 41 | 42 | GPIO_7 | | |
| DGND | 43 | 44 | DGND | | |
| DGND | 45 | 46 | DGND | | |

ပုံ ၄.၁၁: BBB နဲ့ Analog input pin များ။

BBB ရဲ P9_32 က ADC reference voltage (VDD_ADC) 1.8 V ဖြစ်ပြီး၊ P9_34 က ADC ground (GNDA_ADC)။ BBB ရဲ analog input pin တွက် ပုံ ၄.၁၁ မှာ ပြန် ဖော်ပြ ထားပြီး၊ အဲဒီ အဝင် တွက် 1.8 V ထက် ပို မပေး မိအောင် သတိပြု သင့် ပါတယ်။

၄.J.၁ C++ ဖြင့်အသုံးပြုခြင်း

Analog input တွက် C++ နဲ့ သုံးတဲ့ အကြောင်း ဆွေးနွေး ပါမယ်။ ရှေ့မှာ သူတို့ ရဲ device file တွက် 'more' command သုံးပြီး ဖတ်ပြ လိုက် သလိုပဲ၊ C++ မှာ file stream နဲ့ ပြောင်းဖတ် လိုက်တာ ပါပဲ။ အခန်း J မှာ သုံးခဲ့ တဲ့ setup အတိုင်း ပဲ ပြန်သုံး လိုက် ပါမယ်။ အဲဒီ analog input အတွက် potentiometer နဲ့ ဆက်သွယ်မှုကို ပုံ ၄.၁၂ မှာ ပြန် ဖော်ပြ ထား ပါတယ်။ နမူနာ အနေနဲ့ analog input ကို ဖတ်ပေး တဲ့ CE_Ai ဆိုတဲ့ class ကို ce_ai.h (စာရင်း ၄.၉) နဲ့ ce_ai.cpp (စာရင်း ၄.၁၀) မှာ ပြထား ပါတယ်။



ပုံ ၄.၁၂: Analog input အတွက် trimmer (potentiometer) တစ်ခု နှင့် analog output အတွက် LED တစ်ခု ကို ဆက်သွယ်ပဲ။

```

1 //File: ce_ai.h
2 //Description: Reading analog input pins
3 //WebSite: http://cool-emerald.blogspot.sg
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye

6
7 #ifndef CE_AI_H
8 #define CE_AI_H

9
10 #include <string>
11 #include <fstream>
12 #include <sstream>
13 #define AIPATH "/sys/bus/iio/devices/iio:device0/in_voltage"
14 using namespace std;

15
16 class CE_Ai{
17     string fpath;
18 public:
19     CE_Ai(long ain);
20     int Read();
21     template <typename T>
22         string n2s(T Number);
23     template <typename T>
24         T s2n(const string &Text);
25 };
26
27 #endif

```

စွဲပုံး၏ ၄.၁: ce_ai.h

```

1 //File: ce_ai.cpp
2 //Description: Reading analog input pins
3 //WebSite: http://cool-emerald.blogspot.sg
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye

```

```

6
7 #include "ce_ai.h"
8
9 template <typename T>
10    string CE_Ai::n2s(T Number)
11    {
12        ostringstream ss;
13        ss << Number;
14        return ss.str();
15    }
16
17 template <typename T>
18    T CE_Ai::s2n(const string &Text)
19    {
20        istringstream ss(Text);
21        T result;
22        return ss >> result ? result : 0;
23    }
24
25 CE_Ai::CE_Ai(long ain)
26 {
27     fpath = AIPATH+n2s(ain)+"_raw";
28 }
29
30 int CE_Ai::Read()
31 {
32     string str;
33     ifstream rfile;
34     int val=0;
35     rfile.open(fpath.c_str());
36     if (rfile.is_open()) {
37         if (getline(rfile,str)) {
38             val = s2n<int>(str);
39         }
40     }
41     rfile.close();

```

```

42     return val;
43 }
```

စာရင်း ၄.၁၀: ce_ai.cpp

အဲဒီ class ကို အသုံးပြုတဲ့ C++ ပရိုဂရမ် နမူနာ bbai.cpp ကို စာရင်း ၄.၁၁ မှာ ပြထား ပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include "ce_ai.h"
4 using namespace std;
5 int main()
6 {
7     CE_Ai ai(5);
8     for(int i=0;i<60;i++){
9         printf("i=%d ai= %d \n",i,ai.Read());
10        usleep(1000000);
11    }
12    return 0;
13 }
```

စာရင်း ၄.၁၁: bbai.cpp

ပရိုဂရမ် ကို စာရင်း ၄.၁၂ အတိုင်း Build လုပ်ပြီး Run နိုင် ပါတယ်။

```

1 $g++ bbai.cpp ce_ai.cpp -o bbai
2 $ ./bbai
```

စာရင်း ၄.၁၂: Analog input ကို ဖတ်ပြုသည့် ပရိုဂရမ် ကို run ခြင်း။

ပရိုဂရမ် ကို run နေတဲ့ အချိန် trimmer ကို လှည့်လိုက် ရင် ဖတ်လို့ ရတဲ့ analog တန်ဖိုး က အချိုးကျ လိုက်ပြောင်း သွားတာ ကို တွေ့နိုင် ပါတယ်။ BBB ရဲ့ ADC က 12 bits ဖြစ်တဲ့ အတွက် ဖတ်လို့ ရတဲ့ တန်ဖိုး က 0 ကနေ 4095 ထိ ဖြစ်နိုင် ပါတယ်။

၄.J.J Temperature sensor ဖြင့်အပူချိန်ကိုအာရုံခံခြင်း

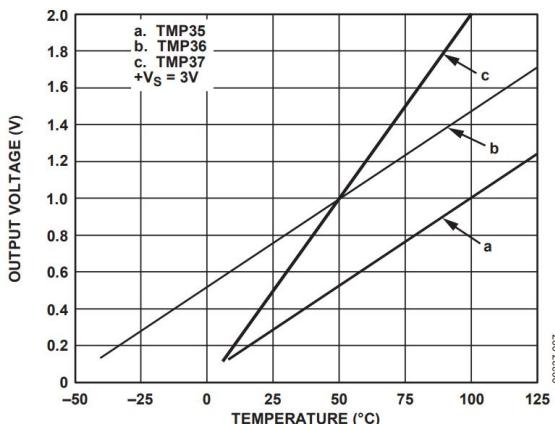
Analog Devices ကထုတ်တဲ့ TMP36 low voltage temperature sensor လေးက ပါဝါ အဝင် 2.7 V ကနေ 5.5 V ထိ အလုပ်လုပ် ပါတယ်။ ထုတ် ပေးတဲ့ analog output မိုက လည်း -55°C ကနေ +125°C ကို 0 V ကနေ 1.8 V အတွင်း ရှိတာ မို့ BBB ရဲ့ analog input reference မို့ (V_{aref}) 1.8 V နဲ့ အဆင်ပြီ ပါတယ်။ သူ့ရဲ့ datasheet [Ana15] မှာ ဖော်ပြုထားတာ က 50°C မှာ 1 V ထွက်ပြီး၊ ပုံ ၄.၁၃ မှာ ဖော်ပြုထားတဲ့ အတိုင်း 10 mV/°C နဲ့ linear ပြောင်းလဲ တယ် လို ဆိုတဲ့ အတွက် အထွက်မို့ (V_o) နဲ့ အပူချိန် (T) ရဲ့ ဆက်သွယ်ချက် ကို အောက်ပါ အတိုင်း ဖော်ပြုနိုင်ပါတယ်။

$$V_o = 1 + 0.01(T - 50) \quad (4.1)$$

BBB ရဲ့ ADC က 12 bits ဖြစ်တာ ကြောင့် ဖတ်လို ရမယ့် တန်ဖိုး (N) နဲ့ အပူချိန် (T) အတွက် ညီမျှခြင်း ၄.၃ အတိုင်း ဖော်ပြုနိုင် ပါတယ်။

$$N = \frac{4095}{V_{aref}}(0.01T + 0.5) \quad (4.2)$$

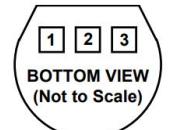
$$T = \frac{N \times V_{aref} - 2047.5}{40.95} \quad (4.2)$$



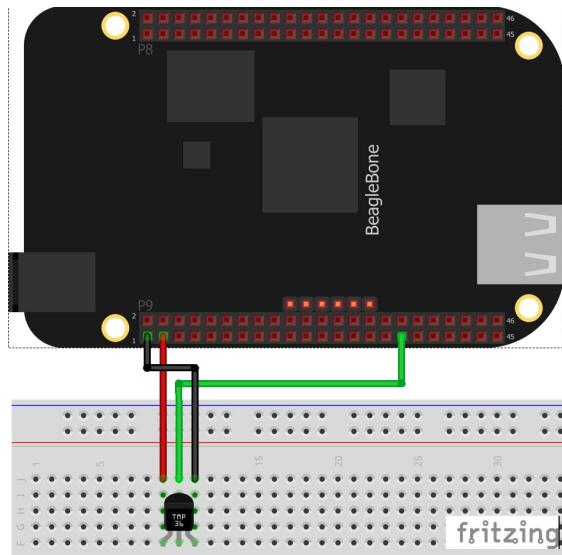
(a) အထွက်မို့ နှင့် အပူချိန် ဆက်သွယ်မှု။

ပုံ ၄.၁၃: TMP36 ၏ အထွက်မို့ နှင့် pin များ။ (From TMP36 datasheet)

ပုံ ၄.၁၃ မှာ TO-92 package အတွက် TMP36 ရဲ့ pin တွေကို ဖော်ပြထား ပြီး သူကို ပုံ ၄.၁၄ မှာ ပြထား သလို BBB နဲ့ ဆက်သွယ် လိုက် ပါမယ်။



(b) TO-92 package အတွက် pin configuration။



ပုံ ၄.၁၄: TMP36GT9Z နှင့် BBB ကို ဆက်သွယ်ပုံ။

အပူချိန်ကို အာရုံခံ တဲ့ ကိရိယာ ကို အသုံးပြု တဲ့ နမူနာ C++ ပရိုဂရမ် `tsensor.cpp` ကို စာရင်း ၄.၁၃ မှာ ပြထား ပါတယ်။ အပိုင်း ၄.၂၁ မှာ ဖော်ပြ ခဲ့တဲ့ `CE_Ai` class ကို ပြန်သုံး ထားပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include "ce_ai.h"
4 using namespace std;
5 int main()
6 {
7     CE_Ai tsensor(4);
8     int n;
9     float v, t;
10    for(int i=0;i<10;i++){
11        n = tsensor.Read();
12        v = 1.8*float(n)/ 4095;
13        t = (float(n)*1.8 - 2047.5) / 40.95;
14        printf("n= %d v=%f t=%f\n",n,v,t);
15        usleep(1000000);
16    }
17    return 0;
18 }
```

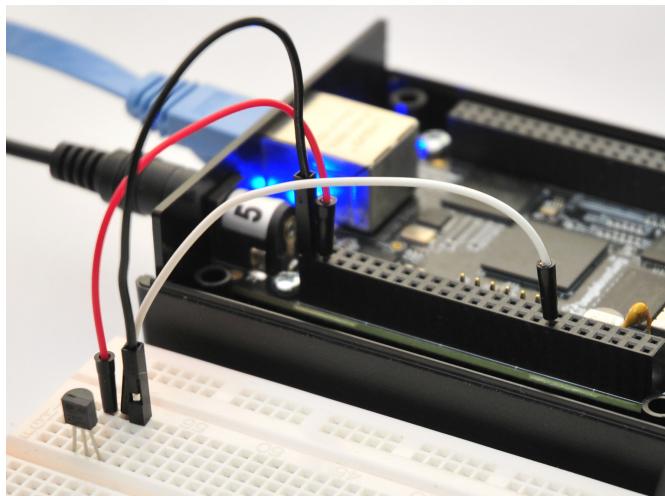
စာရင်း ၄.၁၃: tsensor.cpp

သူကို အောက်ပါ အတိုင်း build လုပ်၊ run လုပ်နိုင်ပြီး၊ ပရီဂရမ် ရဲ့ output ကို ပုံ ၄.၁၅ နဲ့ TMP36GT9Z ကို ပုံ ၄.၁၆ မှာ ပြထား ပါတယ်။

```
$ g++ tsensor.cpp ce_ai.cpp -o tsensor
$ ./tsensor
```

```
debian@beaglebone:/~/bbai$ g++ tsensor.cpp ce_ai.cpp -o tsensor
debian@beaglebone:/~/bbai$ ./tsensor
n= 1846 v=0.766349 t=31.142857
n= 1843 v=0.765104 t=31.010988
n= 1846 v=0.766349 t=31.142857
n= 1843 v=0.765104 t=31.010988
n= 1824 v=0.757216 t=30.175825
n= 1843 v=0.765104 t=31.010988
n= 1842 v=0.764689 t=30.967033
n= 1843 v=0.765104 t=31.010988
n= 1842 v=0.764689 t=30.967033
n= 1829 v=0.759292 t=30.395605
debian@beaglebone:/~/bbai$ █
```

ပုံ ၄.၁၅: tsensor.cpp ၏ရလဒ်။



ပုံ ၄.၁၆: TMP36GT9Z နှင့် BBB၏

၄.၃ Analog output ထုတ်ခြင်း

BBB မှာ digital to analog converter (DAC) မပါရှိ ပါဘူး။ Analog output တွေရဖို့ အတွက် Pulse Width Modulation (PWM) signal ကို ထုတ်သုံး နိုင် ပါတယ်။ PWM device တွေကို enable လုပ်ဖို့ အတွက် /boot/uEnv.txt မှာ edit လုပ်ဖို့လို ပါတယ် [Rol17]။ အဲဒီ အတွက် firmware libraries တွေကို list လုပ်ကြည့်ပြီး /sys/class/pwm အခန်း ကို လည်း list လုပ်ကြည့် နိုင် ပါတယ်။

```
$ ls /lib/firmware | grep PWM
$ cd /sys/class/pwm
$ ls
$ sudo nano /boot/uEnv.txt
```

နိုင်နာ အနေနဲ့ P9_14 က EHRPWM1A ကို အသုံးပြု ကြည့် ပါမယ်။ အဲဒီ အတွက် PWM1 ကို enable လုပ်ဖို့ /boot/uEnv.txt မှာ အောက်က စာကြောင်း ကို ဖြည့်ပြီး တဲ့အခါ စက်ကို reboot လုပ်နိုင် ပါတယ်။

```
uboot_overlay_addr1=/lib/firmware/BB-PWM1-00A0.dtbo
```

အရင် firmware အဟောင်း တွေ အတွက်တော့ နောက်ဆက်တဲ့ A အပိုင်း ၁.၄ မှာ ခွေးနှံး ထားသလို Capemgr သုံးပြီး တပ်ဆင် နိုင် ပါတယ်။

အဲဒီ နောက် /sys/class/pwm ကို list လုပ်ကြည့် တဲ့ အခါ pwmchip0 ဆိုပြီး ပေါ်လာ တာကို တွေ့ရ ပါမယ်။ အဲဒီ အခန်း ထဲကို ဝင်ပြီး EHRPWM1A အတွက် ဆိုရင် ၀၊ EHRPWM1B အတွက် ဆိုရင် ၁ ကို export လုပ်နိုင် ပါတယ်။

```
$ cd /sys/class/pwm
$ ls
$ cd pwmchip0
$ ls
$ echo 0 > export
$ ls
```

Export မှာ ၀ ရေးပြီး တဲ့ အခါ pwm0 ဆိုတဲ့ အခန်း ပေါ်လာ တာကို တွေ့ရ မှာပါ။ အဲဒီ အခန်း ထဲမှာ ထုတ်ချင်တဲ့ PWM waveform ရဲ့ period, duty cycle စတာ တွေကို သတ်မှတ် ပြီး enable လုပ်လိုက်

ရင် PWM waveform ထွက်လာတာ ကို တွေ့နှင်ပါတယ်။ ပုံ ၄.၂၂ မှာ ဖော်ပြထားသလို P9_14 က EHRPWM1A မှာ LED မီးလုံးနဲ့ ဆက်သွယ် ကြည့်ရင် duty cycle အပြောင်းအလဲ ပေါ်မူတည်ပြီး မီးအလင်း၊ အမိန့်ပြောင်းလဲသွားတာ ကို တွေ့နှင်ပါတယ်။

```
$ cd pwm0
$ ls
$ sudo sh -c "echo 1000000000 > period"
$ sudo sh -c "echo 500000000 > duty_cycle"
$ sudo sh -c "echo 1 > enable"
```

အဲဒီ မှာ period နဲ့ duty cycle အတွက် တန်ဖိုး တွေက nanoseconds ဖြစ်တဲ့ အတွက်၊ ဒီ နမူနာ မှာ တစ်စကြေနှင့် တစ်ခါ အဖွင့် အပိတ် ဖြစ်နေတာ ကို တွေ့ရမှာ ပါ။

၄.၃.၁ C++ ဖြင့်အသုံးပြုခြင်း

Analog output အတွက် PWM signal ထွေကို C++ သုံးပြီး ထုတ်တဲ့ အခါ device file ထွေကို ခုနကလို echo command သုံးပြီး မရေးပဲ file stream နဲ့ ပြောင်းရေး လိုက်တာ ပါပဲ။ Analog output နမူနာ C++ class တစ်ခု ကို ce_ao.h (စာရင်း ၄.၁၅) နဲ့ ce_ao.cpp (စာရင်း ၄.၁၆) မှာ ပြထားပါတယ်။ အဲဒီ class ကို သုံးတဲ့ နမူနာ bba0.cpp ကို စာရင်း ၄.၁၆ မှာ ဖော်ပြထားပါတယ်။

```
1 //File: ce_ao.h
2 //Description: pwm output
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_AO_H
8 #define CE_AO_H
9
10 #include <string>
11 #include <fstream>
12 #include <sstream>
13 #include <stdio.h>
14
15 #define CE_PWMPATH "/sys/class/pwm/pwmchip"
```

LOC

အခန်း ၄. INPUT/OUTPUT များအသုံးပြုခြင်း

```
16 using namespace std;
17
18 class CE_Ao{
19     string fpath;
20     int m_chip;
21     int m_output;
22 public:
23     CE_Ao(int pwm_chip_no,int pwm_output_no);
24     ~CE_Ao();
25     int Begin(int pwm_chip_no,int pwm_output_no);
26     int Export();
27     int Period(int t_ns);
28     int Duty(int t_ns);
29     int Enable(bool v);
30     template <typename T>
31         string n2s(T Number);
32 //     template <typename T>
33 //         T s2n(const string &Text);
34 };
35
36 #endif // CE_AO_H
```

စာရင်း ၄.၁၄။ ce_ao.h

```
1 //File: ce_ao.cpp
2 //Description: pwm output
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 "ce_ao.h"
8
9 CE_Ao::CE_Ao(int pwm_chip_no,int pwm_output_no)
10 {
11     Begin(pwm_chip_no,pwm_output_no);
```

```

12 }
13
14 CE_Ao ::~CE_Ao()
15 {
16 // if unexported, pwm cannot be exported again, requiring reboot
17 // ofstream wfile;
18 // string epath=CE_PWMPATH;
19 // epath+=n2s(m_chip)+"/unexport";
20 // wfile.open(epath.c_str());
21 // if (wfile.is_open()) {
22 //     wfile << m_output;
23 // }
24 // wfile.close();
25 }

26
27 int CE_Ao::Begin(int pwm_chip_no,int pwm_output_no)
28 {
29     m_chip=pwm_chip_no;
30     m_output=pwm_output_no;
31     fpath = CE_PWMPATH+n2s(m_chip)+"/pwm"+n2s(m_output);
32     if(Export()<0) return -1;
33     return 0;
34 }

35
36 int CE_Ao::Export()
37 {
38     ofstream wfile;
39     int r=-1;
40     string epath=CE_PWMPATH;
41     epath+=n2s(m_chip)+"/export";
42     wfile.open(epath.c_str());
43     if (wfile.is_open()) {
44         wfile << m_output;
45         r=0;
46     }
47     wfile.close();

```

```
48     return r;
49 }
50
51 int CE_Ao::Period(int t_ns)
52 {
53     ofstream wfile;
54     string path=fpath;
55     int r=-1;
56
57     path+="/period";
58     wfile.open(path.c_str());
59     if (wfile.is_open()) {
60         wfile << t_ns;
61         r=0;
62     }
63     wfile.close();
64     return r;
65 }
66
67
68 int CE_Ao::Duty(int t_ns)
69 {
70     ofstream wfile;
71     string path=fpath;
72     int r=-1;
73
74     path+="/duty_cycle";
75     wfile.open(path.c_str());
76     if (wfile.is_open()) {
77         wfile << t_ns;
78         r=0;
79     }
80     wfile.close();
81     return r;
82 }
83
```

၂၇. ANALOG OUTPUT ဆုတ္တခါး

၁၀၅

```
84 int CE_Ao::Enable(bool v)
85 {
86     ofstream wfile;
87     string path=fpath;
88     int r=-1;
89
90     path+="/enable";
91     wfile.open(path.c_str());
92     if (wfile.is_open()) {
93         wfile << (v?1:0);
94         r=0;
95     }
96     wfile.close();
97     return r;
98 }
99
100 template <typename T>
101     string CE_Ao::n2s(T Number)
102 {
103     ostringstream ss;
104     ss << Number;
105     return ss.str();
106 }
107
108 //template <typename T>
109 //    T CE_Ao::s2n(const string &Text)
110 //{
111 //    istringstream ss(Text);
112 //    T result;
113 //    return ss >> result ? result : 0;
114 //}
```

စာရင်း ၂၃၂: ce_ao.cpp

```
1 #include <stdio.h>
```

```
2 #include <unistd.h>
3 #include "ce_ao.h"
4 using namespace std;
5 int main()
6 {
7     CE_Ao P9_14(0,0); //pwmchip0, pwm0
8     printf("PWM signal output at P9_14\n");
9     P9_14.Period(1000000000);
10    P9_14.Duty(500000000);
11    P9_14.Enable(true);
12    for(int i=0;i<10;i++) usleep(1000000); //wait 10 sec
13    P9_14.Enable(false);
14
15 }
```

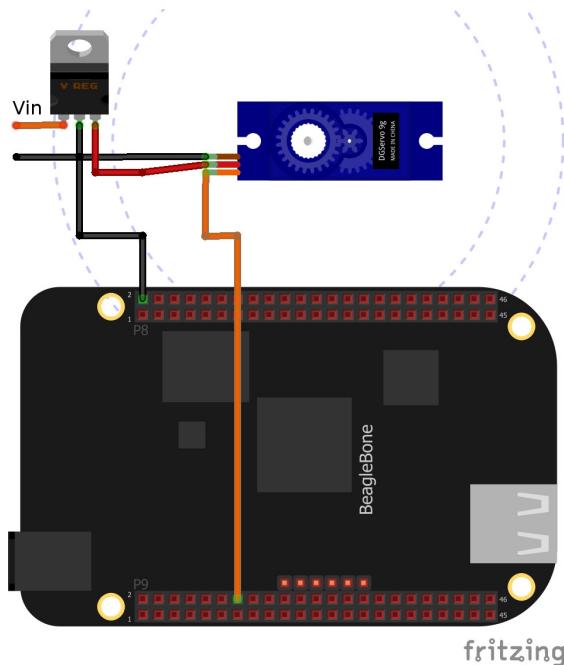
စာရင်း ၄.၁၆: bbao.cpp

ပရိုဂရမ် ကို super user အနေဖြင့် သုံးဖို့လိုပြီး၊ အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run နှင့် ပါတယ်။

```
$ g++ bbao.cpp ce_ao.cpp -o bbao  
$ sudo ./bbao
```

၄.၃.၂ Servo motor ကိုထိန်းချုပ်ခြင်း

BBB ကို သုံးပြီး servo motor အသေးလေး တစ်ခု ကို ထိန်းချုပ် ကြည့်ပါမယ်။ သူမှာ ပါဝါ အတွက် V_m (အနီရောင်ဝါယာ) နဲ့ GND (အညီရောင်ဝါယာ) ရယ်၊ control အတွက် ငှုတ် (လိမ္မာ်ရောင်ဝါယာ) ရယ် စုစု ပေါင်း ငှုတ် သုံးခု ပါပါတယ်။ အခု ဒီ SG90 Tower Pro micro servo motor လေးက 9 g ပဲ လေးပြီး၊ အလုပ်လုပ် တဲ့ ဗိုအား က 4.8 V ကနေ 6 V အထိ ဖြစ် ပါတယ်။ လှည့်ပေးနိုင် တဲ့ အမြန်နှုန်းက $500^{\circ}/s$ ဖြစ် ပါတယ်။ သူကို BBB နဲ့ ဆက်သွယ်တဲ့ ပုံကို ပုံ ၄.၁၇ မှာ ပြထား ပါတယ်။



ပုံ ၄.၁၇: Servo motor ကို ထိန်းချုပ်ခြင်း။

ဒါ servo motor လေးကို ထိန်းချုပ် တဲ့အခါ 50 Hz PWM wave ကို သုံးမှာ ဖြစ်လို့ သူရဲ့ period ကို 20 ms သတ်မှတ်လိုက် ပါတယ်။ Duty cycle 0.6 ms က မောင်တာ ရဲ့ -90° ဖြစ်ပြီး၊ 2.4 ms က +90° အသိုးသိုး အချိုးကျ ဖြစ်ပါတယ် [Hom13]။ Server motor ကို -90° ကနေ +90° ထိ အဆင့်ဆင့် လွည့်ပြတဲ့ နမူနာ ပရိုဂရမ် smotor.cpp ကို စာရင်း ၄.၁၇ မှာ ပြထား ပါတယ်။

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include "ce_ao.h"
4 using namespace std;
5 int main()
6 {
7     CE_Ao smotor(0,0); //pwmchip0, pwm0
8     int duty = 0;
9     int angle = 0;
10    printf("Moving servo motor from -90 deg to +90 deg.\n");
11    smotor.Period(20000000); //20 ms -> 50 Hz
12    smotor.Duty(600000); //0.6 ms for -90
13    smotor.Enable(true);

```

```

14 usleep(3000000);
15 for (int i = -9; i <= 9; i++) {
16     duty = 100000 * (i+15);
17     // -90 deg to 90 deg corresponds to 600000 to 2400000
18     angle = i*10;
19     smotor.Duty(duty);
20     printf("Angle = %d \n", angle);
21     usleep(1000000); // wait
22 }
23 smotor.Enable(false);
24 return 0;
25 }
```

စာရင်း ၄.၁၇: smotor.cpp

```

debian@beaglebone:~/bbao$ sudo ./smotor
Moving servo motor from -90 deg to +90 deg.
Angle = -90
Angle = -80
Angle = -70
Angle = -60
Angle = -50
Angle = -40
Angle = -30
Angle = -20
Angle = -10
Angle = 0
Angle = 10
Angle = 20
Angle = 30
Angle = 40
Angle = 50
Angle = 60
Angle = 70
Angle = 80
Angle = 90
debian@beaglebone:~/bbao$
```

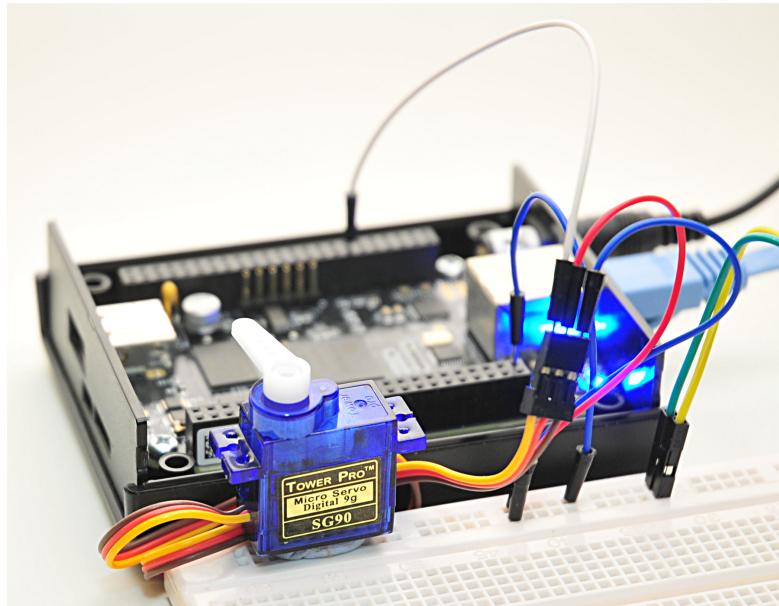
ပုံ ၄.၁၈: smotor.cpp ဇီရလတ်။

ပရိုဂရမ် ကိုအောက်ပါ အတိုင်း build လုပ်ပြီး၊ super user အနေနဲ့ run နိုင် ပါတယ်။

```

$ g++ smotor.cpp ce_ao.cpp -o smotor
$ sudo ./smotor
```

ပရီဂရမ် ရဲရလဒ် နဲ့ micro servo motor ကို BBB နဲ့ဆက်ထား တာကို ပုံ နဲ့ပုံ မှာ တွေ့နိုင် ပါတယ်။



ပုံ ၄.၁၉: Servo motor ကို BBB နှင့်ဆက်သွယ်ပုံ။

အကိုးအကားများ

- [Hom13] HomoFaciens. Servos - working principle and homemade types. 2013. url: <https://www.youtube.com/watch?v=v2jpnyKPH64>.
- [Rol17] Roland. Beaglebone Black PWM on Ubuntu 16.04 Using Device Tree Overlay. 2017. url: <https://www.teachmemicro.com/beaglebone-black-pwm-ubuntu-device-tree/>.
- [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.

အခန်း ၅

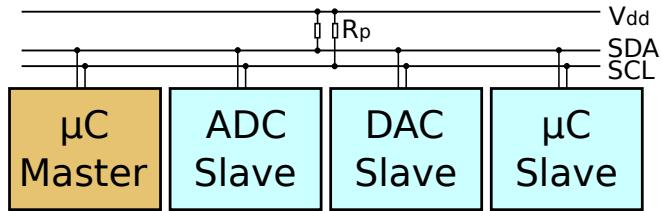
Serial Bus အနီးဆက်သွယ်ရေးများ

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

၅.၁ 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 တိတောင် ခဲ့တာ ပါ။

I2C မှာ Serial Data Line (SDA) နဲ့ Serial Clock Line (SCL) လိုခေါ်တဲ့ 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 ၅.၁ မှာ ပြထား ပါတယ် [Wik18a]။



ပုံ ၅.၁: I2C master အနေနှင့် သုံးထားသည့် microcontroller တစ်ခု ကို slave သုံးခဲ့အား pull-up resistor များသုံး၍ ဆက်သွယ်ပုံ။

BeagleBone မှာ I2C bus သုံးခဲ့ပါပြီး၊ သူတို့၏ Debian Linux သတ်မှတ်ချက်နဲ့ pin တွေကို ဧယား ၅.၁ မှာ တွေ့နှိုင် ပါတယ် [Mol14]။ တချို့ firmware အဟောင်း တွေမှာ i2c-1 နဲ့ i2c-2 က လွှာနေတာ ကို သတိပြု ဖို့ လိုပါတယ်။ ဒါကြောင့် i2c-1 နဲ့ မရ ရင် i2c-2 နဲ့ အပြန် အလှန် ပြန် စမ်းသပ် ကြည့်ဖို့ သတိပြု စေချင် ပါတယ်။

ဧယား ၅.၁: BeagleBone Black I2C bus များ။

| Device | SCL | SDA | ဖော်ပြချက် |
|--------|-------|-------|--|
| i2c-0 | NA | NA | HDMI က အသုံးပြု ထားသည် |
| i2c-1 | P9_17 | P9_18 | ပုံမှန်အားဖြင့် disabled ဖြစ်နေပြီး enable လုပ်၍ သုံးနိုင် သည် |
| i2c-2 | P9_19 | P9_20 | ပုံမှန်အားဖြင့် enable ဖြစ် နေပြီး တန်း အသုံးပြု နိုင်သည် |

I2C1 ကို enable လုပ်ဖို့ အတွက် /boot/uEnv.txt ကို အောက်က အတိုင်း edit လုပ်ပြီး၊ အဲဒီ ဖိုင်ထဲမှာ BB-I2C1-00A0.dtbo ကို စာရင်း ၅.၁ မှာ ပြထား သလိုမျိုး သတ်မှတ် ပြီးတဲ့ အခါ reboot လုပ်ဖို့ လို ပါတယ်။

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

```
1 uboot_overlay_addr2=/lib/firmware/BB-I2C1-00A0.dtbo
```

စာရင်း ၅.၁: I2C1 ကို /boot/uEnv.txt တွင် enable လုပ်ခြင်း။

အရင် firmware အဟောင်း တွေ အတွက် enable လုပ်ဖို့ အတွက်တော့ နောက်ဆက်တဲ့ A အပိုင်း ၁.၅ မှာ ခွေးနေးထား ပါတယ်။

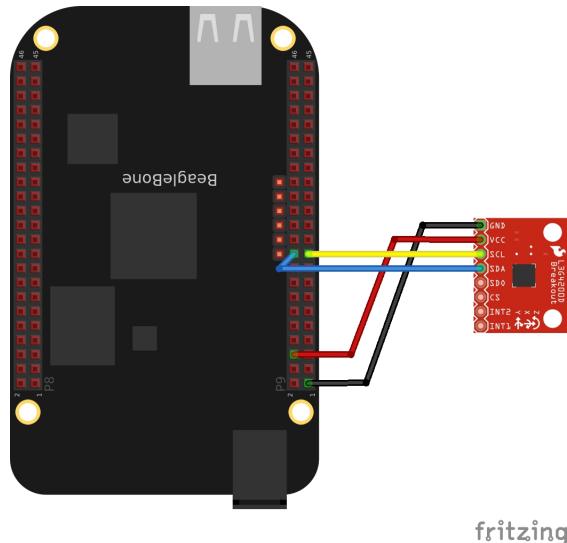
I2C ကို အသုံးပြု ဖို့ အတွက် linux ရဲ့ i2c-tools ကို အောက်ပါ အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
$ 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 တိုက်ရိုက် လုပ်ပေး မှာ ဖြစ်ပါတယ်။ "-r" ဆိုရင်တော့ receive byte ဖြစ်ပြီး probing လုပ်ကြည့်ဖို့ သုံးနိုင် ပါတယ်။ ပုံ ၅.၂ မှာ ပြထား သလို i2c-2 မှာ [L3G4200D Gyroscope module](#) ကို ဆက်သွယ်ပြီး probe လုပ်ကြည့်တဲ့ အခါ ပုံ ၅.၃ အတိုင်း တွေ့ရ ပါတယ်။



ပုံ ၅.၂: Gyroscope ကို I2C သုံး၍ ဆက်သွယ်ခြင်း။

```
$ i2cdetect -y -r 2
```

```
debian@beaglebone:~$ i2cdetect -y -r 2
  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- - - - - - - - - - - - - - - - - - - - - - -
10: - - - - - - - - - - - - - - - - - - - - - - -
20: - - - - - - - - - - - - - - - - - - - - - - -
30: - - - - - - - - - - - - - - - - - - - - - - -
40: - - - - - - - - - - - - - - - - - - - - - - -
50: - - - - - - UU UU UU UU - - - - - - - - - -
60: - - - - - - - - - - - - - - - - - - 69 - - - - -
70: - - - - - - - - - - - - - - - - - - - - - - -
```

ပုံ ၅.၃: I2C bus ကို probe လုပ်ခြင်း။

Probe လုပ်ကြည့် လို ဘာ အဖြစ်မရတဲ့ address တွေမှာ “-” လို ပြ နေမှာပါ။ ”UU” ကတော့ driver တွေ လက်ရှိ သုံးနေတဲ့ address မိ probe မလုပ်ဘူး လို ဆိုလို တာပါ။ အဲဒီ address တွေမှာ chip တွေရှုနေတယ် လို ပြောနိုင် ပါတယ်။ Hexadecimal နံပါတ် တစ်ခု တွေရှင်တော့ chip ကို တွေ့တဲ့ address လို ဆိုလိုတာပါ။ ဒီ နမူနာ မှာ gyroscope ရဲ address ကို 0x69 လို တွေ့ရ ပါတယ်။ L3G4200 ရဲ datasheet [STM10] ထဲမှာ ဖော်ပြထားတဲ့ အတိုင်း SDO pin ကို voltage supply ပေးထားရင် ရမည့် slave address ရဲ binary နံပါတ် 1101001 နဲ့ ကိုက်ညီ တာကို တွေ့ရ ပါတယ်။

Address 0x54 ကို သုံးထားတဲ့ driver ကို သိဖို့ အောက်က command နဲ့ ကြည့်လိုက်ရင် 24C256 EEPROM ကို တွေ့နိုင် ပါတယ်။

```
$ cd /sys/bus/i2c/devices/2-0054
$ more modalias
```

I2C bus မှာ ချုပ်ထားတဲ့ device ရဲ register တွေကို ဖတ်ရှုံး ဖော်ပြန့် အတွက် i2cdump ကို သုံးလိုရ ပါတယ်။ နမူနာ command နဲ့ ရလဒ် ကို ပုံ မှာ ပြထား ပါတယ်။ Address 0x0F က Who am I register ရဲ တန်ဖိုး 0xD3 ကို အဂ္ဂယ် တကူ စစ်ကြည့် နိုင် ပါတယ်။

```
$ i2cdump -y 2 0x69
```

```
debian@beaglebone:~$ i2cdump -y 2 0x69
No size specified (using byte-data access)
    0   1   2   3   4   5   6   7   8   9   a   b   c   d   e   f
00: c4 66 a4 cc 4c d0 9a c1 41 05 e4 d5 38 87 00 d3  0123456789abcdef
10: 36 26 66 44 0c a0 40 40 8c 99 71 80 83 80 81 81 ?f??L???A???8?.?
20: 07 00 00 00 00 00 ff 52 05 58 06 9c f3 00 20 6&fD??@@??q??????
30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ?.....R?X??..
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 52 05 58 06 9c f3 00 20 ?.....R?X??..
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 . .....
debian@beaglebone:~$
```

ပုံ ၅.၄: I2C bus ရှိ device တစ်ခု၏ register များကို ဖတ်ခြင်း။

I2C device ရဲ့ register တန်ဖိုး တွေကို ဖတ်ဖို့၊ ရေးဖို့ အတွက် i2cget နဲ့ i2cset တို့ကို သုံးနှင့် ပါတယ်။ ဥပမာ i2c-2 မှာချိတ်ထားတဲ့ slave address 0x69 ရဲ့ register address 0x21 မှာ ရှိတဲ့ CTRL_REG2 ကို ဖတ်ပြီး၊ တန်ဖိုး ကို 0x09 ပြောင်း ရေး တဲ့ command တွေကို အောက်က အတိုင်း တွေ့နှင့် ပါတယ်။

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

၅.၁.၁ C++ဖြင့်သုံးခြင်း

I2C bus ကို အသုံးပြု ပြီး device တွေနဲ့ ဆက်သွယ် 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 သံဃ္ပီး 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 သံဃ္ပီးနိုင်ပြီး file handle ရယ်။ ဖတ်လို့ ရတဲ့ data တွေ သိမ်းဖို့ buffer ရယ်။ ဖတ်မယ့် အရေအတွက် ရယ် ပေးဖို့လိုပါတယ်။

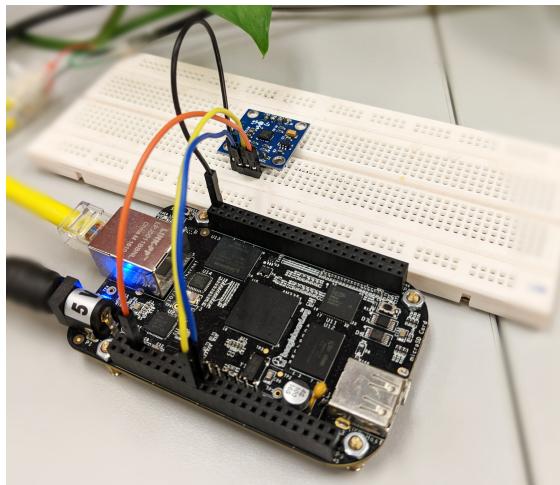
```
1 char buf[2];
2 if (read(file, buf, 2) !=2) {
3     printf("Failed to read from the i2c bus.\n");
4 }
```

ရေးမယ် ဆိုရင်တော့ write ကို သံဃ္ပီးနိုင် ပါတယ်။ ခုနက လိုပဲ file handle ရယ်။ ရေးမယ့် data တွေသိမ်းထားတဲ့ buffer ရယ်။ ရေးမယ့် အရေအတွက် ရယ် ပေးဖို့လိုပါတယ်။

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

၅၁.၂ Gyroscope ကိုအသုံးပြုခြင်း

I2C သုံးပြီး L3G4200D gyroscope ကို အသုံးပြု တဲ့ နမူနာ C++ ပရိုဂရမ် ကို [ceutil.h](#) (စာရင်း ၅.၂), [cei2c.h](#) (စာရင်း ၅.၃), [cei2c.cpp](#) (စာရင်း ၅.၄) နဲ့ [i2cgyro.cpp](#) (စာရင်း ၅.၅) ဖော်ပြ ထား ပါတယ်။ အဲဒီ အတွက် L3G4200D gyroscope module လေးကို ပုံ ၅.၂ နဲ့ ပုံ မှာ ပြထား သလို ဆက်သွယ် ထား ပါမယ်။



ပုံ ၅.၅: L3G4200D gyroscope module ကို တပ်ဆင်အသုံးပြုခြင်း။

```

1 #ifndef CEUTIL_H
2 #define CEUTIL_H
3 #include <string>
4 #include <sstream>
5 using namespace std;
6
7 template <typename T>
8 string ToString(T Number)
9 {
10     ostringstream ss;
11     ss << Number;
12     return ss.str();
13 }
14
15 template <typename T>
```

```

16 T FromString(const string &Text)
17 {
18     istringstream ss(Text);
19     T result;
20     return ss >> result ? result : 0;
21 }
22 #endif

```

စာရင်း ၅.J: ceutil.h

```

1 #ifndef CE_I2C_H
2 #define CE_I2C_H
3
4 #include <string.h>
5 #include <stdio.h>
6 #include <stdlib.h>
7 #include <unistd.h>
8 #include <linux/i2c-dev.h>
9 #include <sys/ioctl.h>
10 #include <fcntl.h>
11
12 //http://elinux.org/Interfacing_with_I2C_Devices
13
14 class CE_I2C
15 {
16     public:
17         CE_I2C();
18         CE_I2C(int bus_id,int slave_address);
19         virtual ~CE_I2C();
20         bool Begin(int bus_id,int slave_address);
21         bool Write(char* buf,int n);
22         bool Read(char* buf,int n);
23     private:
24         int fd;
25 };

```

```
26
27 #endif // CE_I2C_H
```

თურქული: ე.რ: cei2c.h

```
1 #include "cei2c.h"
2 #include "ceutil.h"
3 using namespace std;
4
5 #define PRINT_MES 0
6 CE_I2C::CE_I2C()
7 {
8     //ctor
9 }
10
11 CE_I2C::CE_I2C(int bus_id,int slave_address)
12 {
13     Begin(bus_id,slave_address);
14 }
15
16 CE_I2C::~CE_I2C()
17 {
18     //dtor
19 }
20
21 bool CE_I2C::Begin(int bus_id,int slave_address)
22 {
23     string filename = "/dev/i2c-";
24     filename+=ToString(bus_id);
25     if ((fd = open(filename.c_str(), O_RDWR)) < 0) {
26         #if PRINT_MES==1
27             printf("Failed to open the i2c bus\n");
28         #endif //
29         return false;
30     }
```

```
31     if (ioctl(fd, I2C_SLAVE, slave_address) < 0) {
32         #if PRINT_MES==1
33             printf("Failed to acquire bus access and/or talk to slave.\n");
34         #endif
35         return false;
36     }
37     #if PRINT_MES==1
38         printf("I2C beginning \'%s\' successful.\n",filename.c_str());
39     #endif // PRINT_MES
40     return true;
41 }
42
43 bool CE_I2C::Write(char* buf,int n)
44 {
45     if(write(fd,buf,n)!=n){
46         #if PRINT_MES==1
47             printf("Failed to write to the i2c bus.\n");
48         #endif // PRINT_MES
49         return false;
50     }
51     #if PRINT_MES==1
52         printf("I2C writing %d byte(s) successful.\n",n);
53     #endif // PRINT_MES
54     return true;
55 }
56
57 bool CE_I2C::Read(char* buf,int n)
58 {
59     if(read(fd,buf,n)!=n){
60         #if PRINT_MES==1
61             printf("Failed to read from the i2c bus.\n");
62         #endif // PRINT_MES
63         return false;
64     }
65     #if PRINT_MES==1
66         printf("I2C reading %d byte(s) successful.\n",n);
```

၂၁။ I2C

၁၂၁

```
67     #endif // PRINT_MES  
68     return true;  
69 }
```

၁၃၈။ cei2c.cpp

```
1 #include<stdio.h>  
2 #include"cei2c.h"  
3 #include "ceutil.h"  
4 using namespace std;  
5 int main()  
6 {  
7     char d[6]={0,0,0,0,0,0};  
8     CE_I2C gyro(2,0x69);  
9  
10    //read the "who am i" register at address 0x0F  
11    //its value should be 0xD3  
12    char raddr=0x0F;  
13    gyro.Write(&raddr,1);  
14    gyro.Read(d,1);  
15    printf("I am 0x%02x\n",d[0]);  
16  
17    //Configure Gyro  
18    //CTRL_REG2 = 10 10 | HPM1 / HPM0 / HPCF3 / HPCF2 / HPCF1 / HPCF0 /  
19    //Default = 10 10 10 10 10 10 10 10 10 |  
20    //HPM = 00 => Normal (High Pass filter Mode selection)  
21    //HPC = 1001 => 0.1 High Pass Filter Cut-off freq configuration  
22    //write @address 0x21, value = 0x09  
23    d[0]=0x21;  
24    d[1]=0x09;  
25    gyro.Write(d,2);  
26  
27    //CTRL_REG3 = / I1_Int1 / I1_Boot / H_Lactive / PP_OD / I2DRDY / I2_WTM / I2_ORun /  
28    //I2_Empy /  
29    //Default = 10 10 10 10 10 10 10 10 10 |
```

```

29
30
31 //CTRL_REG4 = /BDU/BLE/FS1/FS0/ - /ST1/ST0/SIM/
32 //Default =  /0  /0  /0  /0  /0  /0  /0  /0  /0  /
33 //BDU = 1 => Block Data Update
34 //BLE = 0 => Little endian
35 //FS = 00 => 250 dps (Full scale selection)
36 //ST = 000 => Disable Self test
37 //write @ address 0x23, value =0x80
38 d [0]=0x23;
39 d [1]=0x80;
40 gyro.Write(d,2);

41
42 //CTRL_REG5 = /BOOT/FIFO_EN/ - /HPen/INT1_Sel1/INT1_Sel0/Out_Sel1/
43 Out_Sel0/
44 //Default =  /0  /0  /0  /0  /0  /0  /0  /0  /0  /0
45 /
46 //BOOT = 0 => Normal mode (Reboot Memory Content)
47 //FIFO_EN = 0 => disable FIFO
48 //HPen = 0 => disable (High Pass Filter)
49 //INT1_Sel = 00 => Non high pass filtered data are used for interrupt
50 generation
51 //Out_Sel = 00 => no filtering
52 //Use Default

53
54 //CTRL_REG1 = /DR1/DR0/BW1/BW0/PD/Zen/Yen/Xen/
55 //Default =  /0  /0  /0  /0  /0  /1  /1  /1  /
56 //DR = 11 => ODR 800 Hz (output data rate)
57 //BW = 10 => Cut-off 50 (Bandwidth 50 Hz)
58 //PD = 1 => Normal
59 //Zen = Yen = Xen = 1 => Enable
60 //write @ address 0x20, value =0xEF
61 d [0]=0x20;
62 d [1]=0xEF;
63 gyro.Write(d,2);

```

```

61     int x,y,z;

62

63     for(int i=0;i<20;i++)//with T=0.1 for 30 sec
64     {
65         //read address 0x28, OUT_X_L register
66         //set MSB bit for auto address increment
67         raddr=0xA8;
68
69         gyro.Write(&raddr,1);
70
71         gyro.Read(d,6);
72
73         x=((int(d[1])<<8)+d[0])|(d[1]&0x80?0xFFFF0000:0x00000000);
74
75         y=((int(d[3])<<8)+d[2])|(d[3]&0x80?0xFFFF0000:0x00000000);
76
77         z=((int(d[5])<<8)+d[4])|(d[5]&0x80?0xFFFF0000:0x00000000);
78
79         //Print results
80
81         //printf("x y z = %02x%02x %02x%02x %02x%02x\n",d[1],d[0],d[3],d[2],d
82         [5],d[4]);
83
84         printf("x y z = %d %d %d\n",x,y,z);
85
86         usleep(500000);
87
88     }
89
90     return 0;
91 }
```

စာရင်း ၅.၅: i2cgyro.cpp

ပရိုဂရမ် ကို အောက်ပါ အတိုင်း build လုပ်ပြီး၊ run လိုက်တဲ့ အခါ Gyroscope ရဲ့ လူပ်ရှား မှုပေါ်
မူတည်ပြီး သက်ဆိုင်ရာ ဝင်ရှိး တွေရဲ့ တန်ဖိုး တွေ လူပ်ရှား ပြောင်းလဲ နေတာ ကို ပုံ ၅.၆ မှာ ပြထား
သလို တွေရ မှာ ဖြစ်ပါတယ်။

```
$ g++ i2cgyro.cpp cei2c.cpp -o i2cgyro
$ ./i2cgyro
```

```
debian@beaglebone:~/i2c-gyro$ ./i2cgyro
I am 0xd3
x y z = -133 -36 -46
x y z = 37 0 61
x y z = 78 62 -6
x y z = 45 4 22
x y z = -37 31 2
x y z = 34 31 3
x y z = -325 121 -9
x y z = 273 -94 -86
x y z = -196 6169 -200
x y z = 993 -6525 -328
x y z = -561 5519 138
x y z = 167 -6150 143
x y z = 0 39 8
x y z = 28 -32 30
x y z = 113 21 -1274
x y z = -105 -11 8005
x y z = 14 39 5201
x y z = 21 -5 1488
x y z = -22 -51 -3009
x y z = 304 -24 -8167
debian@beaglebone:~/i2c-gyro$
```

ပုံ ၅.၆: Gyroscope ဝင်ရှိးများ၏ ပြောင်းလဲမှုတန်ဖိုးများ ကိုဖတ်ခြင်း။

၅.J SPI

SPI လို့ ခေါ်တဲ့ serial peripheral interface bus က IC chip တွေ အကွာအဝေး နီးနီးနားနား ဆက်သွယ်ဖို့ သုံးနိုင် တဲ့ interface သတ်မှတ်ချက် တစ်ခုပါ။ သူမှာ transmit လုပ်ဖို့ ဝါယာ တစ်လိုင်း receive လုပ်ဖို့ ဝါယာ တစ်လိုင်း သပ်သပ်စီ ဖြစ်တဲ့ အတွက် full duplex communication ဖြစ်ပါတယ်။ Master က slave တွေနဲ့ ဆက်သွယ်နိုင် တဲ့ master slave ပုံစံ ဖြစ်ပါတယ်။ ပုံမှန် အားဖြင့် ဝါယာ လေးခု သုံးပြီး၊ အောက်ပါ အတိုင်း ဖြစ်ပါတယ်။

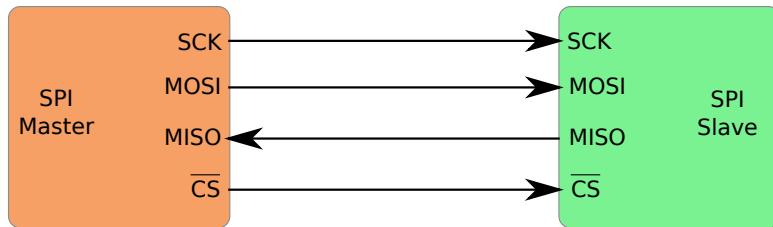
SCK: Serial clock (master က ထုတ်ပေးပါတယ်။)

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 သူဆက်သွယ်ချင်တဲ့ slave ကို enable လုပ်ဖို့ ထုတ်ပေးတဲ့ active low signal ဖြစ်ပါတယ်။)

SPI bus တစ်ခု ရဲ့ နမူနာ ဆက်သွယ်မှု တစ်ခု ကို ပုံ ၅.၇ မှာ ပြထား ပါတယ် [Wik18b]။



ပုံ ၅.၇: SPI bus အတွက် master နှင့် slave ဆက်သွယ်ခြင်း။

ဒေတာ တွေ ပေးပို့ ဆက်သွယ် တဲ့ အခါ clock ရဲ့ polarity (CPOL) နဲ့ clock ရဲ့ phase (CPHA) ပေါ်မူတည်ပြီး mode လေးမျိုး ရှိပါတယ် (လေား ၅.၂, လေား ၅.၃, လေား ၅.၄, လေား ၅.၅)။

လေား ၅.၂: 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 တွင်ပတ်၍ trailing (falling) edge တွင် ပြောင်းသည်။ |

လေား ၅.၃: 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 ပြောင်းခြုံ၊ trailing (falling) edge တွင် ဖတ်သည်။ |

ပေါ်ပြုချက်

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

ပေါ်ပြုချက်

| Mode | CPOL | CPHA | ဖော်ပြုချက် |
|------|------|------|--|
| 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 ပြောင်း၍ trailing (rising) edge တွင် ဖတ်သည်။ |

BeagleBone Black မှာ SPI0 နဲ့ SPI1 ကို header pin တွေ ပေါ်ကနေ သုံးလို ရပါတယ်။ SPI1 ရဲ့ ပင်တခိုက် က HDMI နဲ့ တိုက်နေတဲ့ အတွက် သူကို သုံးဖို ဆိုရင် HDMI ကို disable လုပ်ဖို လိုပါတယ်။ BeagleBone Black ရဲ့ SPI ပင်တွေ ကို ပေါ်ပြုချက် ပါတယ်။

ပေါ်ပြုချက်

ပေါ်ပြုချက်

| Bus | Wire | Header | Device | ဖော်ပြုချက် |
|------|------|--------|-----------|---|
| SPI0 | CS | P9_17 | spi0_cs0 | ပုံမှန်အားဖြင့် disabled ဖြစ်နေပြီး၊ enable လုပ်၍ သုံးနိုင် သည် |
| | MOSI | P9_18 | spi0_d1 | |
| | MISO | P9_21 | spi0_d0 | |
| | SCLK | P9_22 | spi0_sclk | |
| SPI1 | CS | P9_28 | spi1_cs0 | Video က အသုံးပြု ထားသည်။ အသုံးပြု ရန် Video ကို disable လုပ်ရန်လိုပြီး၊ SPI1 ကို enable ပြုလုပ်ရန်လိုသည်။ |
| | MOSI | P9_ | spi1_d1 | |
| | MISO | P9_29 | spi1_d0 | |
| | SCLK | P9_31 | spi1_sclk | |

SPI0 ကို enable လုပ်ဖို့ အတွက် အောက်က command တွေကို သုံးပြီး firmware library တွေကို list လုပ်ကြည့်ပြီး /boot/uEnv.txt ကို edit လုပ်နိုင် ပါတယ်။ အဲဒီ ဖိုင်ထဲမှာ BB-SPIDEV0-00A0.dtbo ကို စာရင်း ၅.၆ မှာ ပြထား သလိုမျိုး သတ်မှတ် ပြီးတဲ့ အခါ reboot လုပ်ဖို့လို ပါတယ်။

```
$ ls /lib/firmware | grep SPI
$ sudo nano /boot/uEnv.txt
```

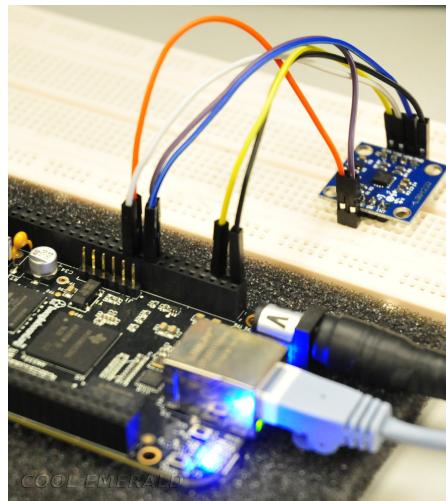
```
1 uboot_overlay_addr3=/lib/firmware/BB-SPIDEV0-00A0.dtbo
```

စာရင်း ၅.၆: SPI0 ကို /boot/uEnv.txt တွင် enable လုပ်ခြင်း။

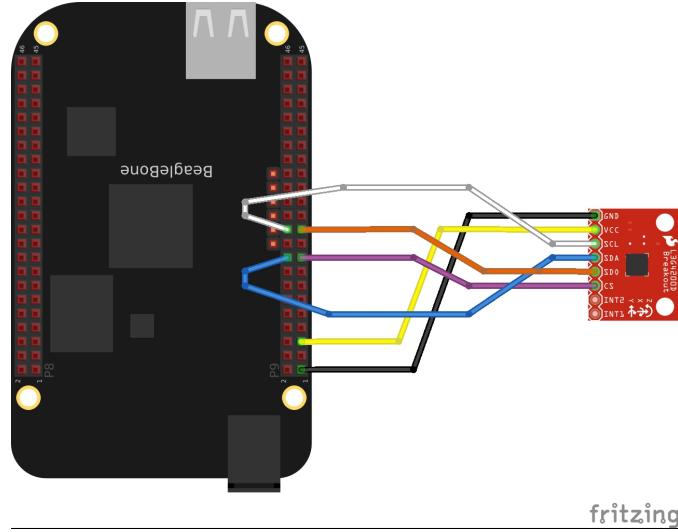
အရင် firmware အဟောင်း တွေ အတွက် တော့ နောက်ဆက်တဲ့ A အပိုင်း C.၆ မှာ ဆွဲးနွေး ထား သလို အသုံးပြု နိုင် ပါတယ်။

၅.၂.၁ Gyroscope ကိုအသုံးပြုခြင်း

SPI0 ကို သုံးပြီး L3G4200D gyroscope ကို အသုံးပြုတဲ့ C++ နမူနာကို စာရင်း ၅.၇,၅.၈ , ၅.၉ တို့မှာ တွေ့နိုင် ပါတယ် [Mol14; Eli17c; Vor07]။ အဲဒီ အတွက် ပစ္စည်းများ ဆက်သွယ်ပုံ ကို ပုံ ၅.၈ နဲ့ ပုံ ၅.၉ မှာ တွေ့နိုင် ပါတယ်။ Module ရဲ့ SDA က SPI mode မှာ SDI ဖြစ်ပါတယ်။ အဲဒီ လိုပဲ SCL က SCLK နဲ့ pin အတူတူ ဖြစ်ပါတယ်။



ပုံ ၅.၈: L3G4200 Gyroscope module ကို SPI နှင့် ဆက်သွယ်ခြင်း။

fritzing

ပုံ ၅.၉: SPI0 နဲ့ Gyroscope သင်္ကေတပါ။

```

1 #ifndef CESPI_H
2 #define CESPI_H
3
4 #include <stdint.h>
5 #include <unistd.h>
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <getopt.h>
9 #include <fcntl.h>
10 #include <sys/ioctl.h>
11 #include <linux/types.h>
12 #include <linux/spi/spidev.h>
13 #include <string>
14 using namespace std;
15 class cespi
16 {
17     public:
18         cespi();
19         virtual ~cespi();
20         bool Begin();
21         void Transfer(char* tx, char* rx, unsigned int n);

```

၁၂. SPI

၁၂၂

```
22     private:  
23         int fd;  
24         unsigned char mode;  
25         unsigned char bits;  
26         int speed;  
27         int delay;  
28         string spidev;  
29     };  
30  
31 #endif // CESPI_H
```

၁၃၈။ ၁၃၉။ cespi.h

```
1 #include "cespi.h"  
2 #define PRINT_MES 0  
3 cespi::cespi()  
4 {  
5     mode=0;  
6     bits=8;  
7     speed=500000; //500 kHz  
8     delay=0;  
9     spidev="/dev/spidev1.0";  
10 }  
11  
12 cespi::~cespi()  
13 {  
14     close(fd);  
15 }  
16  
17 bool cespi::Begin()  
18 {  
19     fd = open(spidev.c_str(), O_RDWR);  
20     if (fd < 0){  
21         printf("Can't open %s.\n", spidev.c_str());  
22         return false;
```

```

23 }
24
25     if(ioctl(fd, SPI_IOC_WR_MODE, &mode)==-1){
26         printf("Can't set SPI mode.\n");
27         return false;
28     }
29
30     if(ioctl(fd, SPI_IOC_RD_MODE, &mode)==-1){
31         printf("Can't get SPI mode.\n");
32         return false;
33     }
34
35     if(ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits)==-1){
36         printf("Can't set bits per word.\n");
37         return false;
38     }
39
40     if(ioctl(fd, SPI_IOC_RD_BITS_PER_WORD, &bits)==-1){
41         printf("Can't get bits per word.\n");
42         return false;
43     }
44
45     if(ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed)==-1){
46         printf("Can't set max speed hz.\n");
47         return false;
48     }
49
50     if(ioctl(fd, SPI_IOC_RD_MAX_SPEED_HZ, &speed)==-1){
51         printf("Can't get max speed hz.\n");
52         return false;
53     }
54
55     printf("SPI mode: %d\n", mode);
56     printf("Bits per word: %d\n", bits);
57     printf("Max speed: %d Hz\n", speed);
58     return true;

```

```

59 }
60
61 void cespi::Transfer(char* tx, char* rx, unsigned int n)
62 {
63     struct spi_ioc_transfer tr;
64     tr.tx_buf = (unsigned long)tx;
65     tr.rx_buf = (unsigned long)rx;
66     tr.len = n;
67     tr.delay_usecs = delay;
68     tr.speed_hz = speed;
69     tr.bits_per_word = bits;
70
71     if(ioctl(fd, SPI_IOC_MESSAGE(1), &tr)<1){
72         printf("Can't send SPI message.\n");
73     }
74 }
```

စာရင်း ၅.၈: cespi.cpp

```

1 #include<stdio.h>
2 #include "cespi.h"
3 #include "ceutil.h"
4 using namespace std;
5 int main()
6 {
7     cespi gyro;
8     gyro.Begin();
9
10    char t[7]={0,0,0,0,0,0,0};
11    char r[7]={0,0,0,0,0,0,0};
12    //b7 = set (1) for reading
13    //b6 = cleared (0) not to auto increase address
14    //b5-b0 = register address
15
16    //read the "who am i" register at address 0x0F
```

```

17 //its value should be 0xD3
18 t[0]=0x8F; //read, not auto 0x0F / 0x80
19 gyro.Transfer(t,r,2);
20 printf("I am 0x%02x\n",r[1]);
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 /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 t[0]=0x21;
29 t[1]=0x09;
30 gyro.Transfer(t,r,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 /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 /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 t[0]=0x23;
45 t[1]=0x80;
46 gyro.Transfer(t,r,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 /0 /

```

```

49 //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/DRO/BW1/BW0/PD/Zen/Yen/Xen /
57 //Default = 10 10 10 10 10 11 11 11 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 t[0]=0x20;
64 t[1]=0xEF;
65 gyro.Transfer(t,r,2);
66
67 t[0]=0xE0;//read ctrl registers
68 gyro.Transfer(t,r,5);//send addr 1 byte + read 4 byte
69 printf("reg = %02x%02x %02x%02x\n",r[1],r[2],r[3],r[4]);
70
71 int x,y,z;
72
73 for(int i=0;i<10;i++)//with T=0.1 for 30 sec
74 {
75     //read address 0x28, OUT_X_L register
76     //set MSB bit for auto address increment
77     t[0]=0xE8;
78     gyro.Transfer(t,r,7);
79     x=((int(r[2])<<8)+r[1])|(r[2]&0x80?0xFFFF0000:0x00000000);
80     y=((int(r[4])<<8)+r[3])|(r[4]&0x80?0xFFFF0000:0x00000000);
81     z=((int(r[6])<<8)+r[5])|(r[6]&0x80?0xFFFF0000:0x00000000);
82     //Print results
83     //printf("x y z = %02x%02x %02x%02x %02x%02x\n",d[1],d[0],d[3],d[2],d

```

```

[5], d[4]);
84     printf("x y z = %d %d %d\n", x, y, z);
85     usleep(500000);
86 }
87
88 return 0;
89 }
```

စာရင်း ၅.၆: spigo.py

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.
- [Eli17c] Elinux. RPi SPI. 2017. url: https://elinux.org/RPi_SPI.
- [Mol14] Derek Molloy. Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. Wiley, 2014. isbn: 1118935128. url: <http://www.exploringbeaglebone.com/>.
- [Vor07] Anton Vorontsov. SPI testing utility (using spidev driver). 2007. url: https://raw.githubusercontent.com/torvalds/linux/master/tools/spi/spidev_test.c.
- [Wik18a] Wikipedia. I2C. 2018. url: <https://en.wikipedia.org/wiki/I%C2%BCC>.
- [Wik18b] Wikipedia. Serial Peripheral Interface Bus. 2018. url: 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 ၆

စက်အမြင်

သင့်၏ စက်တွင် ကင်မရာ တပ်ဆင်ပြီး ဓာတ်ပုံ၊ ဗိုဒ္ဓိယို ရိုက်ကူး ခြင်းများ၊ ပုံရိုပ်ပြုစပ်ခြင်း (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 မြန်ဆန့်စွာ တွက်ချက်မှု နှင့် အချိန် နှင့် အမျှ အရေးကြီး သော real-time အသုံးချမှု များ အတွက် အမိက ထားကာ ဒီဇိုင်း ထုတေသန ပြုလုပ် ထားသည်။ C/C++ သုံး၍ ရေးထား သဖြင့် multi-core processing များ၏ ကောင်းကွက် ကိုလည်း အသုံးချ နိုင်သည်။ OpenCV ကိုအနုပညာ လုပ်ငန်း များမှ အစ၊ မိုင်းရှာခြင်း၊ ပြေပုံများ ချိတ်ဆက်ခြင်း၊ အဆင့်မြှင့် စက်ရပ်များ ဖန်တီးခြင်း အထိ နယ်ပယ် အမျိုးမျိုး တွင် ကျယ်ကျယ် ပြန်ပြန့် အသုံးပြု နေဖြေ သည်။

၆.၁ နောက်ခံအကြောင်းအရာ

OpenCV ၏ အဓိပ္ပာယ် မှာ Open Source Computer Vision Library ဖြစ်၍ မူရင်းကုဒ် များအား ဖွင့်ပြ ထားသည်။ ဂွန်ပျူးတာ နှင့် စက်အမြင် ဆိုင်ရာ လုပ်ငန်း များတွင် ကိုးကား အသုံးပြု စရာ ဆော့ဖို့ library ဖြစ်သည်။ OpenCV ကို စတင် ဖန်တီးစဉ် မှစ၍ ဂွန်ပျူးတာ နှင့် စက်အမြင် ဆိုင်ရာ အများ သူငါ အလွယ် တကူ အသုံးပြု နိုင်သည့် ဘုံယန်ရား 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 ရစေရန် ချိန်ညီ ထားသော နည်းလမ်း (algorithm) ပေါင်း ၂၅၀၀ ကျော် ရှိသည်။ နည်းဟောင်းများ မှစ၍ နောက်ဆုံးပေါ် နည်းလမ်းသစ် များအထိ ပြည့်စုံနှစ်စွာ ပါဝင် သည်။ ထို နည်းလမ်း များကို မျက်နှာ များကို ရှာဖွေ သိရှိခြင်း၊ မှတ်မိခြင်း (face detection and recognition)၊ အရာဝတ္ထု များကို ခွဲခြား သိရှိခြင်း (object identification)၊ ဗိုဒ်ယို အတွင်းမှ လူ အမူ အရာ များကို ခွဲခြား သတ်မှတ်ခြင်း (classification of human actions)၊ ရွှေလျား နေသော အရာ ဝတ္ထု များနောက် လိုက်ခြင်း (tracking moving objects)၊ သုံးဘက်မြင် မော်ဒယ် ထုတ်ခြင်း (3D modeling)၊ ရုပ်ပုံ များကို ဆက်ခြင်း (stiching)၊ သိမ်းထား သည့် ပုံများ ထဲမှ တူသော ပုံကို ရွေးထုတ်ခြင်း၊ ရွှေခံး များကို မှတ်မိခြင်း၊ ဖြည့်စွက် အာရုံရိပ် (augmented reality) များ အတွက် အမှတ် အသား များ လုပ်ခြင်း တို့တွင် အသုံးပြု နိုင်သည်။

၆.J ဖွဲ့စည်းပုံ

OpenCV ကို အပိုင်း (module) များ ခွဲခြား တည်ဆောက် ထားသည်။ ထို့ကြောင့် များစွာသော shared သို့မဟုတ် static libraries များ ပါဝင် နေသည်။ အဓိက 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 ကျေစံလစ် သည့် အခြေခံ ကျသော အပိုင်း ဖြစ်သည်။

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

video ဗိုဒ်ယို များကို ခွဲခြား သုံးသပ်၍ ရွှေလျားမှု ကို ခန်းမှန်း ခြင်းများ၊ နောက်ခံ မြင်ကွင်း ကို ဖယ်ဖျောက် ခြင်းများ၊ အရာ ဝတ္ထု 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 ရှာခြင်း နှင့် descriptor matcher များ ပါဝင်သည်။

objdetect သတ်မှတ် ထားသော အရာ ဝို့ များကို ရှာခြင်း (ဥပမာ မျက်နှာ (face detection))၊ မျက်လုံး၊ မတ်ခွက်၊ လူ၊ ကား၊ စသည် များ)။

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

Video I/O ဗိုဒ္ဓယို များ ဖမ်းယူခြင်း၊ 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။

၆.၃ တပ်ဆင်ခြင်း

၆.၃.၁ 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, libjasper-dev, libdc1394-22-dev

ထို packages များအား စက်ထဲသို့ ထည့်သွင်း လိပ်ကြ Synaptic Manager သုံး၍ သော်လည်းကောင်း၊ terminal တွင် အောက်ရှိ စာရင်း ၆.၁ နှင့် ၆.၂ ပါ command များ ရိုက်နှုပ်၍ သော်လည်းကောင်း 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 ၆.၁: 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 ၆.၂: OpenCV အတွက် optional packages များ ရယူခြင်း။

၆.၃.၂ 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
```

၆.၃.၃ Source မှ build လုပ်ခြင်း

ထိုသို့ မဟုတ်ပဲ လက်ရှိ OpenCV အခြေကျ ဗာရှင်း 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 ဗားရှင်း ကို မယူပဲ နောက်ဆုံးထွက် cutting-edge opencv ဗားရှင်း 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 ၆.၃ ရှိ command များ ရှိကြမည်။

```
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 WITH_OPENCL=OFF \
7 -D WITH_OPENCL_SVM=OFF \
8 -D WITH_OPENCLAMDFFT=OFF \
9 -D WITH_OPENCLAMDBLAS=OFF \
10 -D WITH_VA_INTEL=OFF \
11 -D BUILD_opencv_python2=OFF \
12 -D BUILD_opencv_python3=OFF \
13 -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules ..
```

စာရင်း ၆.၃: OpenCV ကို build လုပ်ခြင်း။

Shared libs ကို unset လုပ်ချင် ပါက စာရင်း ၆.၄ ရှိ option ကို ထည့်နိုင် သည်။

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

စာရင်း ၆.၄: Static libs ပြုလုပ်ခြင်း။

အောင်ဆိုင်း ပြီးသည့် အခါ စာရင်း ၆.၅ အတိုင်း build လုပ်၍ install လုပ်မည်။ တပ်ဆင် ပြီးသည့် OpenCV ဗားရှင်း ကို စာရင်း ၆.၆ တွင် ပြထား သော command ဖြင့် ကြည့် နိုင်သည်။

```
1 $ make
2 $ sudo make install
```

စာရင်း ၆.၅: OpenCV ကို install လုပ်ခြင်း။

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

စာရင်း ၆.၆: OpenCV ဗားရှင်းကို စစ်ခြင်း။

၆.၃.၄ GCC ၊ CMake တိုဖြင့် အသုံးပြုခြင်း

OpenCV ကို သုံးရန် အလွယ်ဆုံး နည်းမှာ CMake ဖြင့် သုံးခြင်း ဖြစ်သည် [Ope17e]။ CMake နှင့် မရင်းနှီး ပါက CMake tutorial (<https://cmake.org/cmake-tutorial/>) တွင် သွား ရောက် လေ့လာ 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 ၆.၇ တွင် ဖော်ပြ ထားသည့် DisplayImage.cpp ဆိုသည့် ပရိုဂရမ် လေးအား ဖန်တီး လိုက်မည်။ ပြီးသည့် အခါ DisplayImage ဟူသည့် folder တစ်ခု ဖွဲ့၍ ထိုင် တွင် သိမ်းဆည်း လိုက်မည်။

```
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( "/home/debian/DisplayImage/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 ၆.၃: ပုံ တစ်ပုံ ကို ဖတ်၍ ပြသည့် DisplayImage ပရီဂရမ်။

ဤနေရာ တွင် thiri.jpg အတွက် path မှာ ”/home/yan/thiri.jpg” ဖြစ်ပြီး သင့်ပုံ ရှိသည့် folder ၏ username တို့နှင့် 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 ရှိက်ပြ၍ ထွက်သွားမည် ဖြစ်ပြီး၊ ဖတ်၍ အောင်မြင် ပါက imshow ကိုသုံး၍ ပုံကို ထုတ်ပြမည် ဖြစ်သည်။ တဖန် CMakeLists.txt ဆိုသည့် ဖိုင်ကို အောက်ပါ စာရင်း ၆.၈ အတိုင်း ဖန်တီးမည်။

```

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

စာရင်း ၆.၈: CMakeLists.txt

ဤတွင် လိုချင်သည့် ပရီဂရမ် ကို အောက်ပါ စာရင်း ၆.၉ ရှိ command များအား terminal တွင် ပိုက်ခြင်းဖြင့် ထုတ်လုပ် ရရှိနိုင်၊ run ကြည့်နိုင် သည်။

```

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

စာရင်း ၆.၉: DisplayImage ကို CMake ဖြင့် build လုပ်၍ run ခြင်း။

GUI application ဖြစ်သည့် အတွက် tightvncserver ကို သုံး၍ run ဖိုလို သည်။ မူလပါရှိသော Terminal မှာ အဆင် မပြောသည့် အတွက် xterm အား အောက်ပါ အတိုင်း တပ်ဆင် နိုင် သည်။

```
sudo apt-get install xterm
```

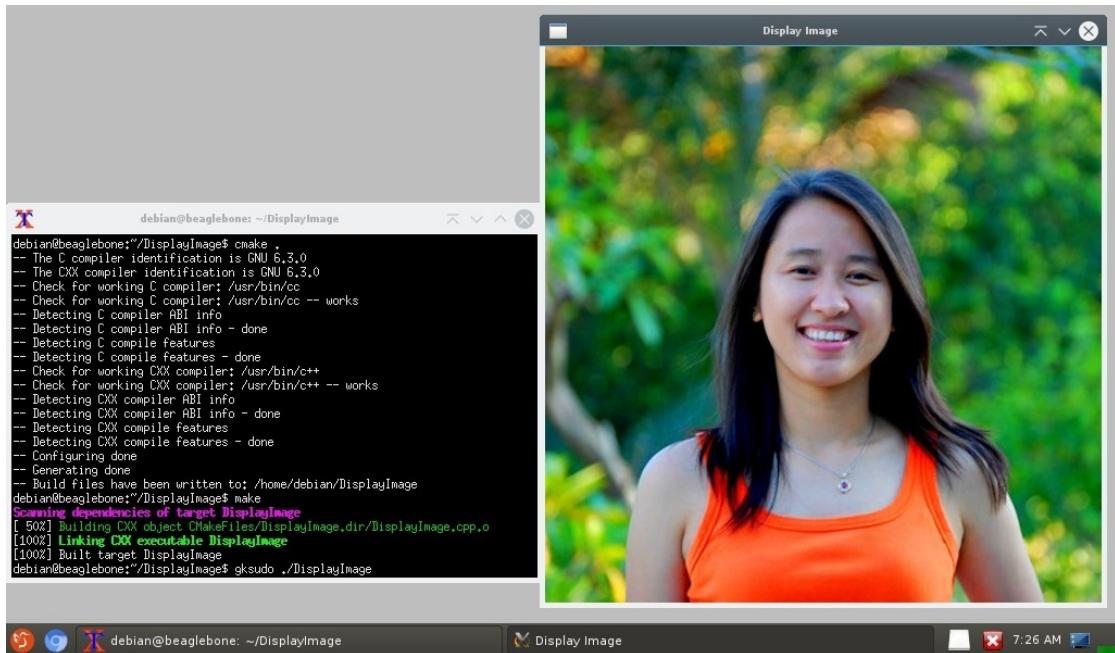
pkg-config

CMake ကို မသုံးပဲ pkg-config ဖြင့် စာရင်း ၆.၁၀ အတိုင်း build လုပ်၍ လည်း run နိုင်သည်။

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

စာရင်း ၆.၁၀: DisplayImage ကို g++, pkg-config တို့ဖြင့် build လုပ်၍ run ခြင်း။

ထိုနောက် xterm တွင် ရလာသော binary ကို run လိုက်သည့် အခါ ပုံ ၆.၁ အတိုင်း ရရှိမည်။



ပုံ ၆.၁: DisplayImage ပရိဂရမ် ၏ ရလာဒ် ကို terminal နှင့် ယူဉ်တဲ့ ပြထား သည်။

Shared lib အား ရာ မရ သည့် error ရဲ့ လျှင် /etc/ld.so.conf.d/opencv.conf ဟူသည့် ဖိုင်အား

အောက်ပါ အတိုင်း ဖန်တီး နှင့်သည်။

```
$ 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။ ထိုနောက် အောက်ပါ command အား terminal တွင် ရိုက်ထည့် နှင့်သည်။

```
$ sudo ldconfig
```

၆.၅ ကင်မရာကိုသုံးခြင်း

OpenCV တွင် ဗိုလ်ချုပ်များကို ဖမ်းယူရန် သို့မဟုတ် ဖတ်ရန် VideoCapture ဟုခေါ်သည့် class ပါရှိသည် [Ope16]။ ဗိုလ်ချုပ်များ ရေးသားရန်အတွက်ကို မူ VideoWriter Class ကိုသုံးနှင့်သည် [Ope17f]။ ဗိုလ်ချုပ် တစ်ခု ကို ပုံရှိပ် များအား အစီအစဉ် လိုက် ပေါင်းစပ် ပြုလုပ် ထားသည် ဟု ဆိုနိုင် သည်။ ထိုသို့ ဖွဲ့စည်း ထားသည့် ပုံရှိပ် တစ်ခု ခြင်းစီ ကို ပြောက် (frame) ဟု ခေါ်ပြီး၊ တစ် စက်နှုန်း အတွင်း ပြောင်းလဲ သွားသည့် ပြောက် အရေ အတွက် ကို ပြနှစ်း (frame rate) ဟု ခေါ်မည် [Lag14]။

နမူနာ အနေနှင့် USB Webcam တစ်ခု ကို စက်တွင် တပ်ဆင်၍၊ ထို ကင်မရာ မှ ပုံရှိပ် များကို ဖမ်းယူ ၍ ပြုသ ကာ၊ ဗိုလ်ချုပ်များ အနေ နှင့်လည်း သိမ်းဆည်း ပေးသည့် ပရိုဂရမ တစ်ခု ကို ဖော်ပြုမည်။ ဤ နမူနာ တွင် Logitech C922 Pro Stream Webcam (ပုံ ၆.၂) ကို အသုံး ပြုထား ပြီး အခြား အဆင်ပြု ရာ webcam များအား လည်း အသုံးပြု နှင့် သည်။ Video capture ပြုလုပ် သည့် ပရိုဂရမ vcap.cpp ကို စာရင်း ၆.၁၁ တွင် ဖော်ပြ ထားသည်။



ပုံ ၆.၂: Logitech C922 Pro Stream Webcam

```

1 #include <stdio.h>
2 #include <iostream>
3 #include <opencv2/opencv.hpp>
4 //#include <string>
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("/home/debian/vcap/sensor.mp4"); //open video file
12
13     if (!cap.isOpened()) {
14         printf("Video is not opened. \n");
15         return -1;
16     }
17     else {
18         printf("Video is opened. \n");

```

```

19 }
20
21     union { int v; char c[5]; } uEx;
22     uEx.v = static_cast<int>(cap.get(CV_CAP_PROP_FOURCC));
23     uEx.c[4] = '\0';
24     printf("Codec: %s \n",uEx.c);
25
26     Size S = Size((int)cap.get(CV_CAP_PROP_FRAME_WIDTH),(int)cap.get(
27         CV_CAP_PROP_FRAME_HEIGHT));
28     printf("Frame size: %d x %d \n",S.width,S.height);
29
30     double rate = cap.get(CV_CAP_PROP_FPS); //Frame rate
31     printf("Frame rate: %f \n", rate);
32     int dperiod = 1000 / rate;
33
34     int ex = CV_FOURCC('M', 'J', 'P', 'G');
35     //int ex = CV_FOURCC('X', 'V', 'I', 'D');//https://www.xvid.com/
36     //int ex = -1;//pop up window to choose
37     rate = 30;
38     const string vpath="/home/debian/vcap/vdowr.avi";
39     VideoWriter outputVideo(vpath, ex , rate, S, true);
40     if (!outputVideo.isOpened())
41     {
42         cout << "Could not open the output video to write."<< endl;
43         waitKey(5000);
44         return -1;
45     }
46
47     Mat frame;
48     for (int i = 0;; i++) {
49         if (!cap.read(frame)) break;
50         outputVideo << frame;
51         imshow("Frame", frame);
52         if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
53     }

```

```

54     cap.release();
55     outputVideo.release();
56     //waitKey(5000);
57     return 0;
58 }
```

စာရင်း ၆.၁၁: Video capturing

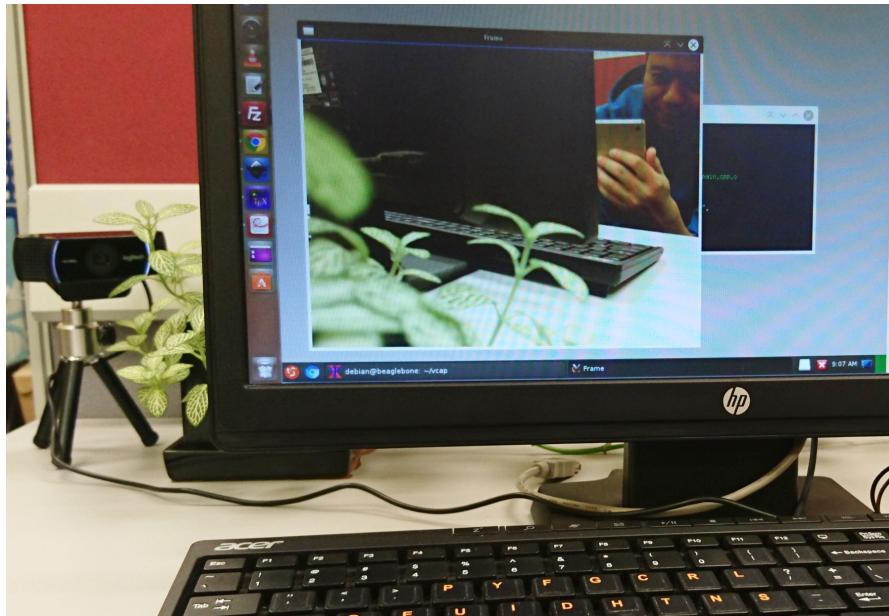
ပရိုဂရမ် အစ ရှိ VideoCapture အတွက် object တစ်ခု ကို initialize ပြုလုပ် ရာတွင် ဖွံ့ဖြိုးပို့ ဖိုင်တစ်ခု ၏ path ကို ထည့်ပေး ပါက ဖိုင်ကို ဖွင့်ဖတ် ပေးမည့် ဖြစ်ပြီး၊ ၀ ကို အသုံးပြု ပါက စက်၏ default ကင်မရာ ကို ဖတ် မည် ဖြစ်သည်။ ထိုနောက် ဖွံ့ဖြိုးပို့ ဖွင့်ခြင်း အဆင်ပြော မပြု မပြု isOpened ဆိုသည့် method ဖြင့် စစ်ဆေး နိုင်သည်။

လက်ရှိ ဖွံ့ဖြိုးပို့ ၏ setting များကို get ဟူသည့် method သုံးကာ ဖတ်ကြည့် နိုင်ပြီး၊ set ကို သုံး၍ ပြုပြင် နိုင်သည်။ ဖွံ့ဖြိုးပို့ ၏ Codec ကို get(CV_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 အချယ် အစား၊ ပြန်နှုန်း တို့ကို CV_CAP_PROP_FRAME_WIDTH , CV_CAP_PROP_FRAME_HEIGHT , CV_CAP_PROP_FPS တို့ ဖြင့် ဖတ်သည်။

ဆက်လက်၍ ကင်မရာ မှ ဖတ်ရှု ရရှိသော ပုံရိပ် များကို ဖွံ့ဖြိုးပို့ 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 CV_FOURCC('M', 'J', 'P', 'G') ဟု ရယူ သည်။ ထိုသို့ မဟုတ်ပဲ <https://www.xvid.com> တွင် free ရယူ နိုင် သော XVID စသည့် codec တို့ကို CV_FOURCC('X', 'V', 'I', 'D') အစ ရှိသဖြင့် သတ်မှတ် နိုင်သည်။ အကယ်၍ ထို argument အတွက် -1 ကို သုံးပါက GUI ဖြင့် ရွှေးချယ်ရန် pop-up window ပေါ်လာ မည်။

နောက်ပိုင်း loop တဲ့တွင် VideoCapture ၏ read ဖြင့် frame များကို တစ်ခု ခြင်း ဖတ်၍ VideoWriter ဖြင့် ရေးသည်။ waitKey သည် သတ်မှတ် ထားသည့် milliseconds အချိန် ရပ်စောင့်၍ ထည့်သွင်း ရှိက်ထည့် သည့် key တစ်ခု ကို စောင့်ဖတ် ပေးသည်။ ထိုသို့ ဖတ်နိုင် ရန် imshow ဖြင့် ပြထား သည့် ပြောက် ဝင်းဒိုး မှာ active ဖြစ်နေရန် လို သည်။ Escape key ၏ တန်ဖိုး 27 ကို ဖတ်၍ ရပါက loop မှ ထွက်၍ ပရိုဂရမ် အဆုံးသတ် သည်။ Webcam တစ်ခု ကို BeagleBone Black တွင်

opencv ဖြင့် ချိတ်ဆက် အသုံးပြု နေပါ ကို ပုံ ၆.၃ တွင် ဖော်ပြ ထားသည်။



ပုံ ၆.၃: Webcam ကို BeagleBone Black ဖြင့် ချိတ်ဆက် အသုံးပြုခြင်း။

၆.၅ Real-time Face Detection

ဗိုလ်ချုပ် အတွင်း မှ မျက်နှာ များအား အချိန် နှင့် တပြီးညီ ရှာဖွေခြင်း ကို ပြုလုပ် မည်။ ထိုအတွက် Cascade Classifier [Ope17a] ကို သုံးနိုင် သည်။ ဤ နမူနာ အတွက် OpenCV နောက်ဆုံး ဗားရှင်း ကို ထည့်သွင်း ထားရန် လိုပြီး eMMC တွင် storage မလောက် ပါက SD card ကို အပိုင်း ၃၃၂ တွင် ဖော်ပြ ထား သလို သုံးနိုင် သည်။ နမူနာ facedetection ပရီ ကရမဲ ကို အောက်ပါ စာရင်း ၆.၁ 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 = "/home/debian/facedetection/
haarcascade_frontalface_alt.xml";

```

```

8 string eyes_cascade_name = "/home/debian/facedetection/
9     haarcascade_eye_tree_eyeglasses.xml";
10 CascadeClassifier face_cascade;
11 CascadeClassifier eyes_cascade;
12
13 void detectAndDisplay(Mat frame)
14 {
15     std::vector<Rect> faces;
16     Mat frame_gray;
17
18     cvtColor(frame, frame_gray, CV_BGR2GRAY);
19     equalizeHist(frame_gray, frame_gray);
20
21     face_cascade.detectMultiScale(frame_gray, faces, 1.1, 2, 0 |
22         CV_HAAR_SCALE_IMAGE, Size(120, 120)); // Detect faces
23
24     for (size_t i = 0; i < faces.size(); i++)
25     {
26         Point center(faces[i].x + faces[i].width*0.5, faces[i].y + faces[i].
27             height*0.5);
28         ellipse(frame, center, Size(faces[i].width*0.5, faces[i].height*0.5), 0,
29             0, 360, Scalar(255, 0, 255), 4, 8, 0);
30
31         Mat faceROI = frame_gray(faces[i]);
32         std::vector<Rect> eyes;
33
34         eyes_cascade.detectMultiScale(faceROI, eyes, 1.1, 2, 0 |
35             CV_HAAR_SCALE_IMAGE, Size(30, 30)); // In each face, detect eyes
36
37         for (size_t j = 0; j < eyes.size(); j++)
38         {
39             Point center(faces[i].x + eyes[j].x + eyes[j].width*0.5, faces[i].y +
40                 eyes[j].y + eyes[j].height*0.5);
41             int radius = cvRound((eyes[j].width + eyes[j].height)*0.25);
42             circle(frame, center, radius, Scalar(255, 0, 0), 4, 8, 0);
43         }

```

```
40 }
41 imshow("Frame", frame);
42 }
43
44 int main(int argc, char** argv)
45 {
46     //Load the cascades
47     if (!face_cascade.load(face_cascade_name)) { printf("--(!)Error loading\n")
48         ; return -1; };
49     if (!eyes_cascade.load(eyes_cascade_name)) { printf("--(!)Error loading\n")
50         ; return -1; };
51
52     VideoCapture cap(0); //Default camera
53     if (!cap.isOpened()) { printf("Video is not opened. \n"); return -1; }
54     else { printf("Video is opened. \n"); }
55
56     union { int v; char c[5]; } uEx;
57     uEx.v = static_cast<int>(cap.get(CV_CAP_PROP_FOURCC));
58     uEx.c[4] = '\0';
59     printf("Codec: %s \n", uEx.c);
60
61 //    cap.set(CAP_PROP_FRAME_WIDTH, 640);
62 //    cap.set(CAP_PROP_FRAME_HEIGHT, 480);
63 //    cap.set(CAP_PROP_FPS, 30);
64 //    Size S = Size((int)cap.get(CV_CAP_PROP_FRAME_WIDTH), (int)cap.get(
65 //        CV_CAP_PROP_FRAME_HEIGHT));
66 //    printf("Frame size: %d x %d \n", S.width, S.height);
67
68
69     double rate = cap.get(CV_CAP_PROP_FPS); //Frame rate
70     printf("Frame rate: %f \n", rate);
71     int dperiod = 33;
72
73     Mat frame;
74     for (int i = 0;; i++) {
75         if (!cap.read(frame)) break;
76         detectAndDisplay(frame);
```

```

73     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
74 }
75 cap.release();
76 //waitKey(5000);
77 return 0;
78 }
```

စာရင်း ၆.၁၂: Real-time face-detection

ဤ ပရိုဂရမဲ့ တွင် မျက်နှာကို ရှာဖွေရန် “haarcascade_frontalface_alt.xml” နှင့် မျက်လုံး များအား ရှာဖွေရန် “haarcascade_eye_tree_eyeglasses.xml” ဆိုသည့် ဖိုင်များ အား သုံးပါမည်။ ထို ဖိုင်များ သည် “/home/debian/opencv/data/haarcascades” အောက်တွင် ရှိသည်။ မျက်နှာ နှင့် မျက်လုံးများ အတွက် CascadeClassifier object နှစ်ခုကို ကြော်ပြီး load ကိုသုံး၍ အဆိပါ ဖိုင် များအား ဖတ်သည်။

အခါန် နှင့် တပြေးညီ ဖိုဒ်ပို့ ရိုက်ရန် VideoCapture object ကို ကြော်သည့် အခါ ဖိုင် နာမည် အစား ၀ ကို သုံးသည့် အတွက် သင့် စက် တွင် တပ်ဆင် ထားသော ကင်မရာ ကို သုံးသည်။ VideoCapture ၏ get ကို သုံး၍ လက်ရှိ ဖိုဒ်ပို့ ၏ setting များကို ဖတ်နိုင်ပြီး၊ set ကို သုံး၍ လိုသလို ပြင်နိုင် သည်။ ဤ ပရိုဂရမဲ့ တွင် ပြောက် အချယ် အစား နှင့် ပြန်နှုန်းကို set နှင့် ပြင်လိုက် သည်။ ထိုနောက် ပြောက် တစ်ခုခြင်း ဖတ်၍ detectAndDisplay ဖန်ရှင်ကို သုံးကာ မျက်နှာ ကို ရှာ၍ ပိုင်းပြသည်။

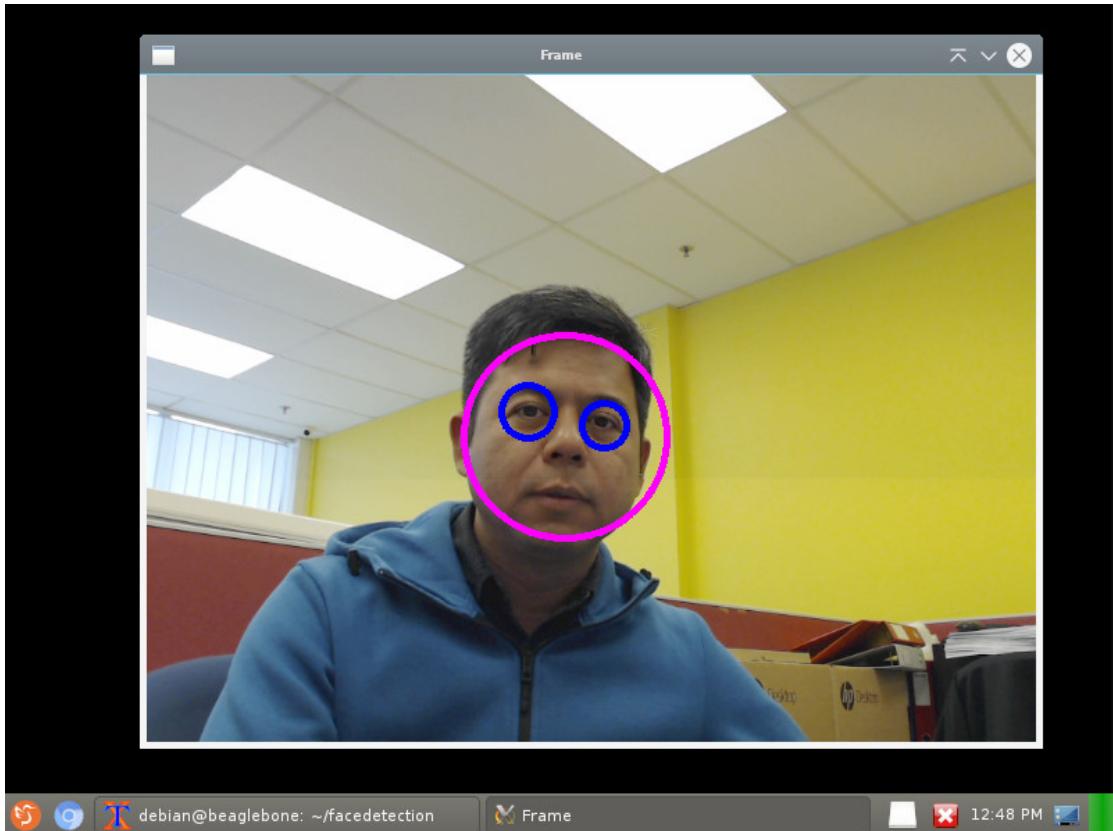
detectAndDisplay ဖန်ရှင် တွင် ပုံရှင်ပို့ အဖြူး အမည်း ပြောင်း၍ detectMultiScale ကို သုံး၍ ရှာသည်။ သူ၏ ပုံစံမှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

```

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())
```

ပထား Parameters ဖြစ်သည့် image မှာ ရှာလို သည့် ပုံရှင် ဖြစ်ပြီး၊ ဒုတိယ objects တွင် ရှာတွေ သည့် နေရာ များ၏ စတုဂံ များအား သိမ်းပေး မည်။ ထိုနောက် ရှာတွေသည့် နေရာ နှင့် အချယ် များကို သုံး၍ circle နှင့် စက်ဝိုင်း များ ဆွဲ၍ ပုံကို imshow နှင့် ပြပေး သည်။

အောက်ပါ ပုံ ၆.၄ တွင် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း နှင့် ရလာ သည့် ရလဒ် တိုကို ပြထား သည်။



ပုံ ၆.၄: တခြားနှင့် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း။

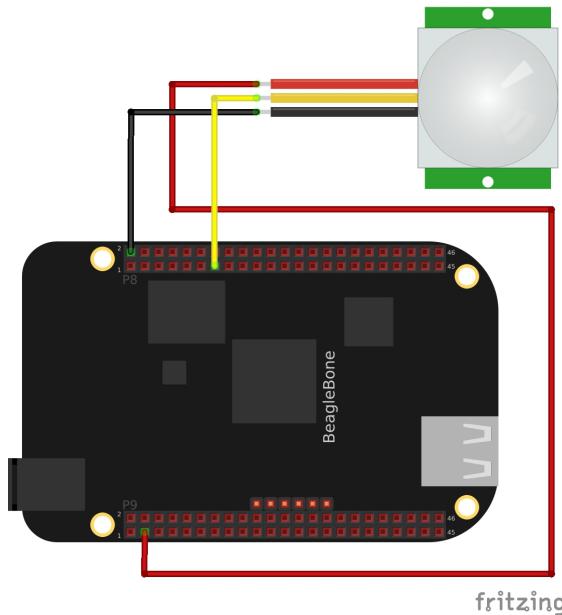
၆.၆ Smart Surveillance ကင်မရာပြုလုပ်ခြင်း

BBB ကို သုံးပြီး smart surveillance ကင်မရာ တစ်ခု ကို ကိုယ်တိုင် ပြုလုပ် ဖန်တီး ကြည့်ပါမယ်။ ပုံမှန် ဘာမှ ထူးခြားမှု မရှိတဲ့ အချိန်မှာ 5x speed နဲ့ နေ့စဉ်၊ ဒါမှ မဟုတ် နာရီ အလိုက် ဗိုဒ္ဓိယို ဖိုင်တွေ ခဲ့ပြီး record လုပ်ပေး နိုင် အောင် လုပ်ပါမယ်။ ဖမ်းယူ ရရှိတဲ့ ပုံရိုပ် တွေကို analysis လုပ်နေပြီး လူ ကိုယ်ခန္ဓာ တွေကို တွေ့တာ နဲ့ ပုံမှာ အစိမ်းရောင် စတုဂံ နဲ့ ဘောင်ခတ် ပြပြီး၊ ပုံမှန် speed နဲ့ ပြောင်းပြီး record လုပ် ပါမယ်။ Passive infrared sensor (PIR sensor) ကိုပါ သုံးပြီး sensor က လူရိုပ် ကို အာရုံခံ မိရင် လည်း၊ ပုံရိုပ် မှာ အနီးရောင် indicator ပြပြီး၊ ပုံမှန် နှုန်းနဲ့ ပြောင်း record လုပ်နိုင် ပါမယ်။

ဒီ နမူနာ ပရိုဂရမ် မှာ လုပ်လို ရတဲ့ အကြောင်း ကုတ် အကြမ်း ရေးပြ ရုံ ဖြစ်ပြီး၊ တစ်ခုခု ထူးခြားရင် ကိုယ့်ရဲ့ အီးမေးလ် ကို လှမ်းပို တာ၊ အီမ်က လူတွေ ရဲ့ မျက်နှာကို မှတ်မိတာ၊ သတ်မှတ် ထားတဲ့ အချိန် အပိုင်း အခြား အလိုက် လိုသလို record လုပ်တာ စတာ တွေကို စိတ်ကူး ရှိရင် ရှိသလို ထပ်ဖြည့် နိုင်

ပါတယ်။

ဈေးပေါ်တဲ့ HC-SR505 mini 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၊ BBB နဲ့ သုံးမှာ ဖြစ်လို 3.3 V ပါဝါပေး၊ sensor ရဲ့ output pin ကို GPIO23 နဲ့ ပုံ ၆.၅ မှာ ပြထား သလို ဆက်နိုင် ပါတယ်။



ပုံ ၆.၅: PIR sensor ကို ချိတ်ဆက်ခြင်း။

ပုံရိပ် ထဲမှာ Body detection လုပ်ဖို အတွက် တော့ အပိုင်း ၆.၅ မှာ face detection လုပ်သလို ပဲ Haar feature-based cascade classifiers ကို သုံးမှာ ဖြစ်လို အဲဒီ မှာ haarcascade_fullbody.xml ဆိုတဲ့ ဖိုင် နာမည် ပြောင်း လိုက်ရုံပါပဲ။ PIR sensor ဆက်ထား တဲ့ input ကို ဖတ်ပြီး၊ active ဖြစ်နေရင် PIR DETECTED ဆိုတဲ့ စာကို puttext ဖန်ရှင် သုံးပြီး အနီရောင် နဲ့ ဖော်ပြ ပါမယ်။ နမူနာ Smart-Cam.cpp ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၆.၁၃ 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 #include <stdio.h>

```

```

8 #include <opencv2/opencv.hpp>
9 #include <string>
10 #include<time.h>
11 #include "ce_io.h"
12
13 using namespace std;
14 using namespace cv;
15
16 #define CE_DAILY 3
17 #define CE_HOURLY 2
18 #define CE_MINUTELY 1
19 #define CE_MAKEVIDEO CE_MINUTELY
20 #define CE_PERIOD_NORMAL 50
21 #define CE_PERIOD_FAST 500
22
23 const string wpath="/home/debian/SmartCam/";
24 string body_cascade_name = wpath+"haarcascade_fullbody.xml";
25 //string body_cascade_name = wpath+"haarcascade_frontalface_alt.xml";
26 CascadeClassifier body_cascade;
27
28 int detectAndDisplay(Mat& frame)
29 {
30     std::vector<Rect> bodies;
31     Mat frame_gray;
32     int r=0;
33
34     cvtColor(frame, frame_gray, CV_BGR2GRAY);
35     equalizeHist(frame_gray, frame_gray);
36
37     body_cascade.detectMultiScale(frame_gray, bodies, 1.1, 2, 0 |
38         CV_HAAR_SCALE_IMAGE, Size(100, 100));// Detect bodies
39
40     for (size_t i = 0; i < bodies.size(); i++)
41     {
42         Point p1(bodies[i].x, bodies[i].y);
43         Point p2(bodies[i].x + bodies[i].width, bodies[i].y + bodies[i].
```

```

        height);
44     rectangle(frame, p1, p2, Scalar(0, 255, 0), 4, 8, 0);
45     r=1;
46 }
47 return r;
48 }

49

50 string GetTimeStr()
51 {
52     time_t t = time(0); //now
53     struct tm * ts=localtime(&t);
54     string str="";
55     string tstr="";
56     tstr=to_string(ts->tm_year+1900);
57     str+=string(4-tstr.length(), '0')+tstr+"-";
58     tstr=to_string(ts->tm_mon+1);
59     str+=string(2-tstr.length(), '0')+tstr+"-";
60     tstr=to_string(ts->tm_mday);
61     str+=string(2-tstr.length(), '0')+tstr;
62     #if (CE_MAKIDEO < CE_DAILY)
63         str+=" ";
64         tstr=to_string(ts->tm_hour);
65         str+=string(2-tstr.length(), '0')+tstr;
66         #if (CE_MAKIDEO < CE_HOURLY)
67             str+=":";
68             tstr=to_string(ts->tm_min);
69             //str+=string(2-tstr.length(), '0')+tstr+":";;
70             //tstr=to_string(ts->tm_sec);
71             str+=string(2-tstr.length(), '0')+tstr;
72         #endif // MAKEVIDEO
73     #endif // MAKEVIDEO
74     return str;
75 }

76

77 int main(int argc, char** argv)
78 {

```

፭.፭. SMART SURVEILLANCE ጽኑዎችበትልዕና

፩፲፭

```
79     printf("SmartCam ... \n");
80     printf("Select the image frame and press ESC to exit.\n");
81
82     //init IO
83     CE_IO pir_sensor(23, INPUT);
84
85     //Load the cascades
86     if (!body_cascade.load(body_cascade_name)) { printf("Error in loading
87         haarcascade.\n"); return -1; }
88
89     VideoCapture cap(0); //Default camera
90     if (!cap.isOpened())
91         { printf("Video is not opened. \n"); return -1; }
92     else
93         { printf("Video is opened. \n"); }
94
94     union { int v; char c[5]; } uEx;
95     uEx.v = static_cast<int>(cap.get(CV_CAP_PROP_FOURCC));
96     uEx.c[4] = '\0';
97     printf("Codec: %s \n", uEx.c);
98
99     Size S = Size((int)cap.get(CV_CAP_PROP_FRAME_WIDTH), (int)cap.get(
100        CV_CAP_PROP_FRAME_HEIGHT));
100    printf("Frame size: %d x %d \n", S.width, S.height);
101
102    double rate = cap.get(CV_CAP_PROP_FPS); //Frame rate
103    printf("Frame rate: %f \n", rate);
104
105    //int ex = CV_FOURCC('X', '2', '6', '4');//small size
106    int ex = CV_FOURCC('M', 'J', 'P', 'G');
107    rate = 10; //changed frame rate to 10 fps
108    int dperiod = 100; //period to wait
109    string fname=GetTimeStr()+" .avi";
110    string previousfname=fname;
111    VideoWriter *outputVideo;
112    outputVideo=new VideoWriter(wpath+fname, ex , rate , S , true);
```

```
113 Mat frame;
114 int dFlag=0;
115 int pFlag=0;
116
117 for (int i = 0;; i++) {
118     if (!cap.read(frame)) break;
119     dFlag=detectAndDisplay(frame);
120     pFlag=pir_sensor.Read();
121
122     if(pFlag){
123         putText(frame,"PIR DETECTED", Point(40,40), FONT_HERSHEY_PLAIN
124 ,2.0, CV_RGB(255, 0, 0),2.0);
125     }
126     fname=GetTimeStr()+" .avi";
127     if(previousfname!=fname){
128         previousfname=fname;
129         outputVideo->release();
130         outputVideo=new VideoWriter(wpath+fname, ex , rate, S, true);
131         if(!outputVideo->isOpened())
132             {
133                 cout << "Could not open the output video to write."<< endl;
134                 waitKey(5000);
135                 return -1;
136             }
137     else{
138         *outputVideo << frame;
139     }
140     imshow("Frame", frame);
141     if(dFlag || pFlag){
142         dperiod=CE_PERIOD_NORMAL;
143     }
144     else
145     {
146         dperiod=CE_PERIOD_FAST;
147     }
```

```

148     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
149 }
150 cap.release();
151 outputVideo->release();
152 //waitKey(5000);
153 return 0;
154 }
```

စာရင်း ၆.၁၃: Smart Surveillance Camera

ဗိုဒ္ဓိယို တွေကို သိမ်းတဲ့ အခါ X264 ကို သုံးရင် ဖိုင် အရွယ်အစား တော်တော် သေးငယ်တဲ့ ဗိုဒ္ဓိယို ဖိုင်တွေကို ရှိနိုင် ပါတယ်။ ဒါ ပေမယ့် CPU usage က တက်လာပြီး နေးလေး နောက် တွေ့နှိုင် ပါတယ်။ MJPG နဲ့ဆိုရင်တော့ ဖိုင် size ကြိုးပြီး တွက်ချက် လုပ်ဆောင်မှု ပို ပေါ်ပါး ပါတယ်။ အဲဒီ ပရိုဂရမ် မှာ USB WebCam ကို သုံးထား ပါတယ်။ အခန်းငဲ့ မှာ ဆွေးနွေးခဲ့တဲ့ GPIO class ကို လည်း ပြန်သုံးထားပါတယ်။ ပရိုဂရမ် မှာ c++11 standard လိုတဲ့ အတွက် build လုပ်တဲ့ အခါ အောက်က အတိုင်း bulid လုပ်ပြီး run နိုင် ပါတယ်။

```

$ g++ SmartCam.cpp ce_io.cpp `pkg-config --cflags --libs opencv` -std=c++11
-o SmartCam
$ gksudo ./SmartCam
```

၆.၇ Crontab ကိုသုံးခြင်း

Linux ရဲ့ cron daemon က သတ်မှတ် ပေးလိုက်တဲ့ အလုပ် တွေ ကို သတ်မှတ်ထားတဲ့ အချိန် ဖေား အလိုက် နောက်ကွယ် ကနေ လုပ်ဆောင် ပေးနိုင် ပါတယ်။ Cron က အချိန်လို့ အဓိပ္ပာယ် ရတဲ့ ကရို စာလုံး chronos ကို ဆိုလို တဲ့ utility ပရိုဂရမ် တစ်ခု ဖြစ်ပြီး လုပ်ဆောင် ချင်တဲ့ လုပ်ငန်း တွေနဲ့ အချိန် စာရင်း ကို crontab ဆိုတဲ့ ဖိုင်မှာ သတ်မှတ် နိုင် ပါတယ်။ သူကို ခိုင်းလို့ ရတဲ့ အမြန်ဆုံး ကြိမ်နှုန်းက တစ် မိနစ် တစ်ခါ ဖြစ်ပြီး အနေးဆုံး ကြိမ်နှုန်းက တစ်နှစ် တစ်ခါ ဖြစ်ပါတယ်။

Cron က အလို့ အလေ့ရာက် ပုံမှန် အချိန် ဖေားနဲ့ backup လုပ်တာ၊ maintenance လုပ်တာတွေ၊ တွေ့ခြား အချိန် ဖေားနဲ့ ပုံမှန် ထပ်ကာ တလဲလဲ လုပ်ရမယ့် လုပ်ငန်း တွေ အတွက် လွယ်ကူး၊ အဆင်ပြု၊ အသုံးဝင် ပါတယ်။ ဥပမာ BBB နဲ့ ဆက်ထား တဲ့ sensor တွေကို အချိန်မှန် ဖတ်၊ controller တွေကို

အချိန်မှန် ထိန်းကျောင်း စတာတွေ အတွက် အလွယ်တကူ အသုံးပြု နိုင် ပါတယ်။ ခုနက ဗိုဒ္ဓိလို စီမံချိန်မှု surveillance ကင်မရာ အတွက် ဆိုရင်လည်း ဗိုဒ္ဓိလို ဖိုင်တွေ များများ လာတဲ့ အခါ စက်မှာ နေရာ မလောက် တော့မယ့် ပြဿနာ ရှိလာ နိုင်ပါတယ်။ အဲဒီ အတွက် ဖိုင် အဟောင်းတွေ ကို သတ်မှတ် ထားတဲ့ ဆာဗာ မှာ အလို အလျောက် 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
```

ပထမ ဆုံး အကြိမ် ဆိုရင် သုံးမယ့် editor ကို သတ်မှတ် နိုင်ပြီး၊ သုံးနေကျ nano ကို ပုံ ၆.၆ မှာ ပြထား သလို ရွှေးလိုက် ပါမယ်။

```
debian@beaglebone:~$ crontab -e
no crontab for debian - using an empty one

/usr/bin/select-editor: 1: /usr/bin/select-editor: gettext: not found
'select-editor'.
/usr/bin/select-editor: 1: /usr/bin/select-editor: gettext: not found
  1. /bin/nano      <---- 
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny

/usr/bin/select-editor: 32: /usr/bin/select-editor: gettext: not found
1-3 [1]: 1
```

ပုံ ၆.၆: Crontab ကို ပြုပြင်ခြင်း။

၆.၇.၁ 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 လိုရေး ရင် မနက် ၆ နာရီ တစ်ခါ ညာနေ 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 (#) သင်္ကာတ 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 ၆.၇)။ အဲဒီ အခါ crontab: installing new crontab ဆိုပြီး သတ်မှတ် လိုက်တဲ့ လုပ်ငန်း အောင်အောင် မြင်မြင် တပ်ဆင် ပြီးတဲ့ အကြောင်း တွေ ရမှာ ပါ။

```
0 * * * * /home/debian/vctask.sh
```

```
GNU nano 2.7.4          File: /tmp/crontab.u6bYCU/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/debian/vctask.sh
```

The terminal window has a menu bar with options like Get Help, Write Out, Where Is, Cut Text, Justify, Cur Pos, Exit, Read File, Replace, Uncut Text, To Spell, and Go To Line.

ပုံ ၆.၇: Crontab တွင် လုပ်ငန်းများသတ်မှတ်ခြင်း။

Cron နဲ့ သတ်မှတ် ထားတဲ့ အလုပ် တွေက ဂွန်ပျူဗာ ပိတ်ထား တဲ့ အချိန်မှာ မလုပ်လိုက် ရဘူး ဆိုရင်၊ ဂွန်ပျူဗာ ပြန်ပွင့် လာတဲ့ အခါ လုပ်ဆောင် ပေးမှာ မဟုတ် ပါဘူး။ အဲဒီ လို မလုပ်လိုက်ရတဲ့ အလုပ် တွေကို စက်စ ပြန်ပွင့် လာတဲ့ အချိန်မှာ လုပ်ဆောင် ပေးစေချင် ရင်တော့ Anacron (anachronistic cron) ကို သုံးနိုင် ပါတယ်။ သူကို အောက်က command နဲ့ တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt install anacron
```

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>.
- [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/>.
- [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.
- [Ope17a] OpenCV. Cascade Classifier. 2017. url: 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.
- [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.
- [Ope17f] OpenCV. VideoWriter Class Reference. 2017. url: http://docs.opencv.org/trunk/dd/d9e/classcv_1_1VideoWriter.html.

အခန်း ၇

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 ကုဒ် 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 အခိုက ကွာခြားချက် ပါ။ Qt က LGPLv3 လိုင်စင် ကို free ပေးထားပြီး၊ ကိုယ်ပိုင် စီးပွားရေး အတွက် သုံးမယ် ဆိုရင် ကန်သတ်ချက် တရီးရှိတာကြောင့် 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 ပျော်ပါး 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 ရင့်ကျက် 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။

၇.၃ wxWidgets ကိုတပ်ဆင်ခြင်း

wxWidgets ကို BBB မှာ ထည့်ဖို့အတွက် အောက်က စာရင်း ၇.၁ အတိုင်း terminal မှာ command တွေရှိကြပြီး တပ်ဆင်ထည့်သွင်းနိုင် ပါတယ် [wxW14]။

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

စာရင်း ၇.၁: Linux ဘွင် wxWidgets ကို တပ်ဆင်ခြင်း။

နမူနာ အနေနဲ့ ရိုးရှင်း တဲ့ wxsimple.cpp ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၇.၂ 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 ၇.၂: wxsimple.cpp

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

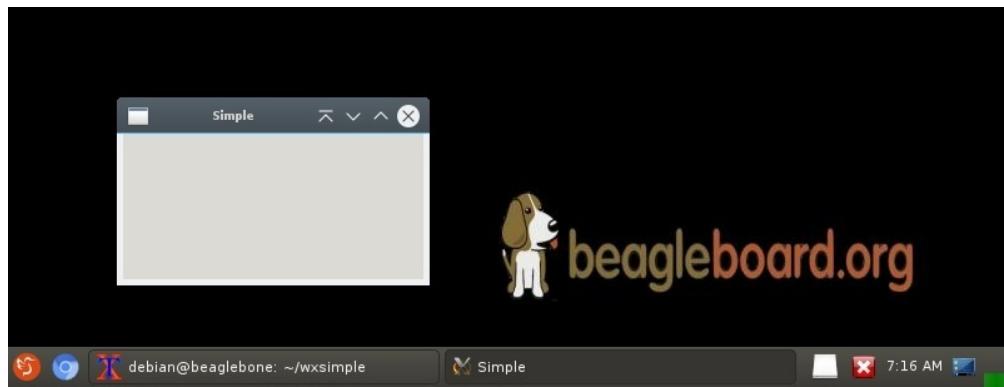
```
$ g++ wxsimple.cpp `wx-config --cxxflags --libs` -o wxsimple
```

GUI application ဖြစ်တဲ့ အတွက် tightvncserver ကို သုံးပြီး run ဖို့လို ပါတယ်။ သူမှာ ပါတဲ့ Terminal က အဆင် မဖြေ ရင် xterm ကို အောက်ပါ အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
$ sudo apt-get install xterm
```

ပြီးရင် xterm မှာ ခုနက ရလာတဲ့ binary ကို run လိုက်တဲ့ အခါ ပုံ ၇.၁ အတိုင်း တွေ့နိုင် ပါတယ်။

```
$ ./wxsimple
```



ပုံ ၇.၁: wxWidgets အတွက် wxsimple.cpp နှမူနာ။

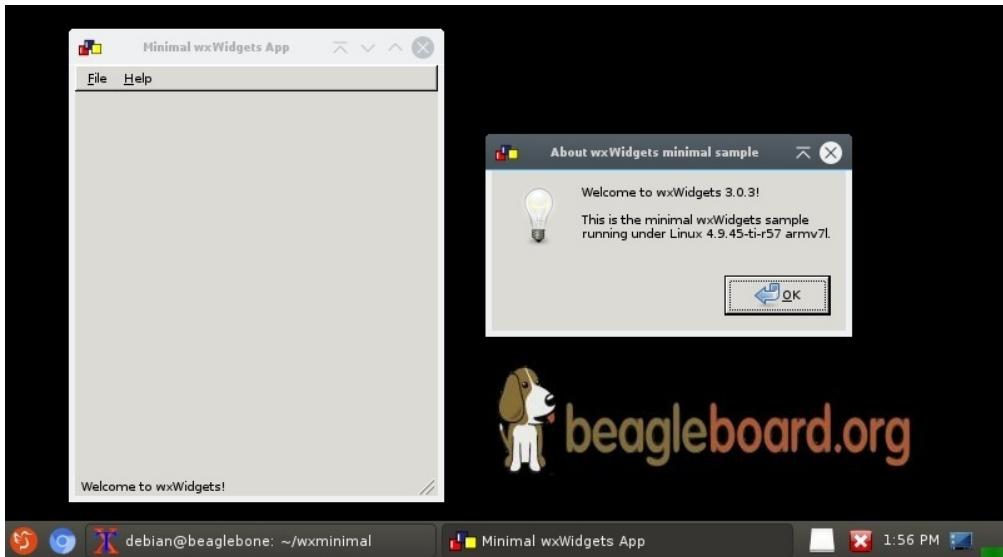
၇.၂ wxWidgets ကို source မှ build လုပ်ခြင်း

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

```
$ sudo apt-get install build-essential
$ sudo apt-get install libgtk-3-dev
$ wget https://github.com/wxWidgets/wxWidgets/releases/download/v3.0.3/
      wxWidgets-3.0.3.tar.bz2
$ tar xjvf wxWidgets-3.0.3.tar.bz2
$ cd wxWidgets-3.0.3
$ 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 wxsimple.cpp (Appendix B စာရင်း ၂.၁) ဆိုတဲ့ နှမူနာကို အောက်က command တွေနဲ့ run ကြည့် လိုက်တဲ့ အခါ ပုံ ၇.၂ အတိုင်း တွေ့ရ ပါတယ်။

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



ံ ၇.၂: wxWidgets ဖွင့် minimal နမူနာကို run ခြင်း။

၇.၃ OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း

OpenCV နဲ့ wxWidgets တဲ့ သုံးတဲ့ အကြောင်း ဆွဲနေးပါမယ်။ Linux နဲ့ terminal ပေါ်မှာ command ရှိက်ထည့် ပြီး build လုပ်တာ လွယ်ကူ ရုံးရှင်း ပါတယ်။ နမူနာ အနေနဲ့ `wxcvssimple.cpp` ဆိုတဲ့ ရုံးရှင်း တဲ့ ပရိုဂရမ် လေး တစ်ခု ရေးကြည့် ပါမယ်။ သူကို စာရင်း ၂၃ မှာ ဖော်ပြထား ပါတယ်။

```
1 //File: wxcusimple.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
43 {
44 public:
45     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 ၇.၃: wxcvssimple.cpp

ပရိုဂရမ် အစမှာ Application ရဲ့ OnInit() ဆိုတဲ့ method ထဲမှာ

```
wxInitAllImageHandlers();
```

ဆိုတာကို ထည့်ပါမယ်။ အဲဒီနောက် MyFrame ဆိုတဲ့ wxFrame ရဲ့ derived class ထဲမှာ ပုံရှိပ် 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), wxDefaultPosition, wxSize(256, 256));

```

Image တွေ အတွက် OpenCV မှာသုံးတဲ့ Mat နဲ့ wxWidgets ရဲ့ wxImage ကို အပြန် အလှန် ပြောင်း ပေးတဲ့ mat_from_wx | wx_from_mat အစ ရှိတဲ့ ဖန်ရှင် တွေကို စာရင်း ၇.၄ 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,CV_GRAY2RGB);}
13     else if (img.channels() == 4) { cvtColor(img, im2, CV_BGRA2RGB);}
14     else {cvtColor(img,im2,CV_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,CV_RGB2BGR);
26     return im2;
27 }
```

စာရင်း ၇.၄: util.h

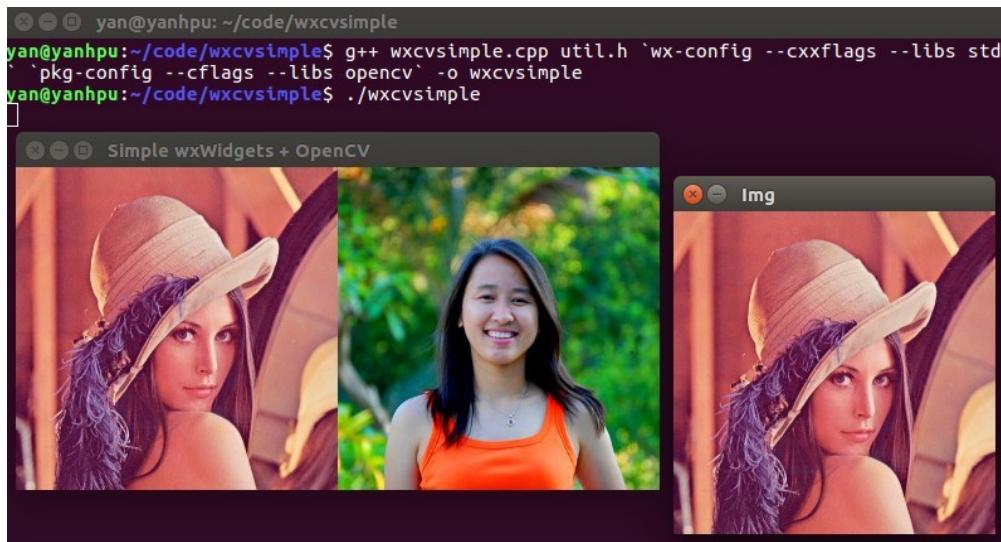
ဆက်ပြီး MyFrame ရဲ့ constructor ထဲမှာ သိရှိ ရဲ့ ပုံကို OpenCV သုံးပြီး imread နဲ့ ဖတ်လိုက်ပါတယ်။ ဖတ်လို ရတဲ့ Mat အမျိုး အစား ပုံကို wx_from_mat ဖန်ရှင် ကို သုံးပြီး wxImage အမျိုး အစား ပြောင်းလိုက် ပါတယ်။ ပြီးတော့ wxStaticBitmap ရဲ့ SetBitmap method သုံးပြီး Frame ပေါ်မှာ ဖော်ပြထိကို ပါတယ်။

နောက်တစ်ခါ wxImage အမျိုး အစား variable တစ်ခု ကြော်ပြီး LoadFile method နဲ့ လီနာရဲ့ ပုံကို wxWidget သုံးပြီး ဖတ်လိုက် ပါတယ်။ ရလာတဲ့ wxImage အမျိုး အစား ပုံကို mat_from_wx ဆိုတဲ့ ဖန်ရှင် သုံးပြီး OpenCV အတွက် Mat အမျိုး အစား ပြောင်းလိုက် ပါတယ်။ ရလာတဲ့ ပုံရှင်ပို့ OpenCV ရဲ့ imshow နဲ့ ဖော်ပြု လိုက် ပါတယ်။

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

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

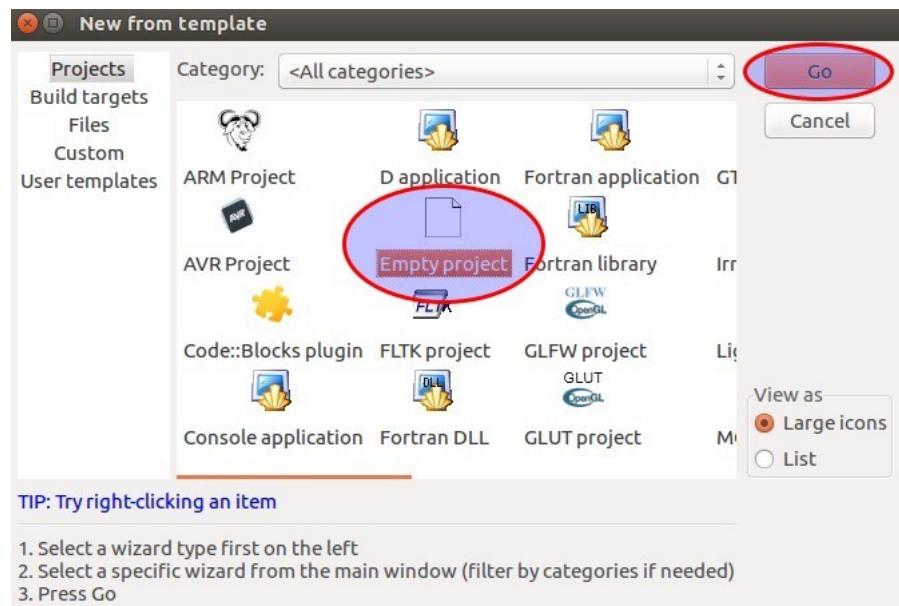
Terminal မှာ ./wxcvsimple ကို ရိုက်ပြီး ပရီကရမဲ ကို run လိုက်တဲ့ အခါ ရလာတဲ့ ရလဒ် ကို ပုံ ၇.၃ မှာ တွေ့နိုင် ပါတယ်။



ပုံ ၇.၃: wxWidgets နှင့် OpenCV ကို တွဲစပ် အသုံးပြု ထားသည့် ရိုးရှင်းသော နမူနာ။

၇.၃.၁ Code::Blocks

Application တစ်ခု ကို BBB ပေါ်မှာ တိုက်ရိုက် develop လုပ်ရ တာ ထက် စာရင် အပိုင်း ၂.၁၁ မှာ ပြောခဲ့သလို desktop ကွန်ပျိုးတာ ပေါ်မှာ ပရီကရမဲ ကို အရင် ရေးပြီး မှ BBB ပေါ် ပြောင်းရ တာ ပိုပြီး လွယ်ကူ အဆင်ပြေ ပါတယ်။ Code::Blocks မှာ wxWidgets ကို OpenCV နဲ့ တွဲသုံးဖို့အတွက် wxWidgets project အသစ် တစ်ခု ကို ပုံ ၇.၄ မှာ ဖော်ပြု ထားတဲ့ အတိုင်း empty project တစ်ခု ဖန်တီးပြီး wxcv.cpp ဆိုတဲ့ project နာမည် ပေးလိုက် ပါမယ်။ ပြီးတဲ့ အခါ GNU GCC Compiler ကို ရွေး ပါမယ်။



ံ ၇.၄: Code::Blocks တွင် Empty project စာစ်ခု ဖန်တီးခြင်း။

File Menu → New → Empty File ကို နိပ်ပြီး စာရင်း ၇.၅ 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();
16     Mat img=imread("grove.jpg",CV_LOAD_IMAGE_COLOR);

```

၁၇

၇.၃. OPENCV နှင့် WXWIDGETS ကိုထွေသုံးခြင်း

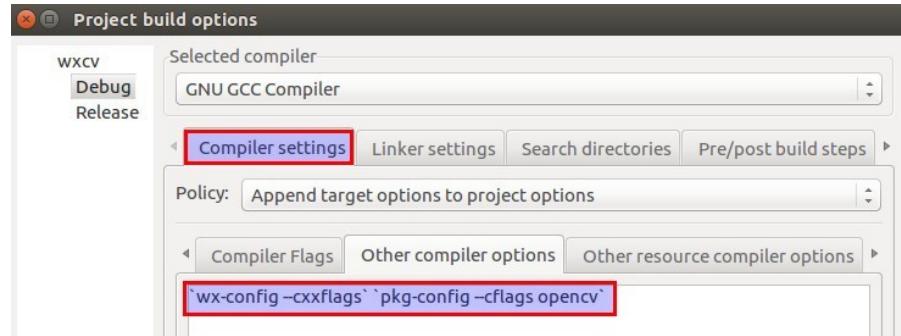
```
17 namedWindow("Img", CV_WINDOW_AUTOSIZE);
18 imshow("Img", img);
19 }
20
21 class MyApp : public wxApp
22 {
23     public:
24         virtual bool OnInit();
25 };
26
27 IMPLEMENT_APP(MyApp)
28 bool MyApp::OnInit()
29 {
30     Simple *simple = new Simple(wxT("Simple"));
31     simple->Show(true);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

စာရင်း ၇.၅: wxcv.cpp

ပြီးရင် Project Menu → Build Options... ကို နှိပ်ပြီး Compiler settings tab → Other compiler options မှာ ပဲ ဂျော် အတိုင်း

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

କ୍ରି �Debug ଅଟୁଗନ୍ତରେବା Release ଅଟୁଗନ୍ତପି ସର୍ବମୁହଁ ଫ୍ରିଦ ପିତାଯି॥

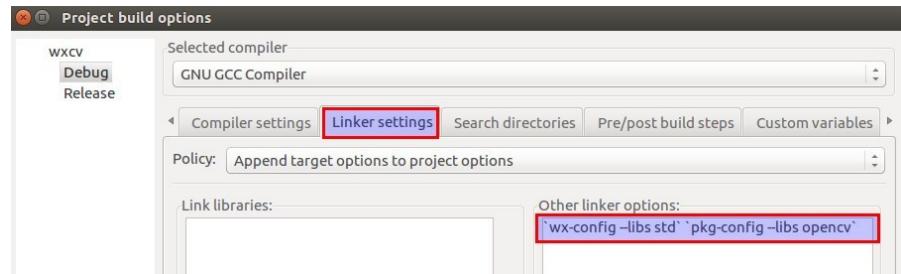


ပုံ ၇.၅: Compiler options များသတ်မှတ်ခြင်း။

နောက်တစ်ခါ Linker settings tab → Other linker options မှာ လည်း ပုံ ၇.၆ အတိုင်း

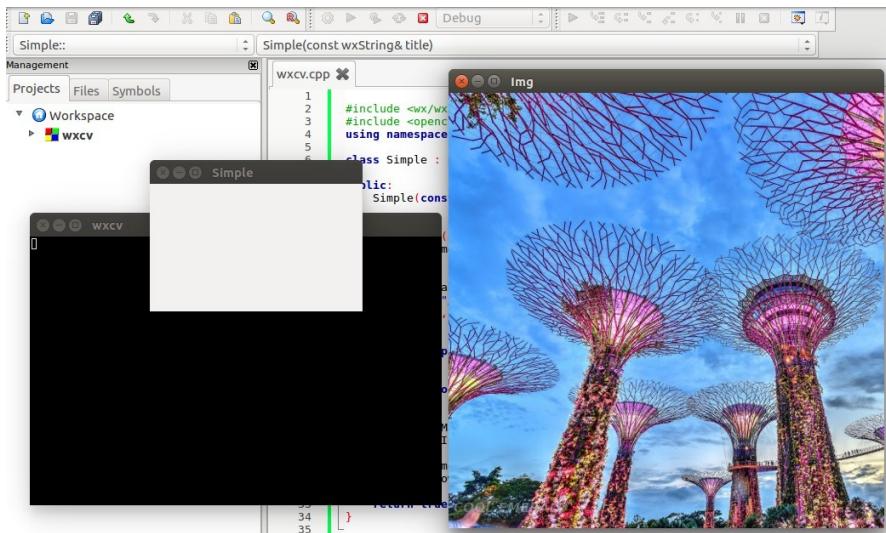
```
'wx-config --libs std` `pkg-config --libs opencv'
```

ကို ထပ် သတ်မှတ်ပါမယ်။



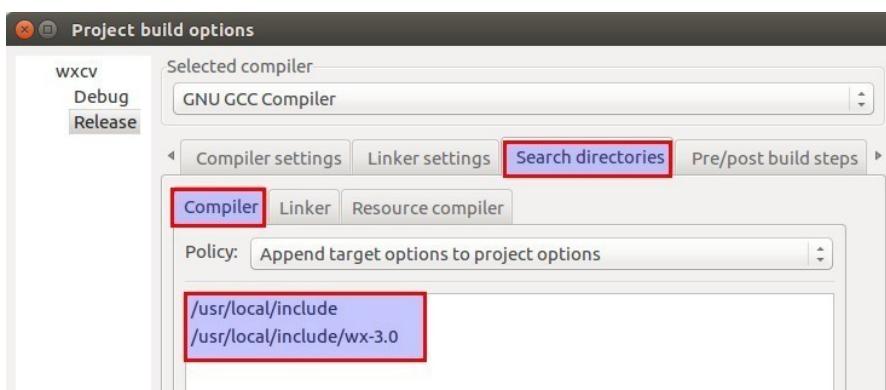
ပုံ ၇.၆: Linker options များသတ်မှတ်ခြင်း။

နောက်ဆုံးမှာ F9 ခလုပ်ကို နိုပ်ပြီး Build and run လုပ်လိုက် တဲ့အခါ ပုံ ၇.၇ မှာ ပြထား သလို ပရိုဂရမ် ရဲ့ output ကို တွေ့နိုင် ပါတယ်။



ပုံ ၇.၇: OpenCV နမူနာ ပရိုဂရမ်ကို Code::Blocks ဘွင် run ခြင်း။

ပရောဂျက် နဲ့ ပတ်သက် တဲ့ အချက်အလက် တွေကို IDE ကို ပိုပြီး သိစေချင်ရင် optional အဆင့်တွေ အနေဖြင့် Project Menu → Build Options... ကို နှိပ်ပြီး Search directories tab → Compiler tab မှာ ပုံ ၇.၈ အတိုင်း /usr/local/include နဲ့ /usr/local/include/wx-3.0 ကို Debug အတွက်ရော၊ Release အတွက်ပါ သတ်မှတ် နိုင် ပါတယ်။

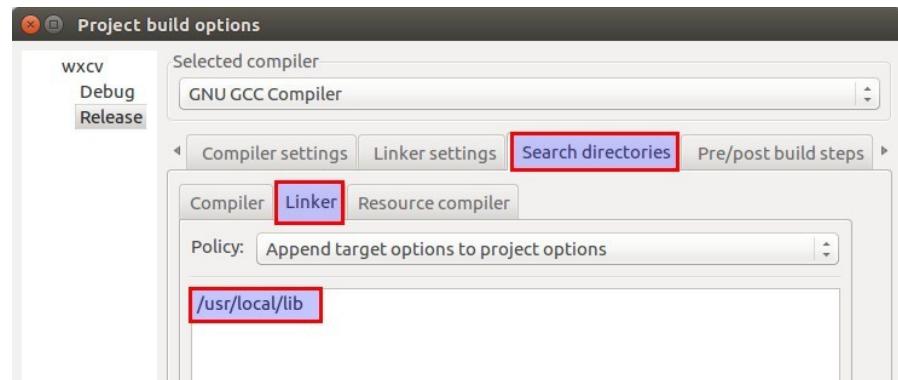


ပုံ ၇.၈: Compiler Search directories များသတ်မှတ်ခြင်း။

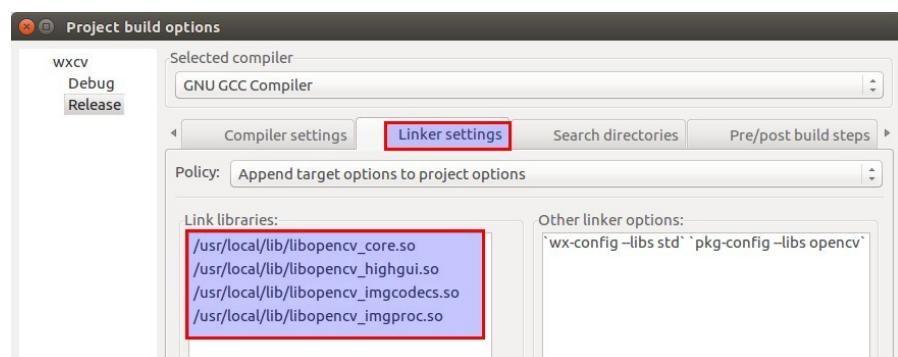
နောက်တစ်ခါ ဘေးဘက် က Linker tab မှာ ပုံ ၇.၉ အတိုင်း lib တွေရဲ့လမ်းကြောင်း /usr/local/lib ကို သတ်မှတ် ပါမယ်။ Linker settings tab မှာလည်း ပုံ ၇.၁၀ အတိုင်း Link libraries တွေ ဖြစ်တဲ့

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

အစ ရှိတဲ့ library တွေကို လိုသလို သတ်မှတ် နိုင် ပါတယ်။



ပုံ ၇.၉: Library ဖိုင်များအတွက် Search directories များသတ်မှတ်ခြင်း။



ပုံ ၇.၁၀: Link libraries များသတ်မှတ်ခြင်း။

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

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

```
$ pkg-config --libs opencv
```

Transparency ပါတဲ့ channel 4 ခု ရှိတဲ့ ပုဂ္ဂိုလ် တွေကို imread နဲ့ ဖတ်တဲ့ အခါ IMREAD_UNCHANGED ကို သုံးနိုင် ပြီး၊ အဲဒီ အတွက် နမူနာ ကို Appendix B စာရင်း J.J မှာ ပြထားတဲ့ [minimal.cpp](#) ဆိုတဲ့ ပရိုဂရမ် မှာ တွေ့နိုင် ပါတယ်။

အကိုးအကားများ

- [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.
- [wxW14] wxWiki. Compiling and getting started. 2014. url: 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 ၈

Serial Port ကိုသုံးခြင်း

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 ပါရှိ ပါတယ်။

၈.၁ 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။

၈.၁.၁ 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။

၈.၁.၂ Baud Rate

Data bit တစ်ခု နဲ့ တစ်ခု ကြားက အချိန် အကွာ အဆေး 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။

၈.၁.၃ Data Bits

Data ပို့တဲ့ အခါ ပို့တဲ့ UART နဲ့ လက်ခံ တဲ့ UART တို့ရဲ့ data bit အရေး အတွက် ခြင်း တူဖို့ လို ပါတယ်။ ပို့တဲ့ အခါ 5 bit data ကနေ 8 bit data ထိ ပို့တတ် ပါတယ်။ ဥပမာ ASCII data ဆိုရင် ပို့တာ 7 bit ပဲ ရှိတဲ့ အတွက် လက်ခံ ရင်လဲ 7 bit လက်ခံ ဖို့ လို ပါတယ်။ Data bits တွေကို ပို့တဲ့ အခါ LSB ကနေ အရင် စပို့ ပါတယ်။ အသုံး များတာ ကတော့ 8 bit ပါ။

၈.၁.၄ 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 အရေး အတွက် စုံ ကဏ္ဍား ဖြစ်အောင် ထည့်ပေး ရင် Even parity ၊ မ ကဏ္ဍား ဖြစ်အောင် ပေါင်းထည့် ပေးရင် Odd parity ပါ။ ပို့တဲ့ UART နဲ့ လက်ခံ တဲ့ UART parity ခြင်းလည်း ကိုက်ညီ ဖို့ လို ပါတယ်။ များသော အားဖြင့် Parity ထည့် သုံးလေ့ မရှိ ပါဘူး။

၈.၁.၅ Stop Bit

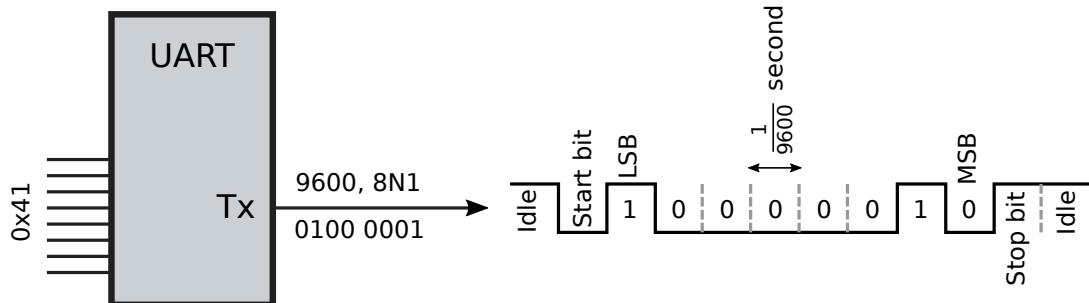
8 bits ရှိတဲ့ data byte တစ်ခု နဲ့ တစ်ခု ကြားမှာ ခြား ပေးဖို့ အတွက် stop bits တွေကို သုံး ပါတယ်။ 1 ကို stop bit အနေ နဲ့ သုံး ပါတယ်။ Stop bit ကို တစ်ခု၊ သို့မဟုတ် နှစ်ခု သုံး လို ရ ပါတယ်။ data တွေ၊ parity bit တွေ ပို့ပြီး တဲ့ အခါ နောက်မှာ 1 တစ်ခု ပေါင်း ပေး၊ ဒါမှ မဟုတ် နှစ်ခု ပေါင်းထည့် ပေး တာပါ။ အသုံး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

အောက်က ပုံ ၈.၁ မှာ နမူနာ အနေ နဲ့ data byte 0x41 ကို UART တစ်ခု ကနေ 9600, 8N1 နဲ့ ပို့တာ 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။

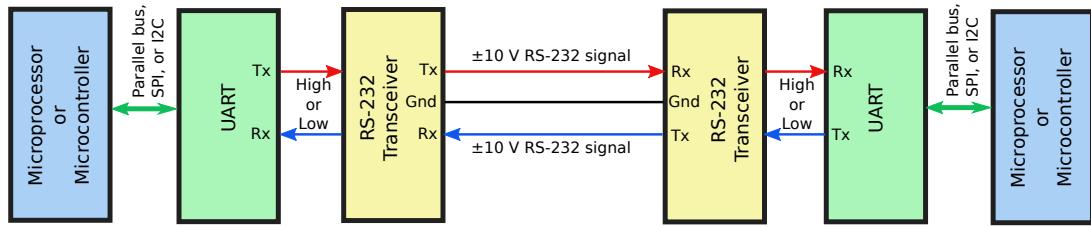


ပုံ ၈.၁: UART တစ်ခု မှ data byte (0x41) ကို serial bit stream (9600, 8N1) အဖြစ် ပြောင်းပေးခြင်း။

၈.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 တစ်ခုခု ခံပြီး 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 ± 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။



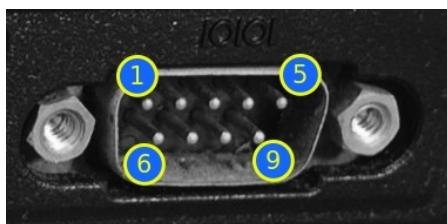
ပုံ ၈.၂: RS-232 ဆက်သွယ် အသုံးပြုခြင်း။

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 ဖို့ မရှိ ပါဘူး။ RS232 ကို unbalanced lines လို ဆို ပါတယ်။ Tx ပါယာနဲ Ground ဝါယာ ကြားက ဖို့က transmit voltage ဖြစ်ပြီး Rx ပါယာ နဲ Ground ဝါယာ ကြားက ဖို့က receive voltage ဖြစ်ပါတယ်။

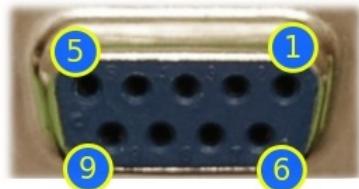
၈.၂၁ Handshaking

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 ၈.၃ 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 ၈.၃: RS-232 အတွက် DB9 connector များ။

DTE (ဂုန်ပျူတာ) 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 တွေ့နိုင် ပါတယ်။

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 ဖို့ မှာပဲ 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 ဖို့ မှာပဲ 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 ဖို့ မှာပဲ 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။

၈.၃ 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 ၈.၄ မှာ ပြထားတဲ့ USB to RS232 converter လေး တွေပါ။

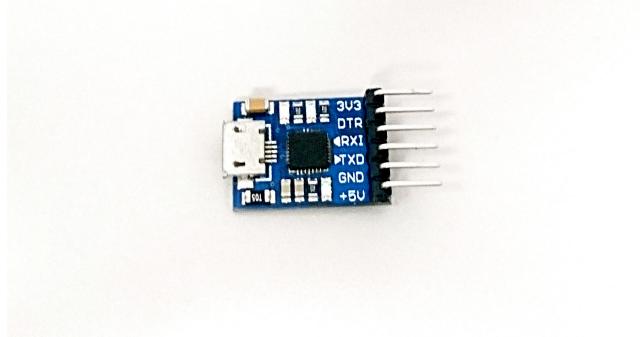


ပဲ ၈.၄: USB to RS232 converter တစ်ခု။

အဲဒီ USB to RS232 converter တွေက RS232 သုံးထား တဲ့ ပစ္စည်း၊ ကိရိယာ တွေနဲ့ ဆက်သွယ် အသုံးပြု ဖို့ အတွက် အဆင်ပြု ပေမယ့် BBB တို့လို့ UART ပဲ ပါပြီး 3.3V ပဲသုံးတဲ့ ကိရိယာ တွေ အတွက်တော့ ± 12 V ထွက်တဲ့ RS232 pin တွေနဲ့ တိုက်ရှိက် ဆက်သုံးလို့ မရ ပါဘူး။ ဒါကြောင့် BBB ကို RS232 ပစ္စည်း တွေနဲ့ ဆက်သုံး ချင်ရင် MAX202 တို့လို့ RS232 transceiver chip တစ်ခု ကြားမှာ ခံဖို့ လိုပါတယ်။ Host computer နဲ့ BBB ကို တိုက်ရှိက် ဆက်သွယ် ချင်ရင်တော့ USB to UART converter တွေဖြစ်တဲ့ FTDI cable (ပဲ ၈.၅) တို့၊ CP2102 USB to UART bridge (ပဲ ၈.၆) တိုကို သုံးနိုင် ပါတယ်။



ပဲ ၈.၅: FTDI cable တစ်ခု။



ပုံ ၈.၆: CP2102 USB to UART bridge တစ်ခု။

FTDI cable အမျိုးမျိုး ရှိပြီး ပုံ ၈.၅ မှာ ပြထားတဲ့ ကြိုးက 3.3V TTL voltage level နဲ့ AliExpress က ဝယ်ထားတဲ့ ဈေးပေါ်တဲ့ အမျိုးအစား ပါ။ သူ အတွက် pin connection တွေကို တော့ အောက်က စာရင်းမှာ တွေ့နှိုင် ပါတယ်။

၁။ အနုက်ရောင်ကြိုး: GND - System Ground

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

၃။ အနီရောင်ကြိုး: VCC - 5 V

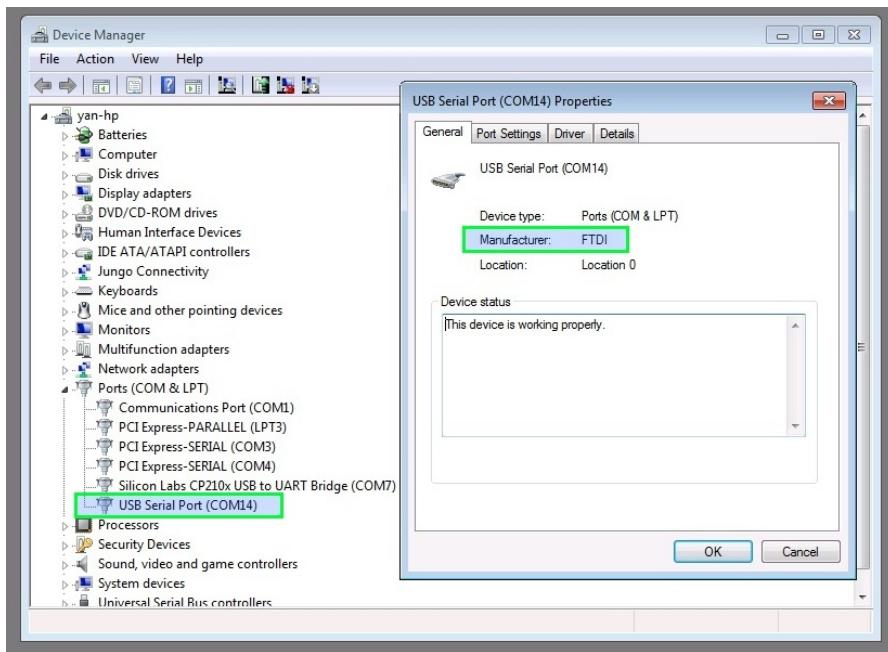
၄။ အစိမ်းရောင်ကြိုး: Tx- Transmit Data (output)

၅။ အဖြူရောင်ကြိုး: Rx- Receive Data (input)

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

၈.၃.၁ 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 တွေ့နှိုင် ပါတယ် (ပုံ ၈.၇)။

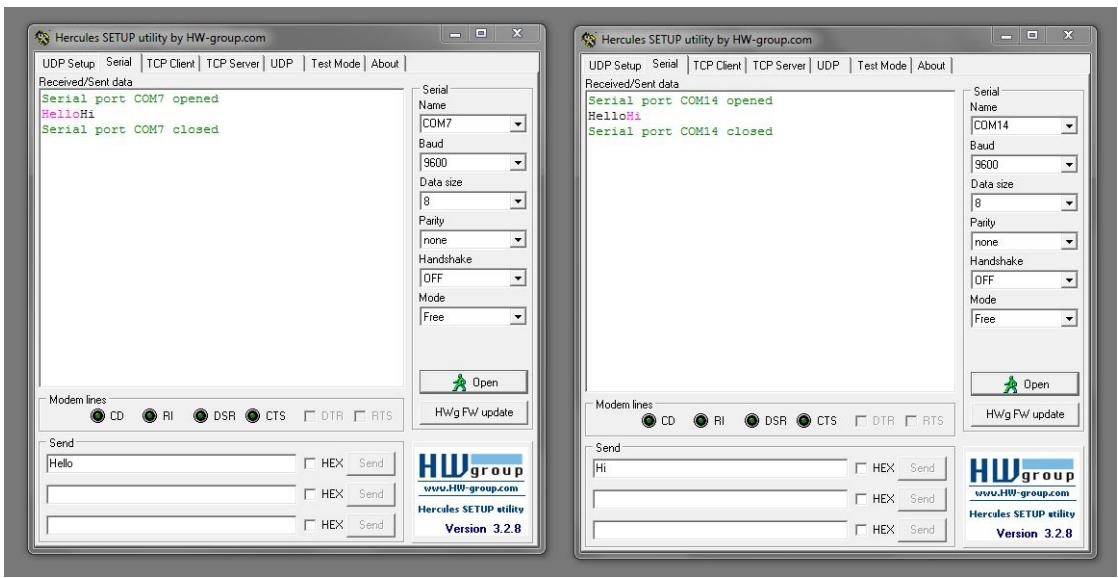


ပုံ ၈.၃: Device manager တွင် တွေ့နိုင်သော COM port နာမည်။

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။

သူတိုကို သုံးပြီး ဒေတာ တွေ စမ်းပို့ ကြည့်ဖို့ အတွက် စက်နှစ်လုံး ပဲဖြစ်ဖြစ်၊ စက်တစ်လုံး ထဲမှာ device နှစ်ခု တပ်ပြီး ပဲ ဖြစ်ဖြစ်၊ အချင်းချင်း ဆက်ပြီး ပို့ကြည့် နိုင် ပါတယ်။ အနည်းဆုံး ပါယာ သုံးခု ဆက်သွယ်ဖို့ လိုပါတယ်။ Ground အချင်းချင်း ဆက်၊ နောက် တစ်ခု ရဲ့ TX ကို တွေား တစ်ခု ရဲ့ RX နဲ့ အပြန်အလုန် ဆက်သွယ်ဖို့ လိုပါတယ်။ အဲလို့ မှ 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 အဆင်ပြေ တာကို တွေ့ရ ပါတယ်။ Hercules ကို Windows ကွန်ပျိုးတာ မှာ တပ်ဆင် လိုက်ပြီး ပရိုဂရမ် ကို ဖွင့်ပြီး တဲ့ အခါ serial tab မှာ port number, baud rate စတာတွေ သတ်မှတ် ပြီး စမ်းသပ် ပေးပို့ ထားတာ ကို ပုံ ၈.၄ မှာ နမူနာ အနေနဲ့ တွေ့နိုင် ပါတယ်။



ပုံ ၈.၈: Hercules ဖြင့် serial port များကို စမ်းသပ်ခြင်း။

၈.၂.၂ Linux တွင်အသုံးပြုခြင်း

Linux ပေါ်မှာ တော့ FTDI cable ရော့၊ CP2012 ရော့ ဘာမှ ထပ်လုပ် စရာ မလိုပဲ တန်း သုံးလို့ ရတာ ကို တွေ့ရ ပါတယ်။ Serial port ရဲ့ နာမည် ကို dmesg သုံးပြီး အောက်က စာရင်း နဲ့ ပုံ ၈.၉ 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:~$
```

ပုံ ၈.၉: FTDI cable ၏ serial port ကို စစ်ခြင်း။

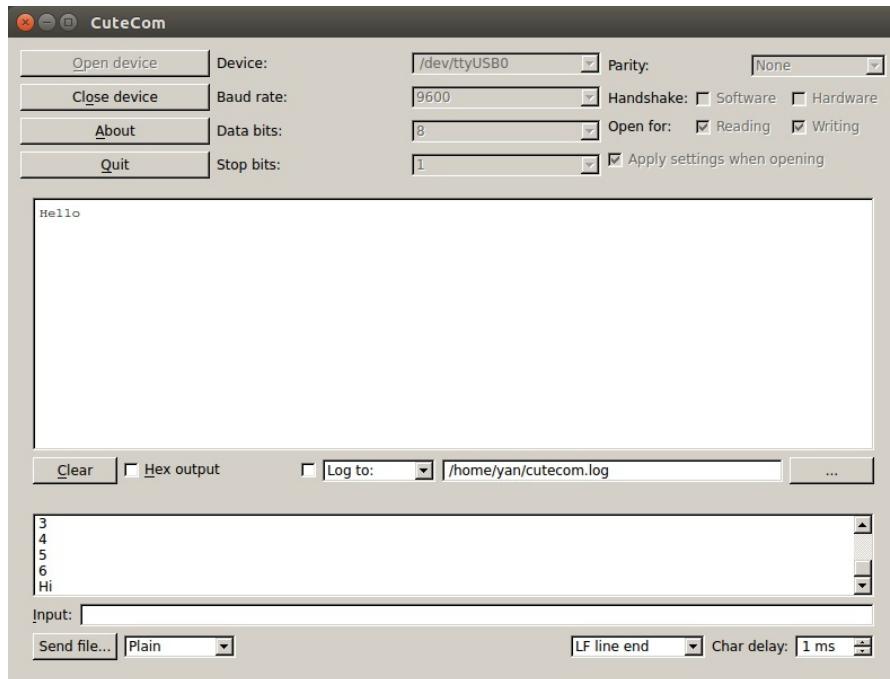
CP2102 အတွက် လည်း အောက်က command နဲ့ စစ် ကြည့်နိုင် ပါတယ်။

```
$ 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 ၈.၁ မှာ ပြထား သလို terminal မှာ superuser အနေနဲ့ တပ်ဆင် အသုံးပြု လိုလည်း ရ ပါတယ်။ CuteCom ကို တပ်ဆင် တဲ့ run တဲ့ အခါ မှာလည်း superuser အနေနဲ့ အောက်က စာရင်း ၈.၂ နဲ့ ပုံ ၈.၁၀ မှာ ပြထား သလို တပ်ဆင် အသုံးပြု လိုရ ပါတယ်။

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

စာရင်း ၈.၁: GtkTerm ကိုတပ်ဆင်အသုံးပြုခြင်း။



ပုံ ၈.၁၀: CuteCom ကိုအသုံးပြုခြင်း။

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

စာရင်း ၈.၂: 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 ကိုအောက်ပါ 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 လိုဂျပြီး 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
```

၈.၄ BBB ၏ serial port ကိုသုံးခြင်း

BeagleBone မှာ onboard serial port ဒဲ ခု ပါ ရှိပါတယ်။ အဲဒီ ထဲက /dev/tty00 က အစ ကတည်းက enabled ဖြစ်နေပြီး၊ serial console header, J1 နဲ့ ဆက်ထား ပါတယ်။ တခြား serial port တွေက disabled ဖြစ်နေပြီး သုံးဖို့ အတွက် enable လုပ်ဖို့ လို ပါတယ်။ ရွှေမှာ ဖော်ပြ ခဲ့ သလို FTDI cable တို့ CP2102 တို့ကို သူရဲ့ USB မှာ တပ်ပြီး သုံး လိုလည်း ရပါတယ်။ BeagleBone Black ရဲ့ serial port တွေကို ပေါ်မှု မှာ တွေ့နိုင် ပါတယ် [Cam13]။

ပေါ်ပေါ်မှုံး။ ရ.၁: BeagleBone Black ၏ serial port များ။

| Port | Device | RX | TX | CTS | RTS |
|-------|------------|-------|-------|-------|-------|
| UART0 | /dev/ttyO0 | J1_4 | J1_5 | NA | NA |
| UART1 | /dev/ttyO1 | P9_26 | P9_24 | P9_20 | P9_19 |
| UART2 | /dev/ttyO2 | P9_22 | P9_21 | P8_37 | P8_38 |
| UART3 | /dev/ttyO3 | NA | P9_42 | P8_36 | P8_34 |
| UART4 | /dev/ttyO4 | P9_11 | P9_13 | P8_35 | P8_33 |
| UART5 | /dev/ttyO5 | P8_38 | P8_37 | P8_31 | P8_32 |

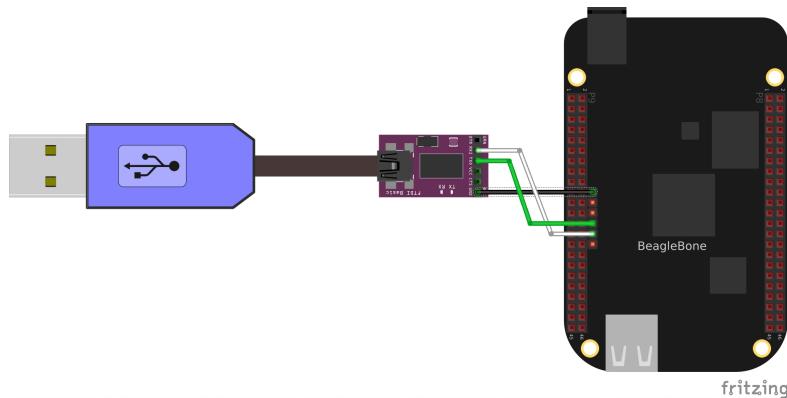
BBB ရဲ့ serial port တွေကို အသုံးပြု နိုင်တဲ့ ပုံစံ နှစ်မျိုး ရှိပါတယ် [Eli17b]။

Linux Console: PC ကို serial console header, J1 နဲ့ ဆက်ပြီး BBB 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: BBB အတွက် serial interface အနေနဲ့ တစ်ခြား device တွေနဲ့ ဆက်သွယ် အသုံးပြု နိုင် ပါတယ်။

၈.၄.၁ Linux Console

BBB မှာ J1 header က UART0 ရဲ့ serial debug connector ပါ။ အဖြူရောင် အစက်လေး ပါတဲ့ အစွမ်း ဘက်က pin က pin 1 ဖြစ်ပြီး GND ပါ။ Pin 4 နဲ့ 5 က RX နဲ့ TX ဖြစ်ပြီး ကျန်တဲ့ pin တွေက ဘာမှ ဆက်မထားတဲ့ (Not Connected) pin တွေ ဖြစ်ပါတယ်။ အဲဒါ မှာ baud rate 115200, 8N1 သုံးပြီး ဆက်လိုက်ရင် BBB Boot လုပ်တဲ့ အချိန် အချက် အလက် တွေကို ကြည့် debug လုပ်နိုင် ပြီး terminal အနေနဲ့ သုံးနိုင် ပါတယ်။ သူကို PC နဲ့ ဆက်သွယ်ဖို့ အတွက် FTDI cable ဖြစ်ဖြစ် သုံးပြီး ပုံ ၈.၁၁ နဲ့ ပုံ ၈.၁၂ မှာ ပြထား သလို ဆက်သွယ် နိုင်ပါတယ်။



ပုံ ၈.၁၁: PC နှင့် BBB ၏ serial console header ကို ဆက်သွယ်ခြင်း။



ပုံ ၈.၁၂: Serial console ကို FTDI cable ဖြင့်ဆက်ခြင်း။

အဲဒီနောက် PC မှာ ရေးမှာဖော်ပြုခဲ့တဲ့ Hercules, CuteCom, GtkTerm စတဲ့ နှစ်သာက် အဆင်ပြုရာ တစ်ခု ခုသုံးပြီး BBB အတွက် console အနေနဲ့ သုံးလို့ ရပါတယ်။ ဒီ နေရာ မှာတော့ နမူနာ အနေနဲ့ Windows ပေါ်မှာ PuTTY သုံးပြီး ဆက်သွယ် ပြထားပါတယ်။ ဆက်သွယ်မယ့် serial interface အတွက် settings တွေကို အောက်က စာရင်း မှာ ပြထားပါတယ်။

Baud rate: 115200

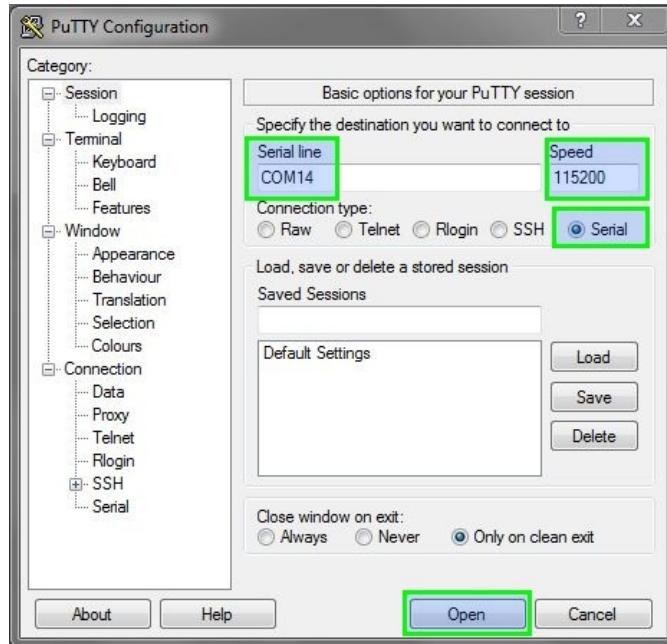
Data bits: 8

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 ထည့်ပြီး Open ကို နိုင်နိုင် ပါတယ် (ပုံ ၈.၃၃)။ ပြီးတဲ့ အခါ BBB ကို ပါဝါ ဖွင့်လိုက် ရင် ပုံ ၈.၃၄ မှာ ပြထား သလို console ကို အသုံးပြုနိုင် တာကို တွေ့ရ မှာပါ။



ပုံ ၈.၃၃: PuTTY သုံး၍ serial port ကို ဆက်သွယ်ခြင်း။

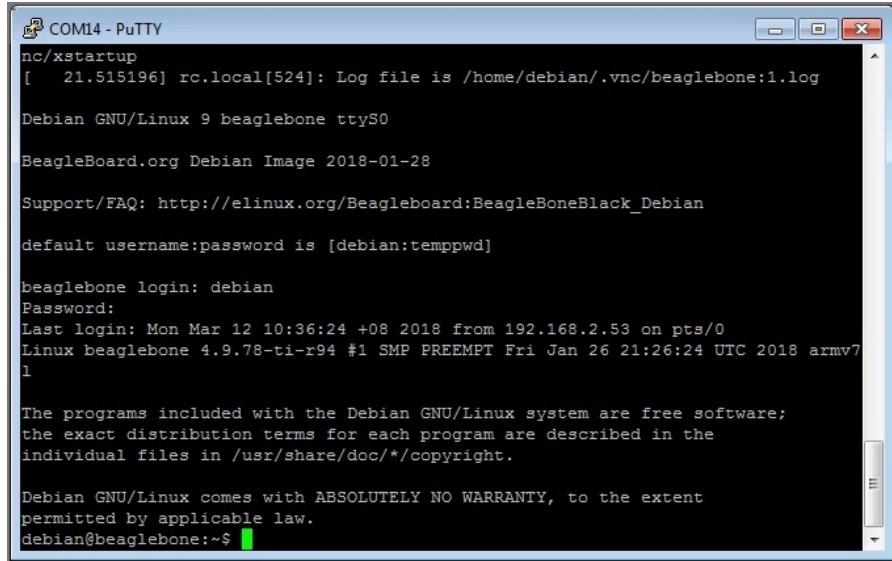
အကယ်၍ Linux PC အတွက် ဆိုရင်တော့ CuteCom , GtkTerm တို့အပြင် screen ဆိုတဲ့ application ကိုလည်း အောက် က အတိုင်း ရိုက်ပြီး သုံး လို့ရ ပါတယ်။

```
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 နိုင် ပါတယ်။



ပုံ ၈.၄: PuTTY ရှိ console ။

၈.၄.၂ Serial Interface

BBB ရဲ့ UART တွေကို P8 နဲ့ P9 header တွေမှာ ဆက်သွယ် အသုံးပြုလို ရပြီး၊ သူတို့ကို /boot/uEnv.txt မှာ U-Boot Overlays သုံးပြီး enable လုပ်နိုင် ပါတယ်။ အဲဒီ အတွက် အောက်ပါ အတိုင်း edit လုပ်နိုင် ပါတယ် [Eli18]။

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

ဥပမာ UART4 ကို enable လုပ်မယ် ဆိုရင် အဲဒီ ဖိုင် ထဲမှာ

```
uboot_overlay_addr0=/lib/firmware/BB-UART4-00A0.dtbo
```

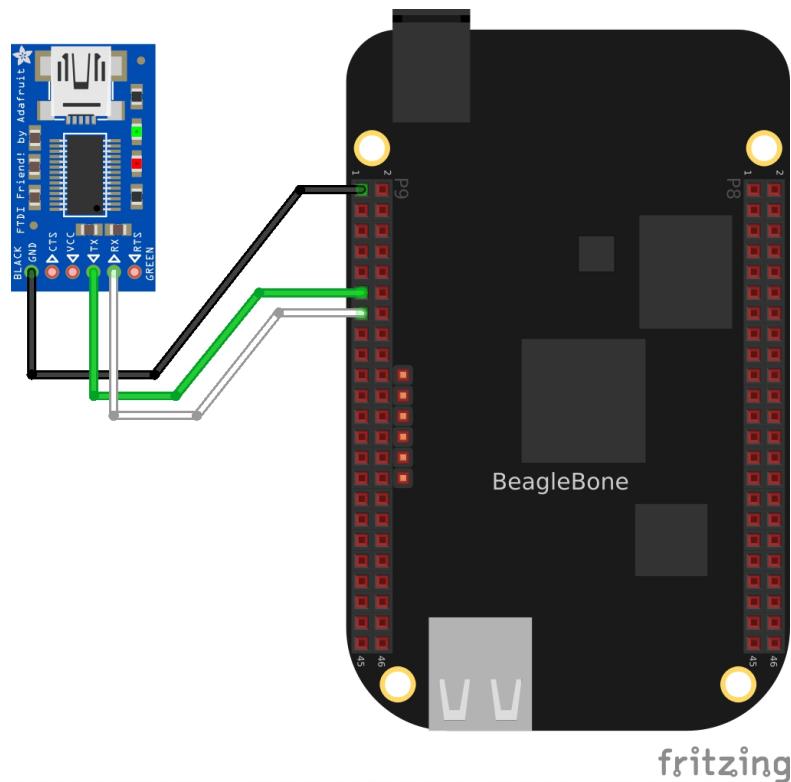
ဆိုတဲ့ စာကြောင်းကို ဖြည့်ပြီး တဲ့ အခါ Ctrl+O နဲ့ သိမ်း၊ Ctrl+X နဲ့ ထွက်ပြီး၊ BBB ကို reboot လုပ်လိုက် ပါမယ်။ Firmware ဗားရှင်း အဟောင်းတွေ အတွက် ဆိုရင်တော့ bone_capemgr ကို သုံးတာ ဖြစ်ပြီး အသေးစိတ် ကို နောက်ဆက်တဲ့ A အပိုင်း C.၁ မှာ တွေ့နိုင် ပါတယ်။

UART4 ကို enable လုပ်ပြီးတဲ့ အခါ မှာတော့ tty04 ကို GtkTerm, CuteCom စတဲ့ အဆင်ပြေ ရာ application တစ်ခု နဲ့ ဖွင့်ပြီး သုံးလို ရတာ ကို တွေ့ရ ပါလိမ့်မယ်။ SSH တို့လို terminal ပေါ်မှာ သုံးပြီး ပို့ချင်ရင် လည်း minicom တို့လို application ကို BBB ပေါ်မှာ အောက်က အတိုင်း တပ်ဆင် အသုံးပြု

နှင့် ပါတယ်။

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

ပြီး တဲ့ အခါ Host ကွန်ပျူးတာနဲ့ BBB နဲ့ အချင်းချင်း အပြန်အလှန် ဆက်သွယ် ကြည့်ပါမယ်။ အဲဒီ အတွက် FTDI ကြိုး တစ်ချောင်း သုံးပြီး ပဲ ၈.၁၅ အတိုင်း ဆက်သွယ် လိုက်ပါမယ်။



ပဲ ၈.၁၅: BeagleBone Black ကို FTDI ကြိုးဖြင့် ဆက်သွယ်ခြင်း။

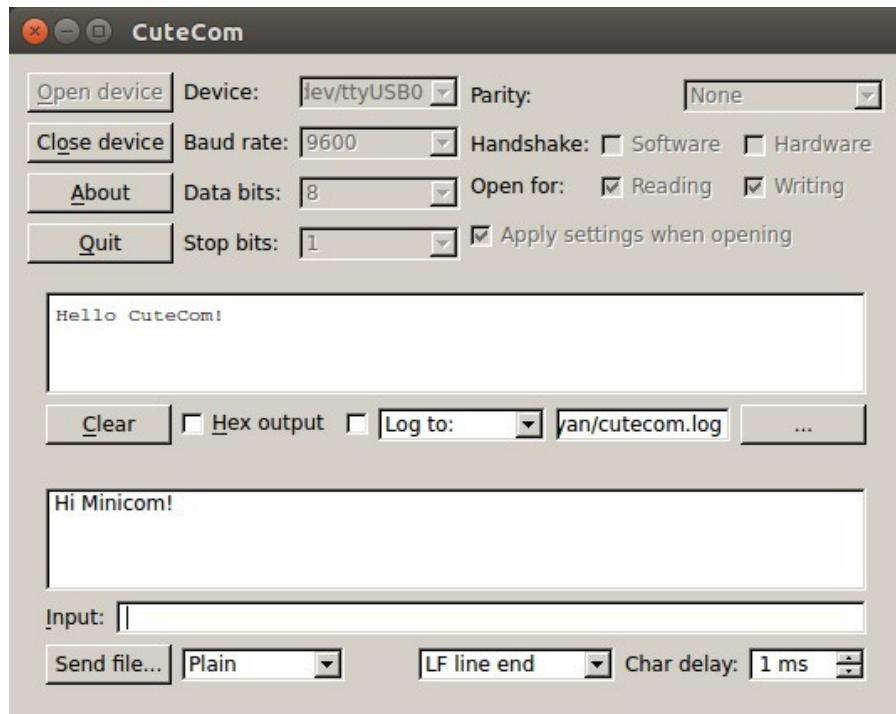
ပြီးတဲ့ အခါ host ကွန်ပျူးတာ နဲ့ BBB တို့ကြား အပြန်အလှန် ဆက်သွယ် ပေးပို့လို ရတာကို ပဲ ??မှာ ပြထားတဲ့ အတိုင်း တွေ့နှင့် ပါတယ်။ Minicom မှာ ရိုက်ထည့် တဲ့ စာလုံးကို ပေါ်ချင်ရင် local echo ကို enable လုပ်နိုင် ပါတယ်။ CTRL-A ကို နှိပ်ပြီး တဲ့ အခါ z ကို နှိပ်လိုက်ပြီး၊ command summary ပေါ်လာရင် e ကို နှိပ်လိုက် တဲ့ အခါ local echo ON သွား ပါလိမ့်မယ်။ ထွက်ချင်ရင် တော့ CTRL-A → z → x → enter ကို နှိပ် နှင့် ပါတယ်။

```
yan@yanhp: ~
Welcome to minicom 2.6.1
OPTIONS: I18n
Compiled on Apr 24 2017, 18:35:09.
Port /dev/tty04

Press CTRL-A Z for help on special keys

Hi Minicom!
```

ပုံ ၈.၁၆: SSH တွင် MiniCom ဖြင့် ဆက်သွယ်ခြင်း။



ပုံ ၈.၁၇: Host computer တွင် CuteCom သုံး၍ BBB နှင့်လှမ်းဆက်သွယ်ခြင်း။

၈.၅ 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 သူနဲ့ သက်ဆိုင်ရာ ttyS0, ttyUSB0 စတဲ့ device file ကို ဖွင့်တာ၊ ရေးတာ၊ ဖတ်တာ တွေကို လုပ်နိုင် ပါတယ်။ အဲဒါ တွေ အတွက် terminal နဲ့ ပတ်သက် တဲ့

သတ်မှတ် ချက်တွေ ပါတဲ့ termios.h ဖိုင် ထိန်းချုပ်မှူ နဲ့ဆိုင်တဲ့ fcntl.h ဗုဏ် စနစ်တွေ အတွက် unistd.h စတဲ့ header ဖိုင်တွေ ကို ထည့် ပါမယ်။

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

Serial port ရဲ configuration တွေကို struct termios ဆိုတဲ့ data structure ကို သုံးပြီး သတ်မှတ် နိုင် ပါတယ်။ အဲဒီ မှာ data bits, parity, stop bits စော့ တွေကို လို 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 ဖွံ့ဖြိုး အခါ open ကို သုံးပြီး အောက်က အတိုင်း ဖွံ့ဖြိုးနိုင် ပါတယ်။ Read/Write အတွက် Non-blocking mode နဲ့ ဖွံ့ဖြိုးမယ် လို ဆိုလို တာပါ။ အဲလို မဟုတ်ပဲ Blocking mode ဆိုရင် ရေးတဲ့ ဖတ်တဲ့ အခါ လုပ်ဆောင် မှု မပြီး မချင်း ပရိုဂရမ် က ရပ်စောင့် နေ ပါမယ်။

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

Serial port ကို ဒေတာ တွေ ရေးဖို့ ဖတ်ဖို့ အတွက် တော့ write နဲ့ read ဖန်ရှင် တွေကို သုံးနိုင် ပါတယ်။ Device file ရယ် ရေးလို သိမ်းလို တဲ့ character buffer ရယ် ရေးဖတ်မယ့် အရေ အတွက် နဲ့ အောက်ပါ အတိုင်း သုံးလို ရ ပါတယ်။ အမှန် တကယ် ရေးလို ဖတ်လို ရလိုက်တဲ့ အရေ 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;
```

နမူနာ အနေနဲ့ serial port ကို အသုံးပြုတဲ့ ရုံးရှင်းတဲ့ SerialSimple.cpp ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၈.၃ 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/tty04"
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

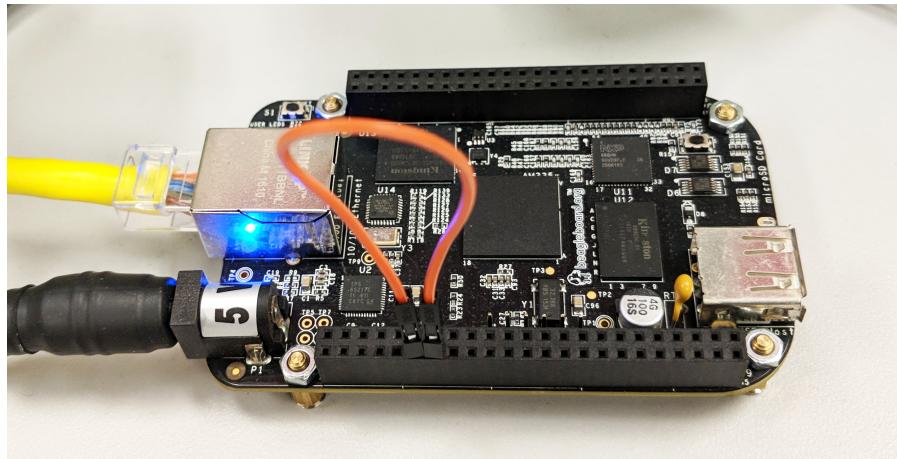
```

```

63     return 0;
64 }
```

စာရင်း ၈.၃: SerialSimple.cpp

သူကို စမ်းကြည့် ဖို့ အတွက် ပုံ ၈.၈ မှာ ပြထား သလို BBB ရဲ့ UART4 ရဲ့ TX pin နဲ့ RX pin ကို loop back ပြန်ဆက် ပြီး run ကြည့်နိုင် ပါတယ်။ ရလာ တဲ့ ရလဒ် ကို ပုံ ၈.၉ မှာ ပြထား ပါတယ်။



ပုံ ၈.၈: TX နှင့် RX ကို loop back ဆက်သွယ်ခြင်း။

```

debian@beaglebone:~/SerialSimple$ ./SerialSimple
Opening serial port.
Port opened successfully.
Writing: Test

Written: 6
Received: Test

Read: 6
Closing serial port.
debian@beaglebone:~/SerialSimple$
```

ပုံ ၈.၉: SerialSimple.cpp ၏ ရလဒ်။

၈.၅.၁ Linux နှင့် Windows အတွက် Serial Class

BBB အတွက် ပါမက ပဲ သူနဲ့ အပြန် အလှန် ဆက်သွယ်မယ့် host computer တွေ အတွက် ပါ serial port ကို သုံးဖို့ လိုတတ် ပါတယ်။ အဲဒီ အတွက် Linux ။ Windows စက်တွေ နဲ့ BBB အတွက် ပါ သုံးနိုင်တဲ့ ကျွန်ုတ် ဖန်တီး ထားတဲ့ C++ class library လေး တစ်ခု ကို နမူနာ ပြောချင် ပါတယ် [Lab17; Den95]။

ပထမ နမူနာ အနေနဲ့ ရုံးရှင်းတဲ့ 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 ([Serial.h](#)) ကို နောက်ဆက်တဲ့ B - စာရင်း [၂.၃](#) မှာ တွေ့နိုင် ပါတယ်။

၈.၇.၂ Console

ကွန်ပျိုတာ ရဲ့ serial port ကို ဖွင့်ပြီး ဒေတာ ပို့ ပြီးတော့ character တစ်လုံး ကို ပြန် ဖတ်ပြ တဲ့ ရုံးရှင်းတဲ့ [serialcon.cpp](#) ဆိုတဲ့ နမူနာ လေးကို စာရင်း [၈.၄](#) 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 "Serial.h"
9 using namespace std;
10 int main()
11 {
12
13 #if defined (__WIN32__) || defined(_WIN32) || defined(WIN32) || defined(
14     __WINDOWS__)
15     Serial com("\\\\\\.\\COM1",9600,8,'N',1); //Windows
16 #else
17     Serial com("/dev/ttyS0",9600,8,'N',1); //Linux
18 #endif
19
20     printf("Opening port %s.\n",com.GetPort().c_str());
21     if (com.Open() == 0) {
22         printf("OK.\n");
23     }
24     else {
25         printf("Error.\n");
26     }
27 }
```

```

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 ၈.၂: serialcon.cpp

အစ မှာ #include "Serial.h" လို့ ကြော်ပြီး Serial class ကို ထည့်လိုက် ပါတယ်။ ပြီးတော့ 'com' လို့ 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 က 'N','E','O','M', သို့ 'S' တို့ကို none, even, odd, mark, နဲ့ space တို့ အတွက် သုံးလို့ ရ ပါတယ်။

Comm port ကို ဖွင့်ဖို့ အတွက် com.Open() ကို သုံးနိုင် ပါတယ်။ Write နဲ့ WriteChar methods တွေကို သုံးပြီး null terminated string ဒါမူမဟုတ် character တစ်လုံး ကို ပို့နိုင် ပါတယ်။ ReadChar method နဲ့ character ကို ဖတ်တဲ့ အခါ စောင့် မနေ ပဲ non-blocking ပုံစံ နဲ့ ဖတ်မှာ ဖြစ်တဲ့ အတွက် character ရှိ မနေရင် successFlag က false လို့ ပြ ပါမယ်။ Close method နဲ့ com ကို ပိတ် ပါတယ်။

Control နဲ့ status လိုင်းတွေ ကိုလည်း အသုံးပြု နိုင် ပါတယ်။ RTS နဲ့ DTR lines တွေကို control လုပ်နိုင် ပြီး၊ CTS တို့၊ DSR တို့လို့ Status lines တွေကိုလည်း ဖတ်နိုင်ပါတယ်။ ဒါ ပရိုဂရမ် "serial-

୧ୟ. PROGRAMMING SERIAL PORT

“con.cpp” ကို Ubuntu Linux ပေါ်မှာ အောက်က အတိုင်း compiled လုပ်၊ run လုပ်နိုင် ပါတယ်။

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

Windows ပေါ်မှာတော့ Visual Studio ဒါမှုမဟုတ် tdm-gcc တို့ mingw တို့လို compiler တစ်ခုခဲ့ကို သုံးပြီးအောက်က လိုမျိုး compile လုပ်ပြီး run နိုင် ပါတယ်။

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

၈၅၃ GUI

နောက်ထပ် နမူနာ တစ်ခု အနေနဲ့ wxWidgets နဲ့ GUI application တစ်ခု ဖန်တီး ပြီး serial port ကို အသုံးပြု တဲ့ အကြောင်း ဆွေးနွေး ပါမယ်။ File menu ကနေ သုံးမယ့် serial port ၊ baud rate စတာ တွေကို ရွေးချယ် သတ်မှတ် လို ရပြီး၊ လက်ခံ ရရှိ တဲ့ ဒေတာ တွေကို text box မှာ ဖော်ပြ ပေးနိုင် ပါတယ်။ အပေါ်ဘက် က text box ထဲမှာ ပို့ချင် တဲ့ text ကို ထည့်ပြီး send လလှတ် ကို နှိပ်ပြီး ပို့နိုင် ပါတယ်။ Control နဲ့ status လိုင်းတွေ ကို အသုံးပြု ပုံကိုပါ ပြထား ပါတယ်။ ဒါ [wxserial.cpp](#) ဆိုတဲ့ နမူနာ ပရိုဂရမ် ကို စာရင်း ၈.၅ မှာ ဖော်ပြ ထား ပါတယ်။

```
1 //File: wxserial.cpp
2 //Description: Serial communication for wxWidgets
3 //WebSite: http://cool-emerald.blogspot.sg/2017/05/serial-port-programming-in
4 //          -c-with.html
5 //MIT License (https://opensource.org/licenses/MIT)
6 //Copyright (c) 2017 Yan Naing Aye
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
30 // Define a new application type, each program should derive a class from
31 // wxApp
32 class MyApp : public wxApp
33 {
34     public:
35         virtual bool OnInit();
36 };
37
38 class MyFrame : public wxFrame
39 {
40     public:
41         // ctor(s)
42         MyFrame(const wxString& title);
43         wxButton *btnSend;
44         wxTextCtrl *txtSend;
45         Serial com;
46         wxTimer m_timer;
47         wxTextCtrl *txtRx;
48         wxCheckBox *chkRTS;
```

១៩. PROGRAMMING SERIAL PORT

JOR

```
48 wxCheckBox *chkDTR;
49 wxCheckBox *chkCTS;
50 wxCheckBox *chkDSR;
51 wxCheckBox *chkRI;
52 wxCheckBox *chkCD;

// 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};
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 loop and the application will run. If we returned false here, the
131     // application would exit immediately.
132     return true;
133 }
134
135 // frame constructor
136 MyFrame::MyFrame(const wxString& title)
137     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(390, 280),
138               wxDEFAULT_FRAME_STYLE ^ wxRESIZE_BORDER), m_timer(this, ID_TIMER)
139 {
140     // set the frame icon
141     SetIcon(wxICON(sample));
142
143 #if wxUSE_MENUS
144     // create a menu bar
145     wxMenu *fileMenu = new wxMenu();
146
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     "));
```

```
154     fileMenu->Append(Serial_Open, wxT("&Open\tAlt-0"), wxT("Open serial port"
155     ));
156     fileMenu->Append(Serial_Close, wxT("&Close\tAlt-C"), wxT("Close serial port
157     "));
158     editMenu->Append(Txt_Clear, wxT("Clea&r\tAlt-R"), wxT("Clear text"));
159     fileMenu->Append(Serial_Port, wxT("&Serial Port\tAlt-S"), wxT("Select
160     serial port"));
161     fileMenu->Append(Serial_Baud, wxT("&Baud Rate\tAlt-B"), wxT("Set baud rate"
162     ));
163     fileMenu->Append(Serial_DataSize, wxT("&Data Size\tAlt-D"), wxT("Set data
164     size"));
165     fileMenu->Append(Serial_Parity, wxT("&Parity\tAlt-P"), wxT("Set parity"));
166     fileMenu->Append(Serial_StopBits, wxT("S&top Bits\tAlt-t"), wxT("Set stop
167     bits"));
168     fileMenu->Append(Minimal_Quit, wxT("E&xit\tAlt-X"), wxT("Quit this program"
169     ));
170
171     // now append the freshly created menu to the menu bar...
172     wxMenuBar *menuBar = new wxMenuBar();
173     menuBar->Append(fileMenu, wxT("&File"));
174     menuBar->Append(editMenu, wxT("&Edit"));
175     menuBar->Append(helpMenu, wxT("&Help"));
176
177     // ... and attach this menu bar to the frame
178     SetMenuBar(menuBar);
179 #endif // wxUSE_MENU
180
181 #if wxUSE_STATUSBAR
182     // create a status bar just for fun (by default with 1 pane only)
183     CreateStatusBar(2);
184     SetStatusText(wxT("Serial Communication"));
185 #endif // wxUSE_STATUSBAR
186     btnSend = new wxButton(this, Button_Send, wxT( "Send"), wxDefaultPosition,
187     wxDefaultSize);
188     txtSend = new wxTextCtrl(this, Txt_Send,wxT("Hello !"),wxPoint(120,5),wxSize
```

0.9. PROGRAMMING SERIAL PORT

JÖQ

```
(250,25));  
182 //lblRx = new wxStaticText(this, ID_LBLRX, wxT("Rx:"), wxPoint(5, 75),  
183 wxSize(35, 25));  
184 txtRx = new wxTextCtrl(this, Txt_Rx, wxT(""), wxPoint(5, 35), wxSize(365,  
125), wxTE_MULTILINE);  
184 chkRTS = new wxCheckBox(this, Chk_RTS, wxT("RTS"), wxPoint(5, 170),  
wxDefaultSize);  
185 chkDTR = new wxCheckBox(this, Chk_DTR, wxT("DTR"), wxPoint(55, 170),  
wxDefaultSize);  
186 chkCTS = new wxCheckBox(this, ID_CHKCTS, wxT("CTS"), wxPoint(155, 170),  
wxDefaultSize);  
187 chkDSR = new wxCheckBox(this, ID_CHKDSR, wxT("DSR"), wxPoint(205, 170),  
wxDefaultSize);  
188 chkRI = new wxCheckBox(this, ID_CHKRI, wxT("RI"), wxPoint(255, 170),  
wxDefaultSize);  
189 chkCD = new wxCheckBox(this, ID_CHKCD, wxT("CD"), wxPoint(305, 170),  
wxDefaultSize);  
190  
191 Connect(Button_Send, wxEVT_COMMAND_BUTTON_CLICKED, wxCommandEventHandler(  
MyFrame::OnSend));  
192 Connect(Minimal_About, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::OnAbout));  
193 Connect(Minimal_Quit, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::OnQuit));  
194 Connect(Serial_Open, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::OnOpen));  
195 Connect(Serial_Close, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::OnClose));  
196 Connect(Serial_Port, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::SelPort));  
197 Connect(Serial_Baud, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::SetBaud));  
198 Connect(Serial_DataSize, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
(MyFrame::SetDataSize));  
199 Connect(Serial_Parity, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(  
MyFrame::SetParity));
```

```
200     Connect(Serial_StopBits, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler
201         (MyFrame::SetStopBits));
202     Connect(Timer1, wxEVT_TIMER, wxTimerEventHandler(MyFrame::OnTimer));
203     Connect(Txt_Clear, wxEVT_COMMAND_MENU_SELECTED, wxCommandEventHandler(
204         MyFrame::ClearText));
205     Connect(Chk_RTS, wxEVT_COMMAND_CHECKBOX_CLICKED, wxCommandEventHandler(
206         MyFrame::OnChkRTS));
207     Connect(Chk_DTR, wxEVT_COMMAND_CHECKBOX_CLICKED, wxCommandEventHandler(
208         MyFrame::OnChkDTR));
209 //Bind(wxEVT_MENU, &MyFrame::OnClose, this, Serial_Close);
210 m_timer.Start(250);
211 chkCTS->Disable();
212 chkDSR->Disable();
213 chkRI->Disable();
214 chkCD->Disable();
215 }
216
217 // event handlers
218 void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
219 {
220     // true is to force the frame to close
221     Close(true);
222 }
223 void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
224 {
225     wxMessageBox(wxString::Format(
226             wxT("Serial Communication! \n ")
227             wxT("Author: Yan Naing Aye \n ")
228             wxT("Web: https://github.com/yan9a/serial")
229             ),
230             wxT("About Serial Comm"),
231             wxOK | wxICON_INFORMATION,
232             this);
233 }
```

1.9. PROGRAMMING SERIAL PORT

JO3

```
232
233 void MyFrame::OnOpen(wxCommandEvent& WXUNUSED(event))
234 {
235     if(com.Open()) txtRx->AppendText(wxString::Format(wxT("Error opening port %s.\n"), com.GetPort()));
236     else txtRx->AppendText(wxString::Format(wxT("Port %s is opened.\n"), com.GetPort()));
237 }
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
245 void MyFrame::SelPort(wxCommandEvent& WXUNUSED(event))
246 {
247     if (com.IsOpened()) {
248         txtRx->AppendText(wxString::Format(wxT("Close Port %s first.\n"), com.GetPort()));
249     }
250     else {
251         wxString cdev=wxString::Format(wxT("%s"), com.GetPort());
252         wxString device = wxGetTextFromUser(wxT("Enter the port"), wxT("Set Port"), cdev);
253         string str = device.ToStdString();
254         if (str.length() > 0) {
255             com.SetPort(str);
256         }
257
258         txtRx->AppendText(wxString::Format(wxT("Port: %s\n"), com.GetPort()));
259     }
260 }
```

```
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 #if defined(__WINDOWS__)
271         wxString parity = wxGetTextFromUser(wxT("Enter the parity ( N, E, O, M,
272             or S )"), wxT("Set Parity"), cdev);
273 #else
274         wxString parity = wxGetTextFromUser(wxT("Enter the parity ( N, E, or O )
275             ), wxT("Set Parity"), cdev);
276 #endif
277
278         string pstr = parity.ToStdString();
279         if (pstr.length() > 0) {
280             com.SetParity(pstr.at(0));
281         }
282         txtRx->AppendText(wxString::Format(wxT("Parity: %c\n"), com.GetParity()))
283     ;
284 }
285
286 void MyFrame::SetBaud(wxCommandEvent& WXUNUSED(event))
287 {
288     if (com.IsOpened()) {
289         txtRx->AppendText(wxString::Format(wxT("Close port %s first.\n"), com.
290             GetPort()));
291     }
292     else {
293         long n = wxGetNumberFromUser(wxT("Enter the baud rate"), wxT("Baud rate")
294             , wxT("Set Baud Rate"), com.GetBaudRate(), 0, 1000000);
295         if (n >= 0) {
296             com.SetBaudRate(n);
297         }
298     }
299 }
```

```
292     }
293 
294     txtRx->AppendText(wxString::Format(wxT("Baud rate: %ld\n"), com.
295         GetBaudRate())));
296 
297 void MyFrame::SetDataSize(wxCommandEvent& WXUNUSED(event))
298 {
299     if (com.IsOpened()) {
300         txtRx->AppendText(wxString::Format(wxT("Close port %s first.\n"), com.
301             GetPort()));
302     }
303     else {
304         long n = wxGetNumberFromUser(wxT("Enter the data size"), wxT("Data Size")
305             , wxT("Set Data Size"), com.GetDataSize(), 5, 8);
306         if (n >= 0) {
307             com.SetDataSize(n);
308         }
309         txtRx->AppendText(wxString::Format(wxT("Data size: %ld\n"), com.
310             GetDataSize()));
311     }
312 }
313 
314 void MyFrame::SetStopBits(wxCommandEvent& WXUNUSED(event))
315 {
316     if (com.IsOpened()) {
317         txtRx->AppendText(wxString::Format(wxT("Close port %s first.\n"), com.
318             GetPort()));
319     }
320     else {
321         long n = wxGetNumberFromUser(wxT("Enter the number of stop bits"), wxT("Data
322             Size"), wxT("Set stop bits"), long(com.GetStopBits()), 1, 2);
323         if (n > 0) {
324             com.SetStopBits(float(n));
325         }
326         txtRx->AppendText(wxString::Format(wxT("Stop bits: %ld\n"), long(com.
```

```
        GetStopBits(0))) ;  
322    }  
323}  
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}  
348  
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())) {
```

```

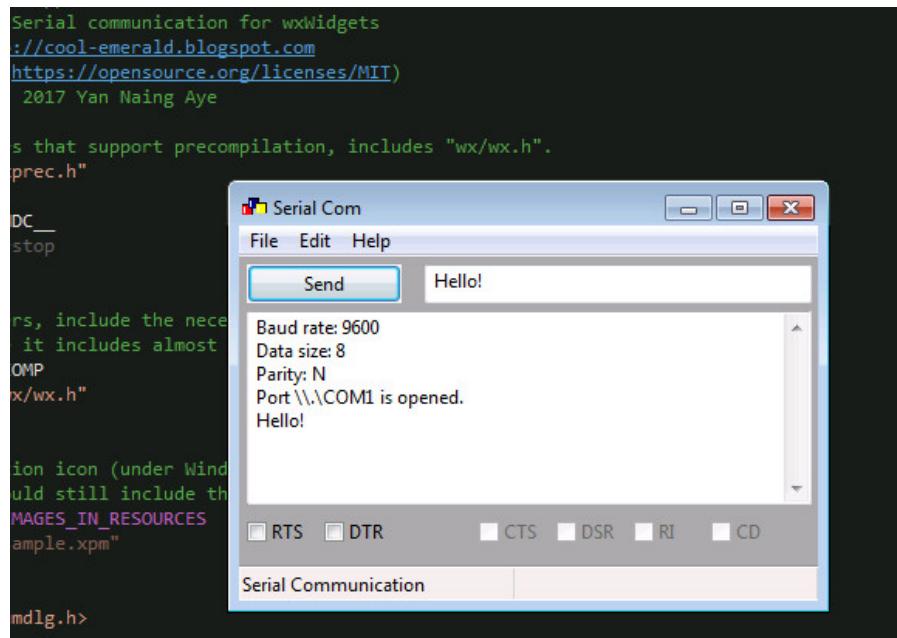
357     txtRx->AppendText(wxT("RTS error.\n"));
358 }
359 }
360
361 void MyFrame::OnChkDTR(wxCommandEvent& WXUNUSED(event))
362 {
363     if (!com.SetDTR(chkDTR->IsChecked())) {
364         txtRx->AppendText(wxT("DTR error.\n"));
365     }
366 }
367
368 void MyFrame::UpdateCommStatus()
369 {
370     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 }
```

စာရင်း ၈၅: wxserial.cpp

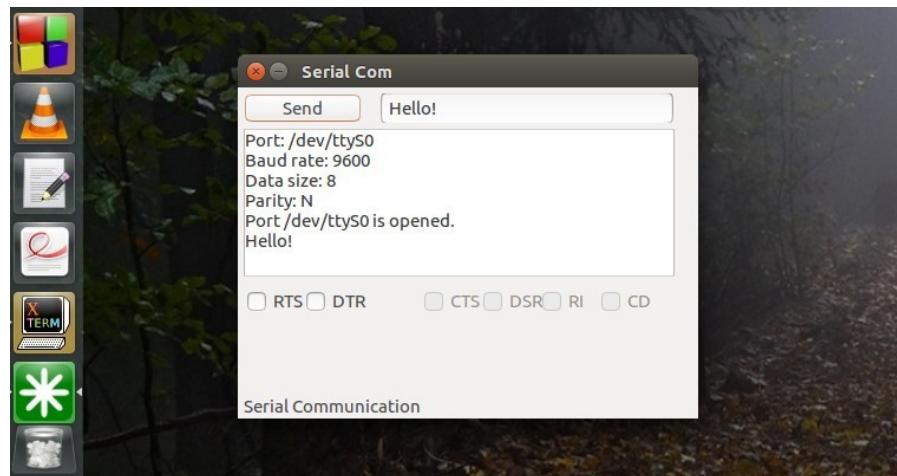
Windows ပေါ်မှာ Visual Studio 2017 နဲ့ wxserial.cpp ကို run လိုက်တဲ့ အခါ ပုံ ၈.၂၀ မှာ ပြထားသလို တွေ့နိုင် ပါတယ်။ အဲဒီ wxserial.cpp program ကိုပဲ ဘာမှ မပြင်ပဲ Debian အခြေဖြေ Linux တွေပေါ်မှာ အောက်ကအတိုင်း built လုပ်၊ run လုပ်လိုက်ပါတယ်။ ပရိုဂရမ် ရဲ့ GUI ကို ပုံ မှာ တွေ့နိုင် ပါတယ်။

```

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



ပုံ ၈.၂၁: Serial class သုံးထော်သော wxWidgets GUI application ကို Visual Studio 2017 ဖွင့် run ခြင်း။



ပုံ ၈.၂၃: Serial class သုံးထော်သော wxWidgets GUI application ကို Ubuntu ပေါ်တွင် run ခြင်း။

အကိုးအကားများ

အကိုးအကားများ

- [Cam13] Cameon. BeagleBone Serial Ports / UART. 2013. url: <http://beaglebone.cameleon.net/home/serial-ports-uart>.
- [Den95] Allen Denver. Serial Communications. 1995. url: <https://msdn.microsoft.com/en-us/library/ff802693.aspx>.
- [Eli17b] Elinux. RPi Serial Connection. 2017. url: https://elinux.org/RPi_Serial_Connection.
- [Eli18] Elinux. Beagleboard:BeagleBoneBlack Debian:U-Boot Overlays. 2018. url: https://elinux.org/Beagleboard:BeagleBoneBlack_Debian_U-Boot_Overlays.
- [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.

၂၁၆

အခန်း ၈. SERIAL PORT ကိုသုံးခြင်း

အခန်း ၉

Internet of Things

အီလက်ထရောနစ် ပစ္စည်း တွေဖြစ်တဲ့ အမိမ 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 အောက်ခံ platform အတွက် ပူစရာ မလိုပဲ အလွယ် တကူ အသုံးပြန်စိုင်တဲ့ socket class ပါ ပါတယ်။ အဲဒီ class ကို မတူညီတဲ့ ပုံစံ နည်းလမ်း အမျိုးမျိုး နဲ့ အသုံးပြန်စိုင်ပြီး အသုံးပြုပဲ နမူနာ တရာ့ကို အောက်မှာ ဆက်ပြီး ဖော်ပြထား ပါတယ်။

၆.၁ 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 ရှိရှင်းလွယ်ကူတဲ့ 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     wxCommandEventHandler(MyFrame::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။

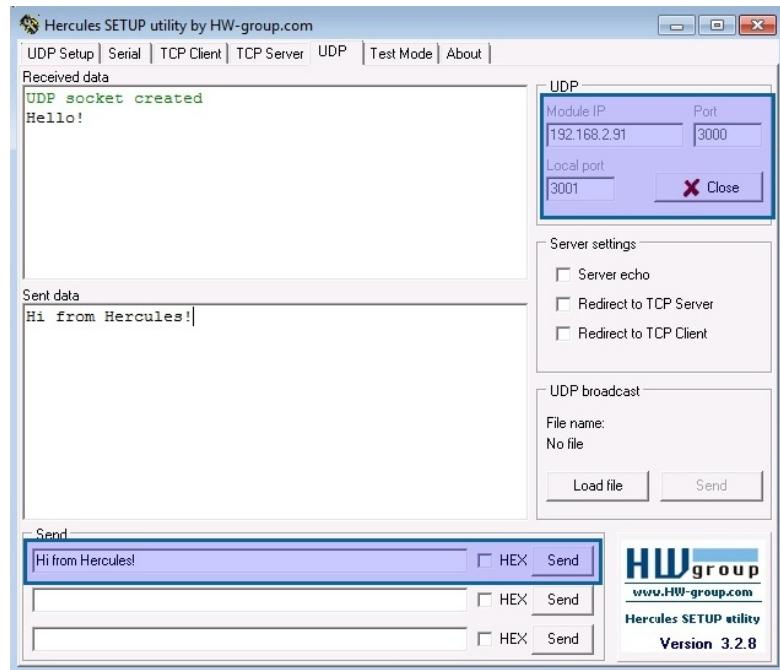
```
n = sock->RecvFrom(addr, buf, sizeof(buf)).LastCount();
```

ဒေတာ ပိုမို အတွက် ပိုမယ့် remote host ရဲ့ address နဲ့ port number ကို သတ်မှတ်ပြီး၊ SendTo method နဲ့ ပို့နိုင် ပါတယ်။

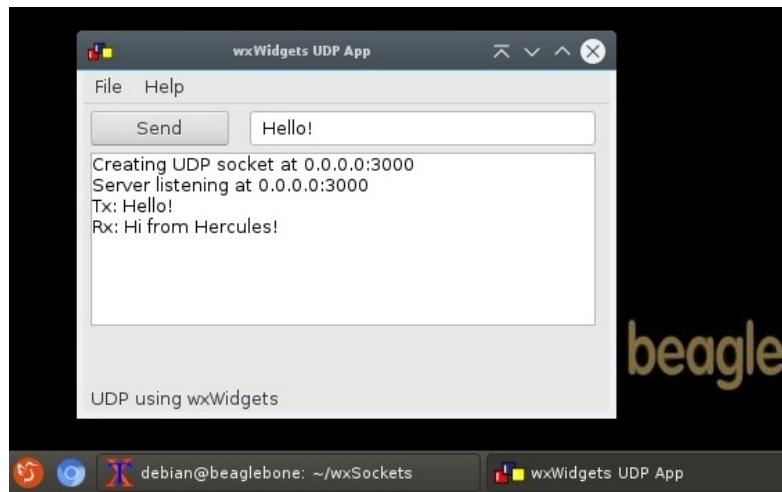
```
IPaddress raddr;
raddr.Hostname("localhost");
raddr.Service(3001);
sock->SendTo(raddr, buf,n)
```

ဒီ ပရိုဂရမ် ကို စမ်းကြည့်ဖို့ အတွက် https://www.hw-group.com/products/hercules/index_en.html 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။



ပုံ ၉.၂: UDP ဖြင့် ဒေတာများ ပို့ခြင်း နှင့် လက်ခံခြင်း။

၉.၂ TCP

TCP (Transmission Control Protocol) က အဓိက ကျတဲ့ network protocol တွေထဲမှာ တစု အပါအဝင် ဖြစ်ပါတယ်။ သူက ဒေတာ တွေပို့ တဲ့ အခါ စိတ်ချ ယုံကြည် ရပြီး၊ အစီအစဉ် တကျ ရောက်အောင်၊ အမှား မပါ အောင် ပို့စိုင် ပါတယ်။ TCP နဲ့ ဒေတာ တွေ ပို့စိုင် ဖို့ အရင်ဆုံး connection ကို ချိတ်ဆက်ဖို့ လိုပါတယ်။ အဲဒီ အတွက် သတ်မှတ်ထား တဲ့ port number ကို နားထောင် နေမယ့် server နဲ့ အဲဒီ ကို လုမ်း 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 ပို့စိုင် 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 နှမူနာ [GZ09] တစ်ခု ကို စာရင်း ၉.၂ မှာ ဖော်ပြထား ပါတယ်။

```

1 // 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/server.cpp, 2009.
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 #include "wx/wxprec.h"
17
18 #ifdef __BORLANDC__
19 #pragma hdrstop
20 #endif
21
22 #ifndef WX_PRECOMP
23 #include "wx/wx.h"
24 #endif
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 IPEndPoint;
30 #else
31     typedef 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_RX, wxT(""), wxPoint(5, 5),
131 wxSize(365, 125), wxTE_MULTILINE);
132
133 // Create the address - defaults to localhost:0 initially
134 IPEndPoint addr;
135 addr.AnyAddress();
136 addr.Service(6000);
137 txtRx->AppendText(wxString::Format(wxT("Creating server at %s:%u \n"))
138 ,addr.IPEndPoint(), 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
port !\n")));
}
```

```
146     return;
147 }
148
149 IPEndPoint 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.IPEndPoint(), 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;
214         if (!sockBase->GetPeer(addr))
215         {
216             txtRx->AppendText(wxT("New connection from unknown client accepted.\n"));
```

```
216     );
217 }
218 else
219 {
220     txtRx->AppendText(wxString::Format(wxT("New client connection from %s:%
221         u accepted \n")),
222         addr.IPAddress(), addr.Service()));
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;
248     case wxSOCKET_LOST: txtRx->AppendText(wxT("wxSOCKET_LOST\n")); break;
249     default: txtRx->AppendText(wxT("Unexpected event !\n")); break;
250 }
```

```
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
272 // Read the message
273 wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
274 if (!lenRd) {
275     txtRx->AppendText(wxT("Failed to read message.\n"));
276     return;
277 }
278 else {
279     txtRx->AppendText(wxString::Format(wxT("Read %d bytes.\n"), lenRd));
280 }
281
282 txtRx->AppendText(wxString::Format(wxT("Rx: %s \n"), wxString::FromUTF8(
283 buf, len)));
284
285 // 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);
316 }
```

အစ frame constructor မှာ socket server တစ်ခု ကို အောက်က အတိုင်း သတ်မှတ် လိုတဲ့ 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(6000);

// 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 အရောအတွက် 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 မှတ်ည်ပြီး အမျိုးမျိုး ဖြစ်နိုင် ပါတယ်။ Setting တွေ သတ်မှတ် တာ မမှန်ရင် ပရိုဂရမ် ရပ်သွား နိုင်တာကြောင့် မှန်မှန် ကန်ကန် သတ်မှတ်ဖို့ အရေးကြီးပြီး၊ အဲဒီ အကြောင်း အသေးစိတ်ကို Julian

Smart ရဲ Cross-Platform GUI Programming with wxWidgets [SH06] ဆိုတဲ့ စာအုပ် အခန်း ၁၈ မှာ ဖော်ပြုထားတာကို ဖတ်ကြည့်သင့်ပါတယ်။

ဒီ နမူနာ မှာတော့ ပထမ ဆုံး byte မှာ message ရဲ အရွယ် အစား ကို ပို့ပေးဖို့လိမ့်ပြီး၊ 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 သူရဲ အရွယ် အစား ၂ ကို ရွှေ့ဆုံး byte မှာ ထည့်ပြီး client ဆိုကို အကြောင်း ပြန်ပါ မယ်။

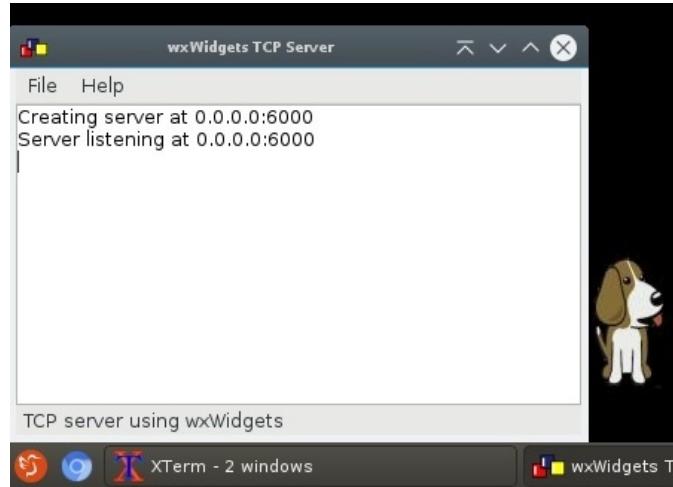
```
len = 2;
buf[0] = 'O';
buf[1] = 'K';
sockBase->Write(&len, 1);
sockBase->Write(buf, len);
```

Connection ကို ဖြတ်တောက် လိုက်တဲ့ အခါ မှာ wxSOCKET_LOST ဆိုတဲ့ OnSocketEvent ဖြစ်တဲ့ အခါမှာ ဖြတ်တောက် သွားတဲ့ socket ကို ဖျက်လိုက် ပါမယ်။

```
sockBase->Destroy();
```

ပရိုဂရမ် ကို အောက်က command တွေနဲ့ build လုပ်ပြီး၊ run လိုက်တဲ့ အခါ ပုံ ၉.၃ မှာ ပြထားသလို client ရဲ 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 ၉.၃: 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 ပို့ပေး၊ server က ပြန်ပို့ တာကို လက်ခံ ဖော်ပြ ပေးတဲ့ `ce_wx_tcp_client.cpp` ဆိုတဲ့ TCP client နမူနာ [Gar99] တစ်ခု ကို စာရင်း ၉.၃ 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,"
```

```
11 // Pearson Education, Inc. 2006. ISBN: 0-13-147381-6.
12
13 // For compilers that support precompilation, includes "wx/wx.h".
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#if wxUSE_IPV6
27typedef wxIPV6address IPEndPoint;
28#else
29typedef wxIPV4address IPEndPoint;
30#endif
31
32#ifndef wxHAS_IMAGES_IN_RESOURCES
33#include "./sample.xpm"
34#endif
35
36class MyApp : public wxApp
37{
38public:
39    virtual bool OnInit();
40};
41
42// Define a new frame type: this is going to be our main frame
43class MyFrame : public wxFrame
44{
45public:
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_MENUS
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                           wxDefaultPosition, wxDefaultSize);
147     txtSend = new wxTextCtrl(this, ID_TXTSEND, wxT("Hello!"),
148                           wxDefaultPosition, wxDefaultSize);
149     txtRx = new wxTextCtrl(this, ID_TXTRX, wxT(""),
150                           wxDefaultPosition, wxDefaultSize, wxTE_MULTILINE);
151
152     Connect(ID_BTNSEND, wxEVT_COMMAND_BUTTON_CLICKED,
153             wxCommandEventHandler(MyFrame::OnSend));
```

```
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     IPEndPoint addr;
216     //addr.AnyAddress();
217     addr.Hostname("localhost");
218     addr.Service(6000);
219     txtRx->AppendText(wxString::Format(wxT("Trying to connect to %s:%u \n"),
220                                         addr.IPEndPoint(), 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
259     // Now we process the event
260     switch (event.GetSocketEvent())
261     {
262         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     // Read the message
280     wxUint32 lenRd = sockBase->Read(buf, len).LastCount();
281     if (!lenRd) {
282         txtRx->AppendText(wxT("Failed to read message.\n"));
283         return;
284     }
285     else {
286         txtRx->AppendText(wxString::Format(wxT("Read %d bytes.\n"), lenRd));
287     }
288
289     txtRx->AppendText(wxString::Format(wxT("Rx: %s \n"),
290                                         wxString::FromUTF8(buf, len)));
291     // Enable input events again.
292     sockBase->SetNotify(wxSOCKET_LOST_FLAG | wxSOCKET_INPUT_FLAG);
293     break;
294 }
295 default:;
296 }
```

```

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။

```

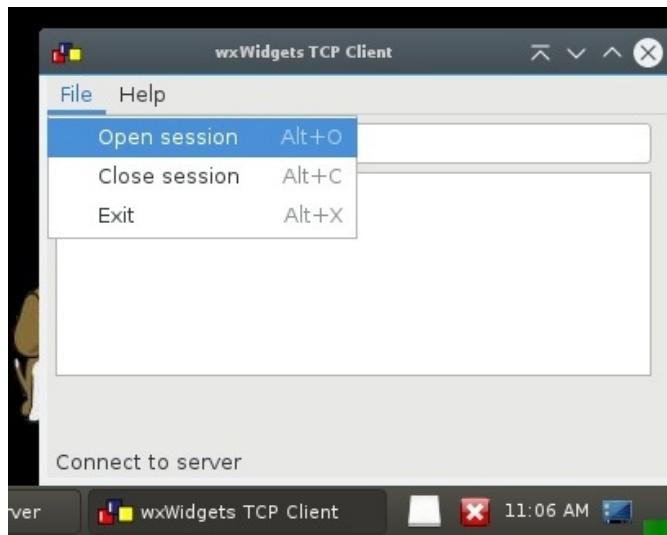
// Create the socket
sock = new wxSocketClient();

// 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 နိုပ်ပြီး 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 ပြုလုပ်ပုံ။

ဆက်သွယ်မှု ပြုလုပ်ဖို့ အတွက် လိပ်စာ နဲ့ port number တွေကို သတ်မှတ်ပြီး Connect ဆိုတဲ့ method ကို သုံးပြီး ဆက်သွယ် နိုင် ပါတယ်။

```
IPaddress addr;
addr.Hostname("localhost");
addr.Service(6000);
sock->Connect(addr, false);
```

ဆက်သွယ် မှု ပြုလုပ်ပြီး တဲ့ အခါ ဒေတာ တွေကို Send ခလုတ် နှိပ်ပြီး ပို နိုင် ပါတယ်။ အောက်မှာ ဖော်ပြ ထားတဲ့ အတိုင်း ပိုမယ့် ဒေတာ တွေရဲ့ ပထမ byte မှာ အရော အတွက် ကို အရင်ထားပြီး buffer ထဲက ဒေတာ တွေကို နောက်က နောက် Write ကိုသုံးပြီး ပို နိုင် 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");
}
}

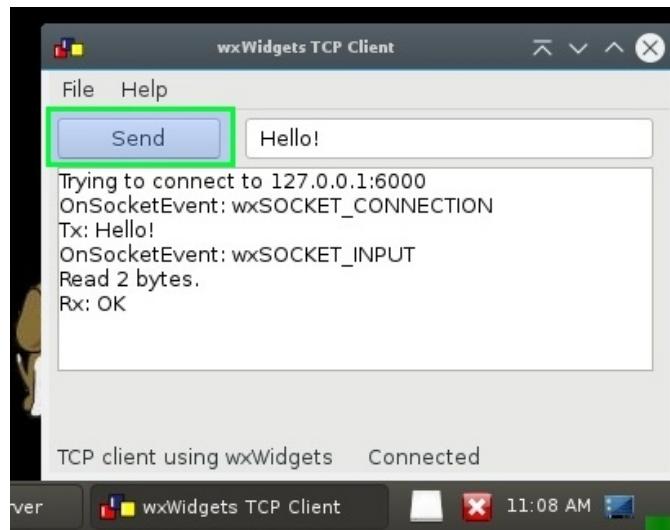
```

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 နိုဝင် နိုင်ပါတယ်။ ဒေတာ တွေကို ပိုပြီး တဲ့ အခါ server က ပြန်ပို့ တာကို ဖော်ပြ ထားတာကို Client ပုံ ၉.၅ နဲ့ server ပုံ ၉.၆ မှာ တွေ့နိုင် ပါတယ်။

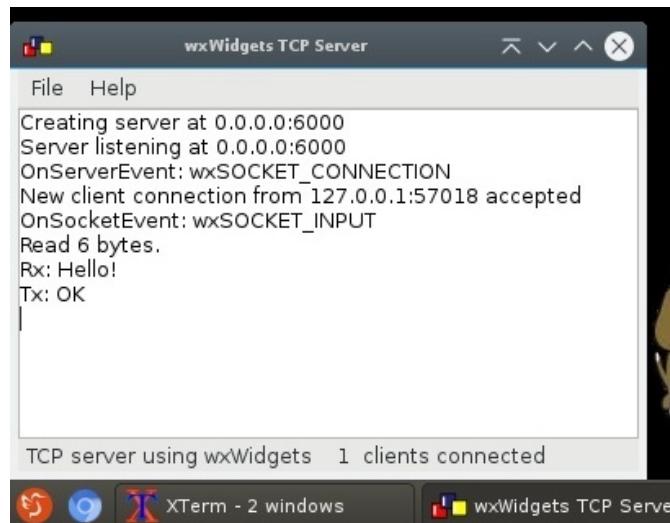
```

wxUint32 lenRd = sockBase->Read(buf, len).LastCount();

```



ပုံ ၉.၅: TCP client ဖြင့် ဒေတာ ပို့ခြင်း နှင့် လက်ခံခြင်း။



ပုံ ၉.၆: TCP server တွင် ဒေတာ လက်ခံရရှိပုံ။

၉.၃ FTP Server တပ်ဆင်ခြင်း

BBB မှာ SSH ရှိ နေတဲ့ အတွက် အပိုင်း ၃.၂ မှာ ပြောခဲ့ သလို SFTP နဲ့ တန်းပြီး ဆက်သွယ် အသုံးပြု လို ရပါတယ်။ အဲလို မဟုတ်ပဲ အကြောင်း အမျိုးမျိုး ကြောင့် ရှိုးရိုး FTP server ကို တပ်ဆင် မယ်

၉.၃. FTP SERVER တပ်ဆင်ခြင်း

၂၅၃

ဆိုရင်လည်း တပ်ဆင် နှင့် ပါတယ်။ FTP server အတွက် vsftpd လို့ ခေါ်တဲ့ Very Secure FTP Daemon ကို အသုံးပြု တဲ့ အကြောင်း ရွေးနွေး 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 မပတ်။ အဲဒါ ကို netstat နဲ့ အောက်ပါ အတိုင်း စစ်ကြည့် နှင့် 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 ထောက်ပုံ ပါတယ်။

| Proto | Recv-Q | Send-Q | Local Address | Foreign Address | State |
|---------------|--------|--------|---------------|-----------------|--------|
| tcp | 0 | 0 | 0.0.0.0:5901 | 0.0.0.0:* | LISTEN |
| 935/Xtightvnc | 0 | 0 | 0.0.0.0:6001 | 0.0.0.0:* | LISTEN |
| 935/Xtightvnc | 0 | 0 | 0.0.0.0:53 | 0.0.0.0:* | LISTEN |
| 944/dnsmasq | 0 | 0 | 0.0.0.0:22 | 0.0.0.0:* | LISTEN |
| 749/sshd | 0 | 0 | :::80 | :::* | LISTEN |
| 1/init | 0 | 0 | :::8080 | :::* | LISTEN |
| 840/apache2 | 0 | 0 | :::21 | :::* | LISTEN |
| tcp6 | 0 | 0 | 28175/vsftpd | :::* | LISTEN |
| tcp6 | 0 | 0 | :::53 | :::* | LISTEN |
| 944/dnsmasq | 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 ပြပြင် 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 ပေးထို အတွက် write_enable=YES ကို ထည့်ဖို့ # ကို ဖျက်ပြီး uncomment လုပ်နိုင် ပါတယ်။ Home directory ကို သတ်မှတ်ဖို့ အတွက် local_root အတွက် path ကို သတ်မှတ် နိုင် ပါတယ်။

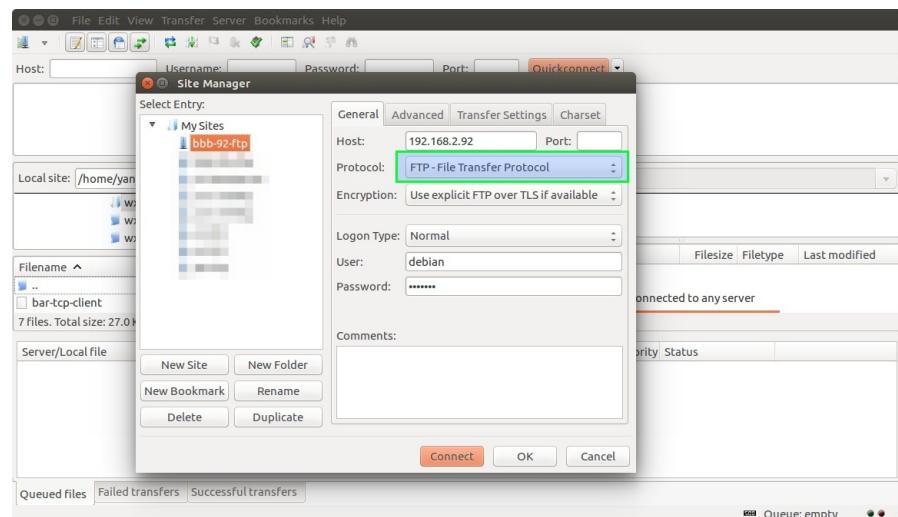
```
local_enable=YES
write_enable=YES
local_root=/home/debian/ftp
```

User တွေ က system တစ်ခု လုံးကို browse လုပ်နိုင် ပြီး၊ သူတို့ အတွက် access ကို home directory မှာပဲ ကန့်သတ် 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
```



ပုံ ၉.၈: FTP server ကို FileZilla client ဖြင့် ဆက်သွယ်ခြင်း။

အဲဒီ နောက် မှာ vsftpd server ကို FileZilla စတဲ့ ftp client တွေ သုံးပြီး ဆက်သွယ် အသုံးပြု နိုင် ပါပြီ (ပုံ ၉.၈)။

၉.၄ Web Server တပ်ဆင်ခြင်း

BBB ကို တြေား ပစ္စည်း ကိရိယာ တွေ ကနေ ဆက်သွယ် ဖို့ အဆင်ပြေပြီး ကောင်းမွန် တဲ့ နည်းလမ်း တစ်ခု ကတေသူ http ကို သုံးပြီး web browser စတာတွေ ကနေ ဆက်သွယ်တဲ့ နည်းပါ။ အဲဒီ အတွက် Apache web server ကို အသုံးပြု တဲ့ အကြောင်း အွေးနေး ချင် ပါတယ် [ras17b]။ BBB မှာ Apache web server တပ်ဆင်ပြီး သား ပါပြီး tcp port 8080 ကို သုံးထား တာကို ထုံးစံ အတိုင်း netstat သုံးပြီး ကြည့်နိုင် ပါတယ်။ Port 80 မှာတော့ init လို့ ပြနေ ပြီး၊ အဲဒါ က bonescript အတွက် သုံးထား တာပါ။

```
$ sudo netstat -npl
```

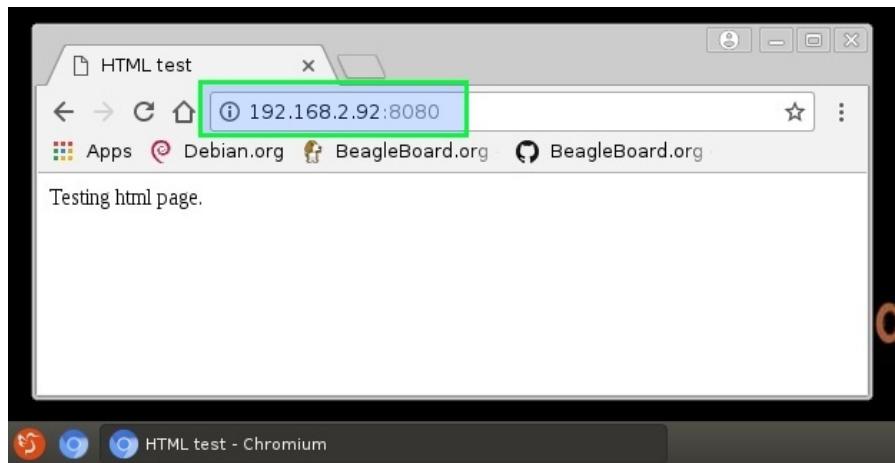
HTML documents တွေ အတွက် root အခန်းက /var/www/html/ ဖြစ်ပြီး၊ web server ကို စမ်းကြည့် ဖို့ အတွက် index.html ဆိုတဲ့ ဖိုင် တစ်ကို nano နဲ့ ဖန်တီး လိုက် ပါမယ်။

```
$ sudo nano /var/www/html/index.html
```

အဲဒီ ဖိုင် ထဲမှာ အောက်က အတိုင်း ရေးပြီး သိမ်းနိုင် ပါတယ်။

```
<html>
<head>
<title>HTML test</title>
</head>
<body>
Testing html page.
</body>
</html>
```

Web browser ဖွင့်ပြီး BBB ရဲ့ IP address : port number ကို ထည့် လိုက် ရင် အဲဒီ html စာမျက်နှာ ပွင့် လာတာ ကို ပုံ ၉.၉ မှာ ပြထား သလို တွေ့နိုင် ပါတယ်။



ပုံ ၉.၉: Web server ကိစစ်ကြည့်ခြင်း။

၉.၄.၁ PHP ကို အသုံးပြုခြင်း

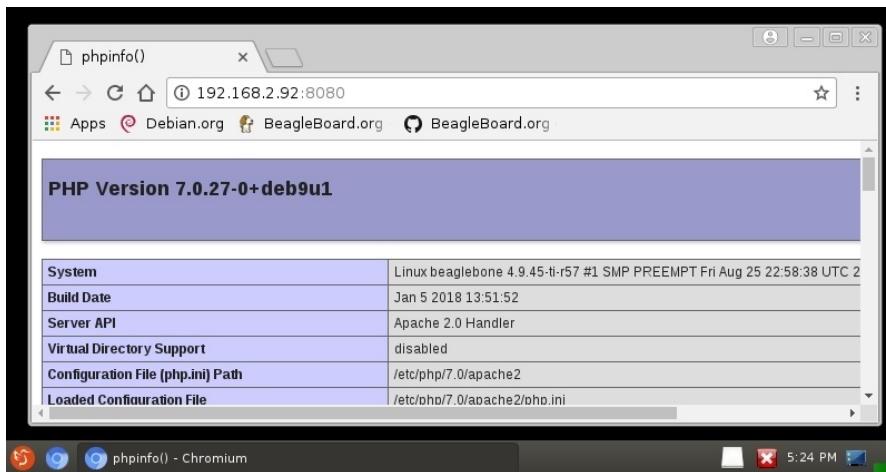
HTML အတွက် root folder ဖြစ်တဲ့ /var/www/html ရဲ့ ownership ကို debian ဖြစ်အောင် ပြောင်းပေးပါမယ်။ PHP ကို တပ်ဆင် လိုက်ပြီး၊ စမ်းသပ် ကြည့်ဖို့ အတွက် မူရင်း index.html ကို ဖျက်ပြီး၊ php ဖိုင် တစ်ခု ကို ဖန်တီး ပါမယ်။

```
$ sudo apt install php libapache2-mod-php
$ sudo chown -R debian /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 ကိုစမ်းသပ်ခြင်း။

အဲဒီလို မဟုတ်ပဲ စက်ထဲ မှာ ရှိတဲ့ အခန်း တစ်ခု ခဲ့ကို root folder အနေနဲ့ သုံးချင်ရင် /var/www/ ထဲက html အခန်း တစ်ခု လုံးကို ဖျက်လိုက်ပြီး symbolic link လုပ်လို့ လည်း ရပါတယ်။

```
$ sudo mv /var/www/html /home/debian/html
$ sudo ln -s /home/debian/html /var/www/html
```

၉.၄.J Bone101 Server ကို Apache ဖြင့် အစားထိုးခြင်း

Web server အတွက် port 80 ကိုပဲ သုံးချင် ရင်တော့ BBB ရဲ့ bone101 ကို ဖြတ်ပစ် လိုက်ပြီး Apache နဲ့ အစားထိုး သုံးနိုင် ပါတယ် [Mol14]။ Bonescript service က အလုပ် လုပ်နေ တာကို အောက်က command တွေနဲ့ ကြည့်နိုင် ပြီး ပိတ်နိုင် ပါတယ် (ပုံ ၉.၁၁)။

```
$ systemctl list-units -t service | grep bonescript
$ sudo systemctl stop bonescript.socket
$ sudo systemctl stop bonescript.service
$ sudo systemctl disable bonescript.socket
$ sudo systemctl disable bonescript.service
```

```
debian@beaglebone:~$ systemctl list-units -t service | grep bonescript
bonescript-autorun.service          loaded active running Bonescript autorun
bonescript.service                  loaded active running Bonescript server
debian@beaglebone:~$ sudo systemctl stop bonescript.socket
debian@beaglebone:~$ sudo systemctl stop bonescript.service
debian@beaglebone:~$ sudo systemctl disable bonescript.socket
Removed /etc/systemd/system/sockets.target.wants/bonescript.socket.
debian@beaglebone:~$ sudo systemctl disable bonescript.service
debian@beaglebone:~$
```

ပုံ ၉.၁၁: PHP ကိုစမ်းသပ်ခြင်း။

အဲဒီ နောက်မှ Apache ရဲ port 8080 နေရာ တွေမှာ 80 နဲ့ အစားထိုးဖို့ configure လုပ်နိုင် ပါတယ်။

```
$ sudo nano /etc/apache2/ports.conf
```

ports.conf ဖိုင် ပွင့် လာတဲ့ အခါ Listen 8080 နေရာ မှာ Listen 80 နဲ့ အစားထိုး ပြီး Apache ကို restart လုပ်နိုင် ပါတယ်။ အဲဒီနောက် BBB ရဲ IP address ကို ဖွင့်လိုက် ရင် bone server အစား Apache web server ဖြစ်သွား ပါပြီ။

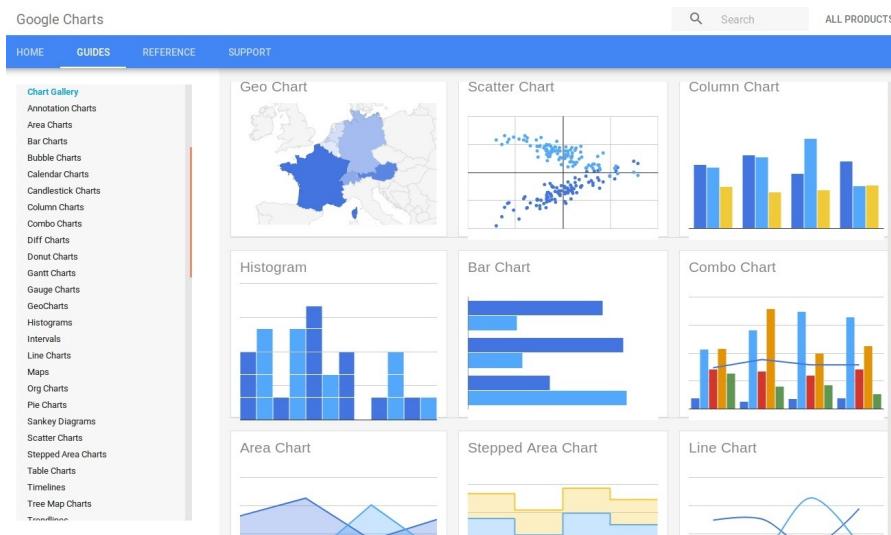
```
$ sudo systemctl restart apache2.service
```

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/) ပါ။ Google chart tools တွေက အစွမ်း ထက်မြက်၊ ရှိုးရှင်းရုံး မကပဲ free လည်း ဖြစ်ပါတယ်။ Data visualization အတွက် chart အမျိုး အစား အများကြီး ပါပြီး၊ HTML5/SVG သန့်သန့် ပဲ သုံးထား တာမို့ ပလက်ဖောင်း အမျိုးမျိုး၊ browser အမျိုးမျိုး မှာ plugins တွေ မလိုပဲ သုံးနိုင် ပါတယ်။ နမူနာ Google chart တရာ့ကို ပုံ ၉.၁၂ မှာ ပြထား ပါတယ်။

၉.၅. GOOGLE CHARTS

၂၀၃



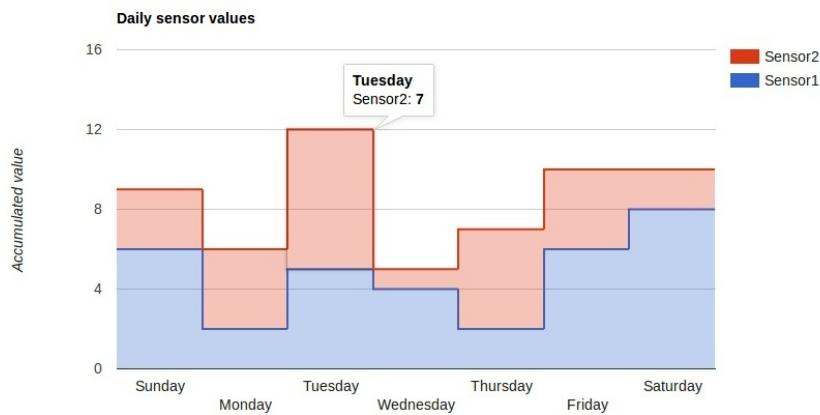
ပုံ ၉.၁၂: Google chart gallery

Google chart တစ်ခု ကို စမ်းသပ် ကြည့်ဖို့ အတွက် အောက်က စာရင်း ၉.၄ မှာ ဖော်ပြ ထားတဲ့ HTML ကုဒ်တွေ ကို gchart.htm စတဲ့ ဖိုင် တစ်ခု အနေနဲ့ သိမ်းပြီး browser တစ်ခု ခုနဲ့ ဖွင့်ကြည့် လိုက်ရင် stepped area chart တစ်ခု ကို ဖော်ပြ ပေးတာ ကို တွေ့နိုင် ပါတယ် (ပုံ ၉.၁၃)။ Stepped area chart အစား bar chart နဲ့ ဖော်ပြ ချင်ရင် အဲဒီ ကုဒ် နမူနာ ထဲက google.visualization.SteppedAreaChart ဆိုတဲ့ နေရာမှာ google.visualization.BarChart လို ပြောင်းသုံး လိုက်ရုံး ပါပဲ။

```
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',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>
```

စုစုပေါင်း ၃.၄: Stepped area chart နမူနာ : gchart.htm



ပုံ ၉.၃: Stepped area chart တစ်ခု ဆွဲခြင်း။

၉.၅.၁ Chart လိုင်သရီထည့်ခြင်း

Google chart ကို သုံးမယ် ဆိုရင် ဝက်ဘ် 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>
```

ပထာမ 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 နှမူနာ မှာတော့ bar, column, line, area, stepped area, bubble, pie, donut, combo, candlestick, histogram, scatter စာတော့ တွေပါတဲ့ corechart ကို သုံးထား ပါတယ်။ ပထာမ argument မှာ ထည့်လိုက်တဲ့ current ၏ latest release ကို သုံးမယ် လို့ ဆိုလို တာပါ။ ပြီးတဲ့ အခါ web page မှာ drawChart ဆိုတဲ့ Javascript function ရှိမယ် လို့ ယူဆ ထားပါတယ်။

Google chart ရဲ့ အားနည်းချက် က off line သုံးလို့ မရ ပါဘူး။ အကယ်၍ off line သုံးချင် ရင်တော့

Chart.js, D3.js စတာ တွေကို သုံးနှင့် ပါတယ်။

၉.၁.၂ ဒေတာများပြင်ဆင်ခြင်း

ဆဲပြု 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 ရဲ့ ပုံစံ ကို စိတ်ကြိုက် ပြင်ဆင် လိုလည်း ရပါတယ်။

```
var options = {
  title: 'Daily sensor values',
  vAxis: {title: 'Accumulated value'},
  isStacked: true
};
```

၉.၁.၃ Chart ကို ဆဲခြင်း

နောက်ဆုံး မှာ အောက်က စာကြောင်းတွေ အတိုင်း သုံးပြီး chart ကို ဆဲနိုင် 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 သုံးဖြီးရိုးရှင်းတဲ့ 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>
```

စာရင်း ၆၂: D3.js bar chart နမူနာ : d3bar.php

d3js.org ကနေ download လုပ်ပြီး ရုံးစွဲတဲ့ d3.min.js ဆိုတဲ့ ဖိုင်ကို d3bar.php နဲ့ folder တစ်ခု ထဲမှာ အတူတူ ထားဖို့ လိုပါတယ်။ နောက်ပြီး ဝက်ဘ် စာမျက်နှာ ဖိုင်ရဲ့ head အပိုင်း မှာ အောက်က စာကြောင်း ကို ထည့်နိုင် ပါတယ်။

```
<script src="d3.min.js"></script>
```

ဒေတာ တွေကို သတ်မှတ် ပြီး bar chart ဆွဲပြဖို့ 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'; });
```

ဒီ နမူနာ မှာ တော့ data ဆိုတဲ့ array ကို တစ်ခါ ထည့် 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

အဲဒီပရိုဂရမ် ကို build လုပ်ဖို့လိုချင် တဲ့ data တန်ဖိုး ရဲ့ index ကို argument အနေနဲ့ ထည့်ပြီး run ဖို့အောက်က နမူနာ command တွေကို သုံးနိုင် ပါတယ်။ အဲဒီမှာ argument အနေနဲ့ 1 ကို သုံးလိုက် တဲ့ အတွက် index 1 ၏ တန်ဖိုး 860 ကို ရှိက်ပြ မှာပါ။

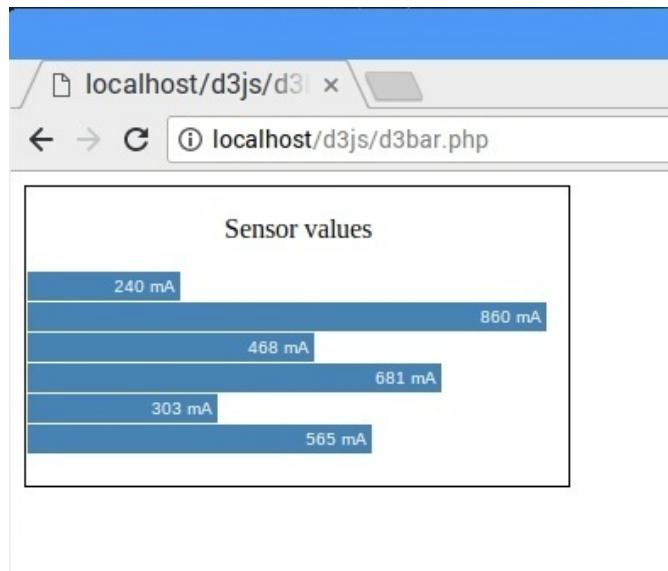
```
$ g++ readsensor.cpp -o readsensor
$ ./readsensor 1
```

Apache web server ရဲ့ home ဖွစ်တဲ့ /var/www/html ထဲမှာ ထည့်ထား တဲ့ အဲဒီပရိုဂရမ် ကို PHP ကနေ လုမ်းပြီး ချိတ်ဆက် အသုံးပြု တဲ့ ကုဒ် အပိုင်းအစ တရီးကို 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 နမူနာ မှာ D3.js နဲ့ ဆွဲလိုက်တဲ့ bar chart ကို ပုံ ၃.၁၄ မှာ ပြထား ပါတယ်။



ံ ၉.၁၄: D3.js ဖြင့် bar chart တစ်ခု အဲခြင်း။

ရှေ့မှာ ဖော်ပြု ခဲ့တဲ့ နည်းတွေ ကို မသုံးပဲ comma separated values တွေကို ဖိုင် တစ်ခု ထဲမှာ စုစုးပြီး PHP ရဲ့ `include` နဲ့ တိုက်ရိုက် ထည့်သုံး မယ်ဆိုရင်လည်း အောက်က ကုဒ် လိုမျိုး အသုံးပြု နိုင် ပါတယ်။

```
var data = [<?php include 'myarr.php';?>];
```

၉.၇ Email ပို့ခြင်း

BBB ကို သုံးပြီး sensor တွေ၊ အမှားပေါ် controller တွေ ပြုလုပ် 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
```

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=beaglebone
rewriteDomain=gmail.com
FromLineOverride=YES
UseSTARTTLS=YES
UseTLS=YES
```

ဖိုင်ကို သိမ်းပြီး ထွက်ပြီး တဲ့ အခါ အောက်က command နဲ့ mail ပိုတာ ကို စမ်းကြည့် နိုင် ပါတယ်။

```
$ echo "Hello world email body" | mail -s "Test Subject" recipientname@domain
.com
```

အကယ်၍ email ပိုတာ မအောင်မြင်ပဲ၊ ပိုတဲ့ account ထဲကို Review blocked sign-in attempt ဆိုတဲ့ email ရောက်လာရင်၊ အဲဒီ ထဲက allowing access to less secure apps ဆိုတဲ့ link ဒါမှ မဟုတ် <https://myaccount.google.com/security> ကို သွားပြီး၊ Allow less secure apps: ON ဖြစ်အောင် ပြောင်းပေး ဖို့လို ပါတယ်။

၉.၇.၁ ဖိုင်ကိုပို့ခြင်း

ဖိုင်ကို email body အနေနဲ့ ပိုချင် ရင်တော့ mpack ကို သုံးလို ရပါတယ်။

```
$ sudo apt install mpack
```

ဥပမာ emailbody.txt ဆိုတဲ့ ဖိုင်တစ်ခု ကို အခန်း တစ်ခုခဲ့ ထဲမှာ ဖန်တီးပြီး စာတွေ ဖြည့်ပြီးရင် အောက်က လိုမျိုး ပို့နိုင် ပါတယ်။

```
$ mpack -s "Test file subject" /home/debian/sendemail/emailbody.txt
recipientname@domain.com
```

၉.၇.၂ C++ ဖြင့်ပို့ခြင်း

ဖော်ပြခဲ့တဲ့ command တွေကို C++ ပရိုကရမဲ့ ထဲမှာ system() ကို သုံးပြီး ခေါ်သုံးလို့ ရပါတယ်။ နမူနာ အနေနဲ့ sendemail.cpp ဆိုတဲ့ ပရိုကရမဲ့ လေးကို စာရင်း ၉.၇ မှာ ပြထားပါတယ်။

```

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++ subject\" /home/debian/sendemail/
8 emailbody.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++ ပရိုကရမဲ့ ဖြင့်ပို့ခြင်း။

သူကို အောက်က အတိုင်း build လုပ်ပြီး၊ run လိုက်တဲ့ အခါ email ပိုပေး တာကို တွေ့ရ ပါလိမ့်မယ်။

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

၉.၈ Watchdog

ကွန်ပျူးတာ စံနစ် တွေ မှားယွင်း၊ ရပ်တန်းမှု မရှိ အောင် Watchdog timer တွေကို အသုံးပြုကြ ပါတယ်။ Watchdog timer ကို အချိန် မလွန်ခင် အချိန်မှန် သွားသွား reset လုပ်ပေး နေဖို့ လိုပါ တယ်။ အဲဒါ ကို watchdog ကို feed ဒါမှ မဟုတ် pat လုပ်တယ် လိုလည်း ခေါ်ပါတယ်။ သတ်မှတ် ထားတဲ့ အချိန် အတွင်း feed လုပ်ဖို့ ပျက်ကွက် ခဲ့ရင် တစ်ခုခဲ့ မှားယွင်း နေပြီ လို့ ယူဆပြီး၊ အဲဒါ စံနစ်ကို watchdog ကနေ reboot ပြန်လုပ် ပေးမှာ ဖြစ်ပါတယ်။ ဥပမာ ကွန်ပျူးတာ hang သွားတယ် ဆိုရင် reset ခလုတ်

ကို ဘယ်သူမှ သွားနိုင် 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 တစ်ခု ကို အောက်မှာ ပြထား ပါတယ်။

```
#define WATCHDOG "/dev/watchdog"
fd = open(WATCHDOG, O_RDWR);
```

Watchdog timer အချိန် ပြည့် သွားမယ့် timeout စကြန် တန်ဖိုး ကို အောက်က အတိုင်း သတ်မှတ်လို့ 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 နေလို့ ရပါတယ်။ မဟုတ်ရင် system က reboot ဖြစ်သွား မှာပါ။ အဲလို့ မဟုတ်ပဲ နောက်ထပ် ပုံမှန် feed လုပ်လို့ ရတဲ့ နည်းကတော့ /dev/watchdog ဆိုတဲ့ ဖိုင်ထဲမှာ 'V' မဟုတ် တဲ့ character တစ်ခုခု ကို ရေးပေး တာပါ။

```
r=ioctl(fd, WDIOC_KEEPALIVE, NULL);
```

Watchdog ကို ရပ်ချင် ရင် 'V' ဆိုတဲ့ character ကို /dev/watchdog မှာ ရေးပြီး၊ ဖိုင်ကို ပိတ်နိုင် ပါတယ်။ Kernel configuration မှာ CONFIG_WATCHDOG_NOWAYOUT ကို enabled လုပ်ထား ရင် တော့ 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 ၉.၈ မှာ ပြထား ပါတယ်။ အဲဒီ class ကို အသုံးပြု ပုံ နမူနာ watchdog.cpp ကို စာရင်း ၉.၉ မှာ တွေ့နိုင် ပါတယ်။

```
1 #ifndef CE_WATCHDOG_H
2 #define CE_WATCHDOG_H
3
4 //References
```

3.9. WATCHDOG

JRC

```
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 class CE_Watchdog {
17     int fd;
18     int interval;
19 public:
20     CE_Watchdog();
21     ~CE_Watchdog();
22     int Begin();
23     int Pat();
24     int Close();
25     int GetInterval();
26     bool IsLastBootByWatchdog();
27 };
28
29 CE_Watchdog::CE_Watchdog()
30 {
31     fd=-1;//invalid handle
32     interval=15;//in seconds
33     Begin();
34 }
35
36 int CE_Watchdog::Begin()
37 {
38     int r=-1;
39     if(fd==(-1)){
40         //if(getuid()!=0){
```

```
41         //printf("You must run this program as root.\n");
42         //}
43
44         if ((fd = open(WATCHDOG, O_RDWR))<0){
45             perror("Error in opening watchdog.\n");
46             return r;
47         }
48         // set the timing interval
49         if (ioctl(fd, WDIOC_SETTIMEOUT, &interval)!=0){
50             perror("Error in setting watchdog interval.\n");
51             return r;
52         }
53         r=0;
54     }
55 }
56
57 CE_Watchdog::~CE_Watchdog()
58 {
59     this->Close();
60 }
61
62 int CE_Watchdog::Pat()
63 {
64     int r=-1;
65     if(fd!=(-1)){
66         r=ioctl(fd, WDIOC_KEEPALIVE, NULL);
67     }
68     return r;
69 }
70
71 int CE_Watchdog::Close()
72 {
73     int r=-1;
74     if(fd!=(-1)){
75         write(fd, "V", 1);
76         r=close(fd);
```

፩.፭. WATCHDOG

JRR

```
77         fd=-1;
78     }
79     return r;
80 }
81
82 int CE_Watchdog::GetInterval()
83 {
84     int v=0;
85     if(fd!=(-1)){
86         if (ioctl(fd, WDIOC_GETTIMEOUT, &v) != 0) {
87             perror("Error in reading watchdog interval.\n");
88             v=0;
89         }
90     }
91     return v;
92 }
93
94 /*
95 bool CE_Watchdog::IsLastBootByWatchdog()
96 {
97     int v=0;
98     if(fd!=(-1)){
99         if (ioctl(fd, WDIOC_GETBOOTSTATUS, &v) != 0) {
100             perror("Error in reading boot status.\n");
101             v=0;
102         }
103     }
104     return (v!=0)?true:false;
105 }
106 */
107 #endif
```

ጠረኞ፡ ፩.፭: Watchdog ቁጥር 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     CE_Watchdog wd;
10    printf("Watchdog interval: %d s \n", wd.GetInterval());
11/*
12    if(wd.IsLastBootByWatchdog()){
13        printf("Last boot is by watchdog \n");
14    }
15    else {
16        printf("Last boot is by power-on-reset \n");
17    }
18 */
19    printf("Enter p to pat the watchdog \n");
20    printf("Enter q to quit \n");
21    do{
22        state = getchar();
23        if(state=='p'){
24            printf("pat... \n");
25            wd.Pat();
26        }
27    } while (state!='q');
28    printf("Closing the application\n");
29    wd.Close();
30    return 0;
31}

```

စာရင်း ၉.၉: ce_watchdog.h အသုံးပြု ပုံ နမူနာ watchdog.cpp

ပရီဂရမ် 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
```

```
debian@beaglebone:~/watchdog$ ls
bar ce_watchdog.h watchdog.cpp
debian@beaglebone:~/watchdog$ g++ watchdog.cpp -o watchdog
debian@beaglebone:~/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
debian@beaglebone:~/watchdog$
```

ပုံ ၉.၁၅: Watchdog အသုံးပြုခြင်း နမူနာ။

ပရီဂရမ် ကို run ပြီးတဲ့ အခါ သူရဲ့ကုစ် ထဲမှာ သတ်မှတ် ထားတဲ့ ၁၅ စွဲနှင့် မကုန်ခင် ပုံမှန် p ခလုတ် ကို နှိပ် enter နှိပ်ပြီး pat လုပ်ပေးရင် ပရီဂရမ် က အလုပ်လုပ် နေမှာ ဖြစ်ပြီး၊ ထွက်ချင် ရင် ဒါ ခလုတ် ကို နှိပ်ပြီး ထွက်လို့ ရပါတယ်။ အဲဒါဆို ပရီဂရမ် က watchdog ကို ရပ်ပြီး အဆုံးသတ် သွားမှာ ပါ (ပုံ ၉.၁၅)။ အဲလို့ ဒါ ကို နှိပ်ပြီး မထွက်ပဲ အချိန်လွန် သွားတဲ့ အထိ ပဲ နဲ့လည်း pat မလုပ်ရင် စက်က အလိုအလျောက် reboot ဖြစ်သွားတာ ကို တွေ့နှင့် ပါတယ်။

အကိုးအကားများ

- [Ada15] Adam. ssmtp to send emails. 2015. url: 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>.

- [Kun10] Kunilkuda. Howto Use Linux Watchdog. 2010. url: <https://embeddedfreak.wordpress.com/2010/08/23/howto-use-linux-watchdog/>.
- [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/>.
- [ras17b] 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

အချိန်တိကျမှုကိုထိန်းသိမ်းခြင်း

BBB က ဈေးသက်သက် သာသာ နဲ့ ထုတ်လုပ် ဖို့ ရည်ရွယ် ထားတာမူး ကွန်ပူ။ တွေမှာလို ပါဝါ ပိတ်ထား တဲ့ အချိန် တွေမှာ ပြားစွဲ ဘက်ထရီ လေးကို သုံးပြီး ဆက် အလုပ်လုပ် နိုင်တဲ့ Real Time Clock (RTC) ပါ မလာ ပါဘူး။ ဒါကြောင့် ပုံမှန် အားဖြင့် ဆိုရင် BBB က အင်တာနက် ပေါ်က time server တွေရဲ့ အချိန်ကို ယူပြီး ညီယူ အသုံးပြု ပါတယ်။

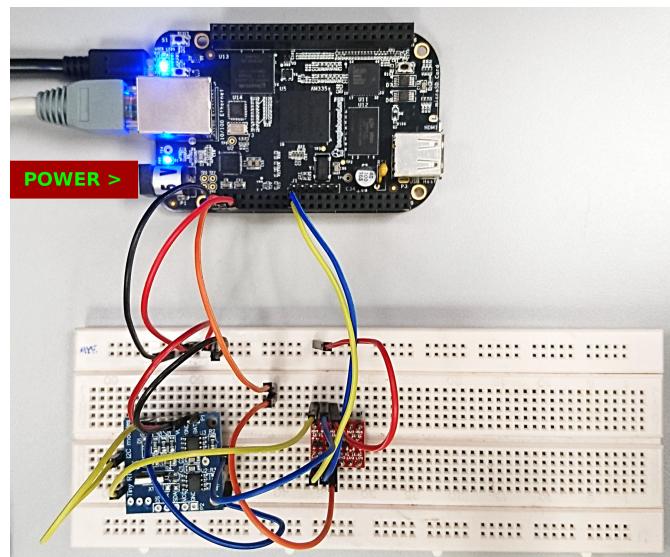
ပြဿနာ က BBB က ပါဝါ ပိတ်ပြီး ပြန်ပွင့် လာတဲ့ အခါ အင်တာနက် ချိတ်ဆက် မထား ရင် မှန်ကန်တဲ့ အချိန်ကို မသိတော့ ပဲ နောက်ဆုံး မှတ်မိတဲ့ အချိန် ကိုပဲ ဆက်သုံး ပါတယ်။ အဲဒါကို ဖြေရှင်းဖို့ ဈေးသက်သာတဲ့ RTC လေးတွေ ကို BBB နဲ့ ချိတ်ဆက် အသုံးပြု တဲ့ အကြောင်း ဈေးနေး ပါမယ်။ ပြီးတဲ့ အခါ စက်တွေ အများကြီး ကို Network Time Protocol နဲ့ နာရီ အားလုံး ပြုတဲ့ ဖြစ်အောင် ချိန်ညီ တဲ့ အကြောင်း ဆက် ဈေးနေး ပါမယ်။

၁၀.၁ Real Time Clock အသုံးပြုခြင်း

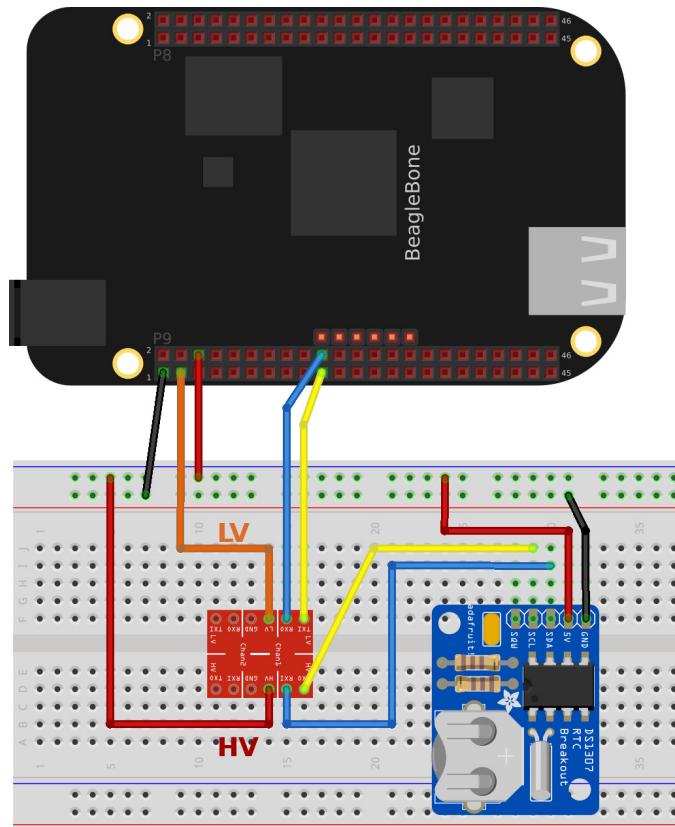
ခေတ်စား၊ အသုံးများ တဲ့ RTC တစ်ခု က DS1307 ပါ။ DS1307 RTC chip အပြင် AT24C32 ဆိုတဲ့ 32k EEPROM လေးပါ အဆင် ပါတဲ့ RTC module လေးတွေက AliExpress မှာ ငါးမူး လောက်ပဲ ပေးရ ပါတယ်။ တခြား ဈေးပေါ်တဲ့ PCF8523 တို့၊ ပိုတိကျ ကောင်းမွန်တဲ့ DS3231 Precision RTC တို့ ကို လည်း သုံးလို့ ရပါတယ်။

၁၀.၁.၁ ဝါယာဆက်သွယ်မှု

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။ ဒါ နမူနာ မှာ သုံးထားတဲ့ [Tiny RTC module](#) က 5V ဖြစ်နေ တဲ့ အတွက် BBB ရဲ့ 3.3V IO တွေနဲ့ အဆင်ပြေ အောင် pull-up resistors တွေကို ဖြတ် ချင် ဖြတ်၊ မဖြတ်ချင် ရင် [bi-directional logic level converter](#) လေးတွေ သုံးနိုင် ပါတယ်။ DS1307 ရဲ့ နမူနာ ဆက်သွယ်မှု ကို ပုံ ၁၀.၁ နဲ့ ပုံ ၁၀.၂ မှာ ပြထား ပါတယ်။ စက်ကို 5 V barrel jack ကနေ ပါဝါ ပေးဖို့ အရေးကြီး ပါတယ်။ USB ပါဝါ ကို သုံးရင် တခါတလေ ပြဿနာ ရှိနိုင် ပါတယ်။ [Tiny RTC module](#) ဆိုရင် D1, R2, R3, R4, R5, နဲ့ R6 တို့ကို ဖြတ်ပစ် ပြီး R6 နေရာ မှာ short circuit လုပ်ပေး နိုင်ရင် ကောင်းပါတယ် [[sai17](#)]။



ပုံ ၁၀.၁: DS1307 RTC ကိုဆက်သွယ်ခြင်း။



ပုံ ၁၀.၂: DS1307 RTC ကိုဆက်သွယ်ပုံ schematic diagram ။

၁၀.၂.၂ I2C ဆက်သွယ်မှု

BBB ရဲ့ I2C2 က ပုံမှန် အားဖြင့် အလုပ် တန်းလုပ် နေမှာ ဖြစ်ပြီး၊ ဝါယာ ဆက်သွယ်မှု အဆင်ပြု မဖြေစိတ်ဖို့ အတွက် terminal မှာ

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

ကို ရိုက်ကြည့် နိုင် ပါတယ်။ အဲဒီ အခါမှာ DS1307 ရဲ့ ID ဖြစ်တဲ့ 68 ကို တွေ့ရ ပါလိမ့်မယ် (ပုံ ၁၀.၃)။ Firware ဗားရှင်း အသစ် တွေမှာ i2c 1 နဲ့ i2c 2 ပြောင်းပြန် ဖြစ်နေနိုင် ပါတယ်။

```
root@beaglebone:~# 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: 50 -- -- -- UU UU UU UU - - - - - - - - - -
60:          - - - - - - - - - - - - - - - - - -
70:          - - - - - - - - - - - - - - - - - -
```

ပုံ ၁၀.၃: I2C ကို scan ဖတ်ကြည့်ခြင်း။

DS1307 ရဲ register address 0 ကနေ 7 အထိက second, minute, hour, weekday, day, month, year အသီသီး ဖြစ်တာဖို့ address 0 က second တန်ဖိုး 0x00 ကနေ 0x59 ထိ ပြောင်းလဲ နေတာကို အောက်က အတိုင်း ဖတ်ကြည့် နိုင် ပါတယ်။

```
i2cget -y 1 0x68 0x00
```

```
root@beaglebone:~# i2cget -y 1 0x68 0x00
0x48
root@beaglebone:~# i2cget -y 1 0x68 0x00
0x50
root@beaglebone:~# i2cget -y 1 0x68 0x00
0x53
```

ပုံ ၁၀.၄: DS1307 ၏ second တန်ဖိုးကိုဖတ်ကြည့်ခြင်း။

၁၀.၅.၃ RTC ကို setup လုပ်ခြင်း

နောက် အဆင့် အနေနဲ့ DS1307 အတွက် driver ကို အောက်က အတိုင်း တပ်ဆင် လိုက် ပါမယ်။

```
echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
```

Root မဟုတ်ပါက အောက်ပါ အတိုင်း တပ်ဆင် နိုင်သည်။

```
sudo sh -c "echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-2/new_device"
```

ပြီးလို့ I2C bus ကို scan ပြန်ဖတ်ကြည့်လိုက် ရင် driver က အသုံးပြု ထောက်ပံ့ နေတဲ့ အတွက် 68

၁၀.၁. REAL TIME CLOCK အသုံးပြုခြင်း

၂၈၁

အစား ပူးကျော် လိုအပ်သူမှုတွေရ ပါမယ်။ RTC ရဲ့နာမည်၊ အချိန်၊ ရက်စွဲ တွေကို လည်း စာရင်း ၁၀.၁ အတိုင်း ကြည့်နိုင် ပါတယ်။

```
1 i2cdetect -y -r 1
2 cat /sys/class/rtc/rtc1/name
3 cat /sys/class/rtc/rtc1/time
4 cat /sys/class/rtc/rtc1/date
5 ls /dev/rtc*
```

စာရင်း ၁၀.၁: RTC1 ကို စစ်ကြည့်ခြင်း။

```
root@beaglebone:~# cat /sys/class/rtc/rtc1/name
ds1307
root@beaglebone:~# cat /sys/class/rtc/rtc1/time
22:27:03
root@beaglebone:~# cat /sys/class/rtc/rtc1/date
2017-12-27
```

ပုံ ၁၀.၅: RTC1 ကိုဖတ်ကြည့်ခြင်း။

နောက် တစ်ခါ DS1307 ရဲ့အချိန်ကို အောက်က အတိုင်း ဖတ်ကြည့် နိုင် ပါတယ်။

```
sudo hwclock -r -f /dev/rtc1
```

စက်ရဲ့ အချိန်ကို စာရင်း ၁၀.၂ သို့မဟုတ် စာရင်း ၁၀.၃ အတိုင်း တိုက်နိုင် ပါတယ်။ နာရီ တိုက်ပြီး တဲ့ အခါ စာရင်း ၁၀.၁ အတိုင်း ပြန်ဖတ် ကြည့်ရင် စက်ရဲ့ အချိန် ပြောင်းသွား တာကို တွေ့နိုင် ပါတယ်။

```
1 timedatectl
2 sudo timedatectl set-ntp no
3 sudo timedatectl set-time "2017-12-27 17:15:43"
4 sudo hwclock -w -f /dev/rtc1
5 date
6 timedatectl
```

စာရင်း ၁၀.၂: Firmware အသစ်များ အတွက် RTC1 ကို အချိန် တိုက်ခြင်း။

```

1 date +%Y%m%d -s "20171227"
2 date +%T -s "12:53:10"
3 sudo hwclock -w -f /dev/rtc1
4 sudo hwclock -r -f /dev/rtc1
5 date

```

စာရင်း ၁၀.၃: Firmware အပေါင်းများ အတွက် RTC1 ကို အချိန် တိုက်ခြင်း။

၁၀.၁.၄ Service ဖန်တီးခြင်း

စက်ပွင့် လာတဲ့ အချိန်မှာ rtc ကို အလို အလျောက် တပ်ဆင် ပြီး အချိန်ကို တိုက်ဖို့ service တစ်ခု ကို ဖန်တီး ပါမယ်။ အဲဒီ အတွက် script ကို အောက်က အတိုင်း ပြုလုပ် ပါမယ် [Coo15]။

```

sudo mkdir /usr/share/rtc_ds1307
sudo nano /usr/share/rtc_ds1307/clock_init.sh

```

ပြီးရင် script ဖိုင်ထဲ မှာ အောက်က အတိုင်း ဖြည့်ပြီး သိမ်းလိုက် ပါမယ်။

```

#!/bin/bash
sleep 15
echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
hwclock -s -f /dev/rtc1
hwclock -w

```

စက်ပွင့် လာတဲ့ အချိန်မှာ အလိုအလျောက် လုပ်ဆောင်ပြီး ခုနက script ကို လုပ်ပေးမယ့် service ကို အောက်က အတိုင်း ဖန်တီးလိုက် ပါမယ်။

```

sudo nano /lib/systemd/system/rtc-ds1307.service

```

ပြီးတဲ့ အခါ အောက်က အတိုင်း ဖြည့်ပြီး သိမ်းလိုက် ပါမယ်။

| |
|--------------------------------|
| [Unit] |
| Description=DS1307 RTC Service |

```
[Service]
Type=simple
WorkingDirectory=/usr/share/rtc_ds1307
ExecStart=/bin/bash clock_init.sh
SyslogIdentifier=rtc_ds1307

[Install]
WantedBy=multi-user.target
```

Service ကို enable လုပ်ဖို့အောက်က အတိုင်း command ပေးနိုင် ပါတယ်။

```
sudo systemctl enable rtc-ds1307.service
```

Service ကို အောက်ပါ အတိုင်း manually စနိုင်၊ ရပ်နှင့် ပါတယ်။

```
sudo systemctl start rtc-ds1307.service
sudo systemctl stop rtc-ds1307.service
```

ပြီးတဲ့ အခါ စက်ကို reboot လုပ်ပြီး အားလုံး အဆင်ပြေ မပြေ စစ်ကြည့် နိုင် ပါတယ်။

```
sudo shutdown -r now
```

၁၀.J NTP Server

Network Time Protocol (NTP) က ကွန်ပျူတာ စနစ် တွေ ကို နာရီ တွေ ကိုက်ညီအောင် ချိန်ညိုဖို့
တိုက်ဖို့ အသုံးပြု တဲ့ protocol တစ်ခု ဖြစ်ပါတယ် [Wik17a]။ နောင့်နေားမှု အမြိုးမျိုး ရှိနိုင်တဲ့ packet-
switched ဒေတာ network ဆက်သွယ်မှု ကို အသုံးပြု နိုင်ပါတယ်။ NTP က ကွန်ပျူတာ တွေကို Coor-
dinated Universal Time (UTC) ကနေ မိလို စတုန် အနည်းငယ် လောက်ပဲ အမှား အယွင်း ရှိအောင်
ချိန်ညို ပေးနိုင် ဖို့ ရည်ရွယ် ပါတယ်။

ဒါ protocol ကို client-server ပုံစံ နဲ့ ဖော်ပြလေ့ ရှိပေးမယ့်၊ တစ်ခု ကို တစ်ခု အချိန် ကိုးကား လိုရတဲ့
peer-to-peer ပုံစံ မျိုးနဲ့ လည်း အလွယ် တကူ သုံးလို ရပါတယ်။ User Datagram Protocol (UDP) ကို

port နံပါတ် 123 သုံးပြီး timestamp တွေ ပို့တာတို့ လက်ခံ တာတို့ ကို လုပ်ပါတယ်။ Client တွေက နားထောင် တာပဲ လုပ်တဲ့ broadcasting ဒါမူ မဟုတ် multicasting ကိုလည်းပဲ သုံးလို့ ရပါတယ်။ NTP က leap second ချိန်ညီမှု စတာ တွေအတွက် အသိပေးချက် တွေ ပို့ပေးနိုင် ပေမယ့်၊ local time zones တို့ 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 စက်ရဲ့ အချိန်ကို ကြည့်ဖို့ အခြေခံ command တစ်ခု ဖြစ်တဲ့ [Bou17].

```
date
```

ကို သုံးနိုင် ပါတယ်။ ပုံမှန် အားဖြင့် စက်တွေ က UTC time zone ဆို ပြီး ပုံ ၁၀.၆ မှာ ပြထား သလို ဖြစ် ရော်နှင့် ပါတယ်။

```
debian@beaglebone:~$ date
Thu Dec 28 02:32:07 UTC 2017
```

ပုံ ၁၀.၆: 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 တို့၊ 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 အရင်လို ntpd client ကို မသုံးပဲ timesyncd ကပဲ အချိန် ကို ပုံမှန် စစ်ပြီး နာရီ တိုက်ပေး ပါတယ်။ လက်ရှိ အချိန်နဲ့ ပတ်သက် တဲ့ timedatectl နဲ့ timesyncd အခြေ အနေ အချက် အလက် တွေကို အောက်ပါ အတိုင်း စစ်ကြည့် နိုင် ပါတယ်။

```
timedatectl status
```

```
debian@beaglebone:~$ timedatectl status
    Local time: Thu 2017-12-28 02:31:56 UTC
    Universal time: Thu 2017-12-28 02:31:56 UTC
          RTC time: Thu 2017-12-28 02:31:57
            Time zone: Etc/UTC (UTC, +0000)
      Network time on: yes
    NTP synchronized: yes
      RTC in local TZ: no
```

ပဲ ၁၀.၃: Checking current status of time.

အကယ်၍ NTP ရှိနေပြီး လုပ်ဆောင် နေတယ် ဆိုရင် "NTP synchronized" မှာ yes လို့ ပြပါမယ်။ Network time on: yes ဆိုတာက တော့ timesyncd က enabled ဖြစ်နေ တယ်လို့ ဆိုလို တာပါ။ အကယ်၍ timesyncd က enabled ဖြစ် မနေရင် အောက်က အတိုင်း ဖွင့် ပေးနိုင် ပါတယ်။

၂၈၆

အခန်း ၁၀. အချိန်တိကျမှုကိုထိန်းသိမ်းခြင်း

```
sudo timedatectl set-ntp on
```

၁၀.၂.၂ ntpd ပြောင်းသုံးခြင်း

အကယ်၍ timeserver လုပ်ချင်တာ ဖြစ်ဖစ်၊ ပိုတိကျ တဲ့ နာရီ တိုက်နည်း ကို အလို ရှိ တာပဲ ဖြစ်ဖစ် ဆုံးရင် timesyncd အစား ntpd ကို ပြောင်းသုံး နိုင် ပါတယ်။ ntpd ကို မတပ်ဆင် ခင်မှာ timesyncd ကို အောက်က အတိုင်း ပိတ်နိုင် ပါတယ်။

```
sudo timedatectl set-ntp no
```

ပိတ် မပိတ် အောက်က အတိုင်း ပြန်စစ် နိုင် ပါတယ်။

```
timedatectl
```

Network time on: no လို့ပြရင် timesyncd ပိတ်သွား ပြီလို သိနိုင် ပါတယ်။ အဲဒီ နောက် ntp package ကို apt-get သုံးပြီး အောက်က အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
sudo apt-get install ntp
```

တပ်ဆင် ပြီးသွားရင် ntpd က အလို အလျောက် စတင် ပါတယ်။ သူရဲ့ လုပ်ဆောင်မှု အဆင်ပြီ မပြော status ကို အောက် က command နဲ့ စစ်နိုင် ပါတယ်။

```
sudo ntpq -p
```

```
debian@beaglebone:~$ sudo ntpq -p
      remote          refid      st t when poll reach   delay    offset  jitter
===== 
 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
+210.23.18.200 .PPS.          1 u    19  64    3   8.371   -6.973  1.079
*ntpmon.dcs1.biz .PPS.        1 u    20  64    3   7.618   -6.809  1.028
+ntp.sg.eria.one 10.84.87.146 2 u    76  64    2   2.601   -6.029  1.709
#52.187.42.158 216.239.35.12 2 u    18  64    3   5.132   -12.004 0.852
-128.199.224.229 219.216.128.25 3 u    19  64    3   3.241   -1.509  1.213
#dyn107-b57-acce 133.100.11.8   3 u    17  64    3   2.545   4.684  1.095
-ntp01.cosmicflu 27.114.150.10  3 u    20  64    3   3.251   -2.738  1.209
-47.88.159.10 79.78.134.71   2 u    18  64    3   3.291   -2.552  1.269
-188.166.176.19 118.189.177.157 2 u    19  64    3   2.975   -5.492  1.062
#a.sin.pobot.net 71.80.83.115  2 u    15  64    3   2.732   -2.657  1.371
-makaki.miuku.ne 218.186.3.36  2 u    15  64    3   2.826   -6.519  1.481
-pontoon.latt.ne 44.24.199.34  3 u    19  64    3  34.967  -10.450 6.172
-188.166.215.214 122.193.230.220 3 u    16  64    3   3.006   -11.111 1.478
debian@beaglebone:~$
```

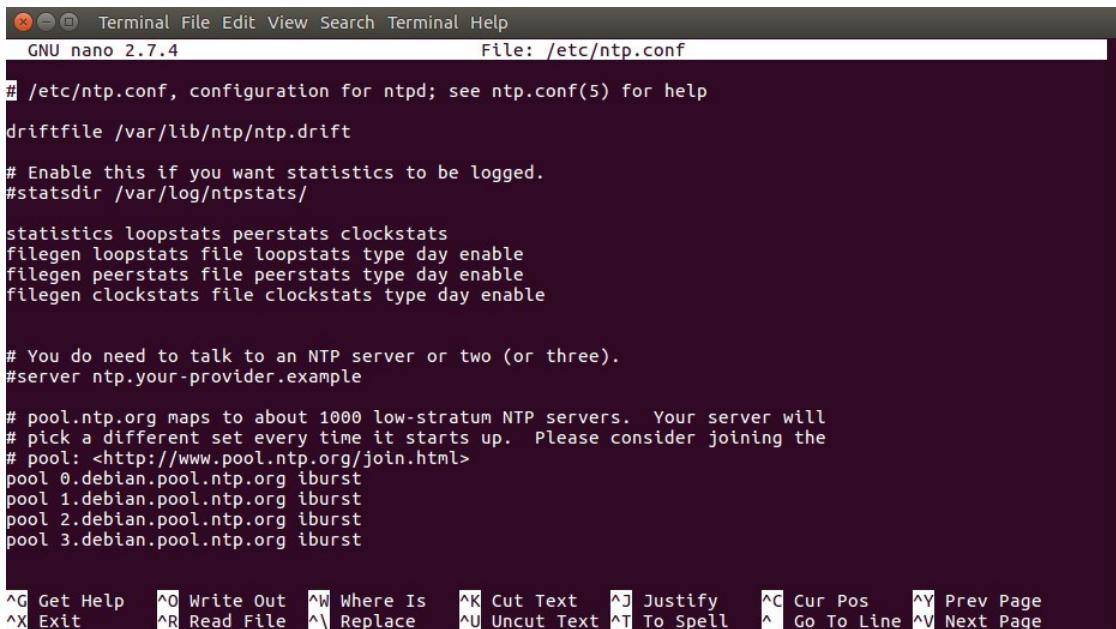
ဦး ၁၀.၈: NTP status information.

ntpq ဆိတာ က ntpd အတွက် query လုပ်ပေးတဲ့ ကိရိယာ တစ်ခု ဖြစ်ပါတယ်။ အဲဒီမှာ -p ဆိတဲ့ flag က ntpd ဆက်သွယ် နေတဲ့ NTP servers ဒါမှ မဟုတ် peer တွေကို ဆိုလို ပါတယ်။ စက် တခုနဲ့ တစ်ခု အနည်းငယ် ကွားခြား နိုင် ပါတယ်။ ပုံမှန် အားဖြင့် pool server တရာ့နဲ့ တခြား server တရာ့ ပါပါ လိမ့်မယ်။ မိနစ် အနည်းငယ် ထိ ကြောနိုင် တာကို သတိပြုဖို့ လို ပါတယ်။

၁၀.၂.၃ ntpd ကို ပုံစံသွင်းခြင်း

ntpd လက် ပုံစံ အနေအထား ကို သတ်မှတ်တဲ့ configuration ဖိုင်ကို /etc/ntp.conf မှာ တွေ့နှင့် ပါတယ် [mbs08]။ သူကို ပြင်ဖို့ အောက်က command ကို သုံးနိုင် ပါတယ်။

```
sudo nano /etc/ntp.conf
```



```

Terminal File Edit View Search Terminal Help
GNU nano 2.7.4                               File: /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

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

```

ပုံ ၁၀.၉: 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 ၁၀.၄: နှမူနာ time server စာရင်း

Server ရဲ နောက်မှာ 'iburst' ကို ထည့်ပေး ထားရင် ntpd က စက်ဖွင့် တာနဲ့ အဲဒီ server ကို အမြန် တိုက်ပါ လိမ့်မယ်။

time server အနေနှင့်လုပ်ဆောင်ခြင်း

ntp က အလုပ်လုပ် နေပြီး၊ အချိန် လည်း တိုက်ထား ပြီးတာနဲ့ အဲဒီ စက်ကို တွေ့ခြား 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 ၁၀.၅: 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 တခြား 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

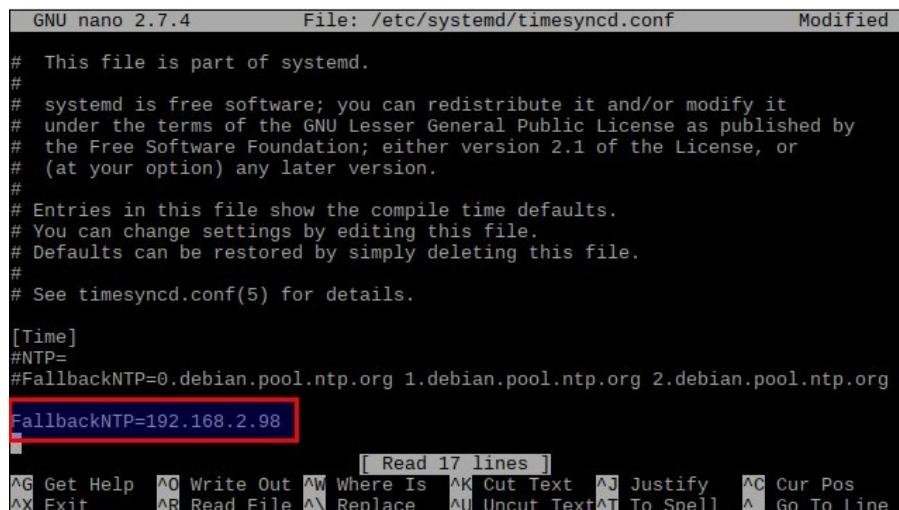
timesyncd

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
```

ဥပမာ time server ရဲ့ IP address ၏ 192.168.2.98 ဆိုရင် ပဲ ၁၀.၁၀ မှာ ပြထား သလို ပြင်နိုင် ပါတယ်။

```
FallbackNTP=192.168.2.98
```



```
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 ^\ Replace   ^U Uncut Text ^T To Spell ^L Go To Line
```

ပဲ ၁၀.၁၀: Configuring time server for timedatectl.

ပြီးတဲ့ အခါ အဲဒီ client စက်ကို restart လုပ်ပြီး၊ သူရဲ့ system log ကို ကြည့်လိုက် ရင် သတ်မှတ် ထားတဲ့ server နဲ့ နာရီ တိုက်လိုက် တာကို တွေ့နိုင် ပါတယ်။

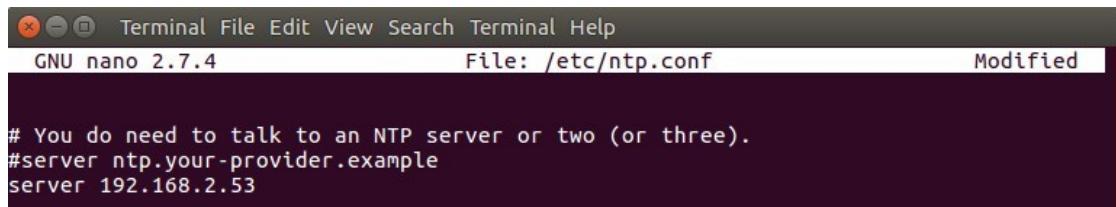
၁၀.၃. NTP CLIENT

၂၆၁

ntpd

NTP client စက်တဲ့မှာ ntpd တပ်ဆင် ထားတယ် ဆိုရင် time server ကို /etc/ntp.conf မှာ ဖွံ့ဖြိုးပေါ်လို့ အသုတေသနပေးနိုင်ပါတယ်။

```
server 192.168.2.98
```

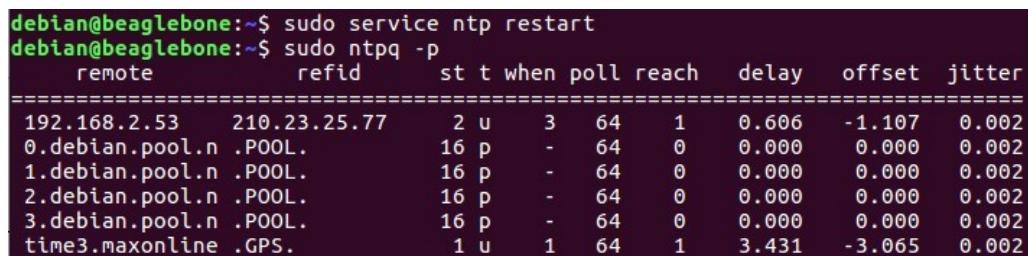


```
# You do need to talk to an NTP server or two (or three).
#server ntp.your-provider.example
server 192.168.2.53
```

ဖွံ့ဖြိုးပေါ်လို့ အသုတေသနပေးနိုင်ပါတယ်။

ဖြိုးရင် ntp service ကို ပြန်စပ်ပြီး status ကို ပွဲပါ။

```
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 |

ဖွံ့ဖြိုးပေါ်လို့ အသုတေသနပေးနိုင်ပါတယ်။

ဒါမူ မဟုတ် တကြိမ် ပဲ နာရီ တိုက်ဖို့ -q option နဲ့ အောက်ပါ အတိုင်း သုံးနိုင်ပါတယ်။

```
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 တိုက်ဖို့အောက်က အတိုင်း ရိုက်ထည့် နိုင်ပါတယ်။

```
sudo service ntp stop
sudo ntpdate -u 192.168.2.98
sudo service ntp start
```

၁၀.၃.၂ Windows

Windows စက်တွေ ကတော့ NTP ကို ရိုးရှင်း အောင် လျော့ချ ထားတဲ့ Simple Network Time Protocol (SNTP) ဆိုတာကို သုံးပြီး NTP server တွေနဲ့ နာရီ တိုက်နိုင် ပါတယ်။ အဲဒါ အတွက် အချိန် ပြနေ တဲ့ အပေါ်မှာ ကလစ် နှစ်ချက် နိုပ်ပြီး "Internet Time" ဆိုတဲ့ tab ကိုဖွင့် ပါမယ်။ ပြီးရင် Server ဆိုတဲ့ နေရာ မှာ ခုန် က လုပ်ထားတဲ့ server ရဲ့ IP address ကို ဖြည့်နိုင် ပါတယ်။

အကိုးအကားများ

- [Bou17] 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>.
- [Coo15] Justin Cooper. Adding a Real Time Clock to BeagleBone Black. 2015. url: <https://learn.adafruit.com/adding-a-real-time-clock-to-beaglebone-black>.
- [mbs08] mbsullivan. HOWTO: Set Up an NTP Server. 2008. url: <https://ubuntuforums.org/showthread.php?t=862620>.
- [sai17] saintgimp. Add a real-time clock to a Beaglebone Black. 2017. url: <https://saintgimp.org/2017/11/05/add-a-real-time-clock-to-a-beaglebone-black/>.

- [Wik17a] Wiki. Network Time Protocol. 2017. url: 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>.

၂၉၄

အခန်း ၁၀. အချိန်တိကျမှုကိုထိန်းသိမ်းခြင်း

Appendix A

Kernel Overlays

အရင် firmware ဗားရှင်း အဟောင်း တွေ မှာ kernel overlays တွေကို bone_capemgr နဲ့ သုံးတာ ဖြစ်ပြီး၊ ပြဿနာ တွေ များတဲ့ အတွက် နောက်ပိုင်း မှာ support မလုပ်တော့ ပဲ U-Boot Overlays ကိုပဲ သုံးတော့ မှာပါ။ ပြောင်းလဲ မယ့် အစီအစဉ် က အောက်ပါ အတိုင်း ဖြစ်ပါတယ် [Eli18]။

Stage 1: Disable Kernel Overlays (bone_capemgr.uboot_capemgr_enabled=1 is passed thru /proc/cmdline)

Stage 2: Disable the slots file (/sys/devices/platform/bone_capemgr/slots) (v4.4.x -> 4.14.x)

Stage 3: Disable bone_capemgr dir (/sys/devices/platform/bone_capemgr/) (v4.15.x+)

အကယ်၍ U-Boot Overlays ကို မသုံးချင်ပဲ Kernel overlay ကို ပဲ သုံးမယ် ဆိုရင် /boot/uEnv.txt မှာ "enable_uboot_overlays=1" ဆိုတာ ကို comment လုပ်နိုင် ပါတယ်။

```
#I will be on my own with Kernel Overlays:  
#enable_uboot_overlays=1
```

အရင် firmware ဗားရှင်း အဟောင်း တွေ အတွက် bone_capemgr သုံးပြီး peripheral တွေကို enable လုပ်တဲ့ အကြောင်း အောက်မှာ ဆက်လက် ဖော်ပြ ထားပါတယ်။

၁.၁ UART

UART ကို enable လုပ်ဖို့ အတွက် အောက်ပါ command တွေ သုံးနိုင် ပါတယ်။ နမူနာ အနေနဲ့ UART4 ကို သုံးထားပြီး၊ တစ္ခိုး UART တွေအတွက် လည်း သက်ဆိုင်ရာ နံပါတ် ကို အစားထိုး ပြီး enable လုပ်နိုင် ပါတယ်။

```
# echo BB-UART4 > /sys/devices/bone_capemgr.*/slots
# more /sys/devices/bone_capemgr.*/slots
# ls /dev/tty0*
```

စက်ဖွင့် လိုက်တာနဲ့ အလို အလောက် enabled ဖြစ်ချင် ရင်တော့ BeagleBone Black Rev C အတွက် ဆိုရင်တော့ /boot/u-boot/uEnv.txt ကို ပြင်ဖို့ လိုပါတယ်။ အရင် ဗားရှင်း အဟောင်း တွေဆို ရင်တော့ /media/BEAGLEBONE/uEnv.txt ပါ။ အဲဒီ ဖိုင် မရှိသေး ရင် ဖန်တီးဖို့ လိုပါတယ်။ အဲဒီ ဖိုင် ထဲမှာ capemgr.enable_partno= ဆိုပြီး enable လုပ်ချင်တဲ့ port တွေကို comma တွေကြားခံပြီး သတ်မှတ်နိုင် ပါတယ်။ ဥပမာ UART4 ကို enable လုပ်မယ် ဆိုရင် အောက်က စာရင်း အတိုင်း /boot/u-boot/uEnv.txt မှာ ဖြည့်ပြီး reboot လုပ်နိုင် ပါတယ်။

```
capemgr.enable_partno=BB-UART4
```

၁.J Disabling HDMI Cape

HDMI Cape ကို disable လုပ်ချင်ရင်တော့ /boot/u-boot/uEnv.txt ဖိုင် မှာ အောက် ပါ အတိုင်း ဖြည့်ပြီး reboot လုပ်နိုင် ပါတယ်။

```
optargs=capemgr.disable_partno=BB-BONELT-HDMI,BB-BONELT-HDMIN
```

ပြီးတဲ့ အခါ အောက်က အတိုင်း slot တွေကို ကြည့်လိုက် ရင် 'L' ပါတဲ့ cape တွေက enable ဖြစ်နေတာ ကို ဆိုလိုပြီး၊ မပါတဲ့ cape တွေက disable ဖြစ်နေတယ် လို ဆိုလိုတာ ဖြစ်ပါတယ်။

```
# more /sys/devices/bone_capemgr.*/slots
```

၁.၃ Analog Inputs

Analog အဝင် တွေကို enable လုပ်ဖို့ အတွက် စာရင်း ၁.၁ က အတိုင်း command တွေ ထည့်ပြီး Cape Manager (<http://www.elinux.org/Capemgr>) [Eli17a] ကို သုံးနိုင် ပါတယ်။

```
1 $ cd /sys/bus/iio/devices
2 $ ls
3 $ echo BB-ADC > /sys/devices/bone_capemgr.*/slots
4 $ ls
5 $ more /sys/devices/bone_capemgr.*/slots
6
```

စာရင်း ၁.၁: Analog input များကို enable လုပ်ခြင်း။

အဲဒီ အခါ iio:device0 ဆိုတဲ့ device အသစ် တစ်ခု ပေါ်လာကို တွေ့ရမှာ ဖြစ်ပါတယ်။
Overlay ကို ပြန် ဖော်ချင်ရင်တော့

```
$ more /sys/devices/bone_capemgr.*/slots
```

ဆိုတဲ့ command ကို သုံးပြီး တွေ့နိုင် တဲ့ device ရဲ့ slot နံပါတ် ကို အနုတ် လက္ခဏာ ထည့်ပြီး အောက်ပါ အတိုင်း ပြန်ဖြေတ် နိုင် ပါတယ်။ အဲဒီ လို လုပ်လိုက် တဲ့ အခါ BBB က ရပ်သွား တတ် ပါတယ်။

```
$ echo -7 > /sys/devices/bone_capemgr.*/slots
```

စက်ကို Boot လုပ်တဲ့ အခါတိုင်း overlay ကို အင့် အလျောက် တပ်ချင် ရင် uEnv.txt ဆိုတဲ့ ဖိုင် တစ်ခုကို ဖန်တီးပြီး အောက်က စာကြောင်းကို ထည့်ဖို့ လိုပါတယ်။ ပြီးရင် ကွန်ပျူတာနဲ့ ဆက်ရင် ပေါ်လာတဲ့ BeagleBone ဆိုတဲ့ drive ထဲမှာ ထည့်သွားဖို့ လိုပါတယ်။ အကယ်၍ အဲဒီဖိုင် ရှိပြီးသား ဆိုရင်တော့ အဲဒီ စာကြောင်းကို ထပ်ဖြည့်နိုင် ပါတယ်။

```
optargs=capemgr.enable_partno=BB-ADC
```

အဲလို မလုပ်ပဲ terminal ကနေပဲ အောက်က အတိုင်း edit လုပ်ရင်လည်း ရပါတယ်။

```
$ mkdir /mnt/vfat
$ mount /dev/mmcblk0p1 /mnt/vfat
$ cd /mnt/vfat
$ ls
$ nano uEnv.txt
$ cd ..
$ umount /mnt/vfat
$ reboot
$ more /sys/devices/bone_capemgr.*slots
```

၁.၅ Analog Outputs

PWM (Pulse Width Modulation) signal တွေထုတ် ဖို့ အတွက် overlay နှစ်ခု ကို Capemgr သုံးပြီး တပ်ဆင်ဖို့ လိုပါတယ်။ ဥပမာ P9_14 ကို သုံးပြီး PWM signal ထုတ်မယ် ဆိုရင် အောက်က command တွေကို သုံးနိုင် ပါတယ်။

```
# echo bone_pwm_P9_14 > /sys/devices/bone_capemgr.*slots
# echo am33xx_pwm > /sys/devices/bone_capemgr.*slots
# cd /sys/devices/ocp.3/
# ls
# cd pwm_test_P9_14.17
# echo 1000000 > period
# echo 500000 > duty
# echo 1 > run
```

နူမူနာ C++ ပရိုဂရမ် pwmtest.cpp က BBPWM ဆိုတဲ့ class (bb pwm.h နဲ့ bb pwm.cpp) ကို သုံးထားပြီး၊ သူတို့ကို အောက်က စာရင်း ၁.၂, စာရင်း ၁.၃ နဲ့ စာရင်း ၁.၄ တွေမှာ ဖော်ပြ ထားပါတယ်။

¹ `#ifndef BBPWM_H_INCLUDED`

```

2 #define BBPWM_H_INCLUDED
3
4 #include <string>
5 #include <fstream>
6 #include <sstream>
7 //http://www.cplusplus.com/reference/fstream/fstream/
8 //www.cplusplus.com/articles/D9j2Nwbp/
9 #define PWMPATH "/sys/devices/ocp.3/pwm_test_"
10 using namespace std;
11
12 class BBPWM{
13     string fpath;
14 public:
15     BBPWM(string header_pin);
16     void Period(int t_ns);
17     void Duty(int t_ns);
18     void Run(bool v);
19 };
20
21 #endif // BBPWM_H_INCLUDED

```

օօՂԸ օ.յ: bb pwm.h

```

1 #include "bb pwm.h"
2
3 BBPWM::BBPWM(string header_pin)
4 {
5     fpath = PWMPATH+header_pin+".17";
6 }
7
8 void BBPWM::Period(int t_ns)
9 {
10     ofstream wfile;
11     string path=fpath;
12     path+="/period";

```

```

13     wfile.open(path.c_str());
14     if (wfile.is_open()) {wfile << t_ns;}
15     wfile.close();
16 }
17
18
19 void BBPWM::Duty(int t_ns)
20 {
21     ofstream wfile;
22     string path=fpath;
23     path+="/duty";
24     wfile.open(path.c_str());
25     if (wfile.is_open()) {wfile << t_ns;}
26     wfile.close();
27 }
28
29 void BBPWM::Run(bool v)
30 {
31     ofstream wfile;
32     string path=fpath;
33     path+="/run";
34     wfile.open(path.c_str());
35     if (wfile.is_open()) {wfile << (v?"1":"0");}
36     wfile.close();
37 }
```

૭૦૧: કોડનું બેબ્લોક: bb pwm.cpp

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <string>
4 #include "bb pwm.h"
5 using namespace std;
6 int main()
7 {
```

```

8     BBPWM P9_14("P9_14");
9     printf("PWM signal output at P9_14\n");
10    P9_14.Period(10000);
11    P9_14.Duty(5000);
12    P9_14.Run(true);
13    int d=0, j=0;
14    for(j=0; j<10; j++){
15        for(int i=0; i<10; i++){
16            d=i*1000;
17            P9_14.Duty(d);
18            usleep(200000);
19        }
20        printf("Cycle %d.\n", j);
21    }
22    printf("%d cycles produced.\n", j);
23    return 0;
24 }
```

စာရင်း ၁.၄: pwmtest.cpp

၁.၅ I2C

I2C1 ကို enable လုပ်ဖို့ အတွက် အောက်က command ကို သုံးနိုင် ပါတယ်။

```
# echo BB-I2C1 > /sys/devices/bone_capemgr.*/slots
# more /sys/devices/bone_capemgr.*/slots
# ls /dev/i2c*
```

၁.၆ SPI

SPI0 ကို enable လုပ်ဖို့ အတွက် အောက်က command ကို သုံးနိုင် ပါတယ်။ ပြီးတဲ့ အခါ load လုပ်ပြီး တဲ့ overlay ကို ပုံ ၁.၁ အတိုင်း ထွေနိုင် ပါတယ်။

```
# echo BB-SPIODEV0 > /sys/devices/bone_capemgr.*/slots
# more /sys/devices/bone_capemgr.*/slots
# ls /dev/spi*
```

```
root@beaglebone:~/gyroi2c# more /sys/devices/bone_capemgr.*/slots
0: 54:PF---
1: 55:PF---
2: 56:PF---
3: 57:PF---
4: ff:P-0-L Bone-LT-eMMC-2G,00A0,Texas Instrument,BB-BONE-EMMC-2G
5: ff:P-0-L Bone-Black-HDMI,00A0,Texas Instrument,BB-BONELT-HDMI
7: ff:P-0-L Override Board Name,00A0,Override Manuf,BB-ADC
8: ff:P-0-L Override Board Name,00A0,Override Manuf,BB-SPIDEV0
```

ဗိုလ်ချုပ် SPI0 အတွက် overlay ကို load လုပ်ခဲ့သည်။

အကိုးအကားများ

- [Eli17a] Elinux. Capemgr. 2017. url: <http://www.elinux.org/Capemgr>.
- [Eli18] Elinux. Beagleboard:BeagleBoneBlack Debian:U-Boot Overlays. 2018. url: https://elinux.org/Beagleboard:BeagleBoneBlack_Debian_U-Boot_Overlays.

Appendix B

Code Listing

J.C 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     // Define a new frame type: this is going to be our main frame
47 class MyFrame : public wxFrame
48 {
49 public:
50     // ctor(s)
51     MyFrame(const wxString& title);
52
53     // event handlers (these functions should _not_ be virtual)
54     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     // menu items
63     Minimal_Quit = wxID_EXIT,
64
65     // it is important for the id corresponding to the "About" command to
66     // have
67     // this standard value as otherwise it won't be handled properly under
68     // Mac
69     // (where it is special and put into the "Apple" menu)
70     Minimal_About = wxID_ABOUT
71 };
72
73 // the event tables connect the wxWidgets events with the functions (event
74 // handlers) which process them. It can be also done at run-time, but for the
75 // simple menu events like this the static method is much simpler.
76 wxBEGIN_EVENT_TABLE(MyFrame, wxFrame)
77     EVT_MENU(Minimal_Quit, MyFrame::OnQuit)
78     EVT_MENU(Minimal_About, MyFrame::OnAbout)
79 wxEND_EVENT_TABLE()
80
81 // Create a new application object: this macro will allow wxWidgets to create
82 // the application object during program execution (it's better than using a
83 // static object for many reasons) and also implements the accessor function
84 // wxGetApp() which will return the reference of the right type (i.e. MyApp
85 // and
86 // not wxApp)
87 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 J.၁: Minimal wxWidgets sample, minimal.cpp

J.၂ Alpha channel ပါရိသည့် 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("C:/opencv/thiri.
26         png")), wxBITMAP_TYPE_PNG), wxPoint(256, 0), wxSize(512,512));
27
28 //From opencv to wx
29 Mat imcv1 = imread("C:/opencv/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 IMPLEMENT_APP(MyApp)
42 bool MyApp::OnInit()
43 {
44     if (!wxApp::OnInit())
45         return false;
46     wxInitAllImageHandlers();
47     MyFrame *frame = new MyFrame(wxT("Simple wxWidgets and OpenCV"));
48     frame->Show(true);
49
50     return true;
51 }
```

စာရင်း J.J: Alpha channel ပါရှိသည့် ပုံရိပ် များကို အသုံးပြုသည့် နမူနာ minimal.cpp

J.R Serial.h

```

1 //File: 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 //           -c-with.html
5 //MIT License (https://opensource.org/licenses/MIT)
6 //Copyright (c) 2017 Yan Naing Aye
7
8 // References
9 // https://en.wikibooks.org/wiki/Serial_Programming/termios
10 // http://www.silabs.com/documents/public/application-notes/an197.pdf
11 // https://msdn.microsoft.com/en-us/library/ff802693.aspx
12 // http://www.cplusplus.com/forum/unices/10491/
13
14 #ifndef SERIAL_H
15 #define SERIAL_H
16
17 #include <stdlib.h>
18 #include <stdio.h>
19 #include <string.h>
20 #include <string>
21 using namespace std;
22
23 #if defined (__WIN32__) || defined(_WIN32) || defined(WIN32) || defined(
24     __WINDOWS__) || defined(__TOS_WIN__)
25     #define CEWIN
26 #endif
27
28 #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 (!Opened()) 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_TCLEAR |
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     return 0;
272 }
273
274 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     DWORD dwRead;
328     DWORD length=1;
329     BYTE* data = (BYTE*)(&rxchar);
330     //the creation of the overlapped read operation
331     if (!fWaitingOnRead) {
332         // Issue read operation.
333         if (!ReadFile(hComm, data, length, &dwRead, &osReader)) {
334             if (GetLastError() != ERROR_IO_PENDING) { /*Error*/}
335             else { fWaitingOnRead = TRUE; /*Waiting*/}
336         }
337         else {if(dwRead==length) success = true;}//success
338     }
339 }
```

```
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 opertion 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     {
375         if (value)
376             if (EscapeCommFunction(hComm, SETRTS)) r = true;
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 |= PARENB;
472
473     if (parity == '0') 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 }
```

```
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 #endif
625
626 #endif // SERIAL_H
```


Appendix C

စကားလုံးဖွင့်ဆိုချက်များ

BBB BeagleBone Black သိမဟုတ် BeagleBone Blue

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

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 |
| 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 |
| ထိုင် | Column |
| တန်း | Row |
| ပုံရိပ်ပြုစပ်ခြင်း | Image Processing |
| ပြေားမြှောက်ထရီ | Frame |
| ပြနှံး | Frame rate |
| ပြားစွဲဘက်ထရီ | Button cell, coin battery |
| ဖန်ရှင် | Function |

ဖပ်တာ 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 လိုပဲ ခေတ်စား တဲ့ BeagleBone Black SBC ကို အသုံးပြုပြီး hardware တွေနဲ့ interface လုပ်တာ၊ OpenCV ကိုသုံးပြီး image processing လုပ်တာ စတဲ့ အကြောင်း တွေကို ဆွေးနွေး ထားပါတယ်။ Robotic system တွေ အတွက် အခြေခံ ဖြစ်တဲ့ sensor/actuator တွေကို ဆက်သွယ် အသုံးပြုတာ နဲ့ control အတွက် BeagleBoard တွေကို အသုံးပြုပဲ အခြေခံ တွေကို ခြိုင်မိ ဖို့ ရည်ရွယ်ပါတယ်။

စာရေးသူ၏ကိုယ်ရေးအကျဉ်း



ဒေါက်တာ ရန်နိုင်အေး သည် စက်ဌူးပူ နိုင်ငံ Nanyang Technological University (NTU) တွင် ၂၀၁၆ နှစ် Robotics နည်းပညာပိုင်း ဖြင့် Ph.D. ဘွဲ့ကို ရရှိ ခဲ့သည်။ NTU ရှိ Robotic Research Centreတွင် ၂၀၁၀ မှ ၂၀၁၅ ထိ ၇ နှစ်ကြာ နည်းပညာပိုင်း ဆိုင်ရာ သုတေသနများ လုပ်ကိုင် ခဲ့သည်။ ယခု အခါ Sun Singapore System Pte. Ltd. တွင် Senior R&D Engineer အဖြစ် သုတေသန လုပ်ငန်း များ ဆောင်ရွက် လျက် ရှိသည်။

အခြားရေးသားထားသောစာအုပ်များ

KiCad အားစတင်အသုံးပြုခြင်း - yan9a.github.io/KiCad/kicadmm.pdf

OpenCV ကိုလေ့လာခြင်း - yan9a.github.io/opencv/opencv.pdf

Raspberry Pi အား C++ ဖြင့်အသုံးပြုခြင်း - yan9a.github.io/rpi/rpi.pdf