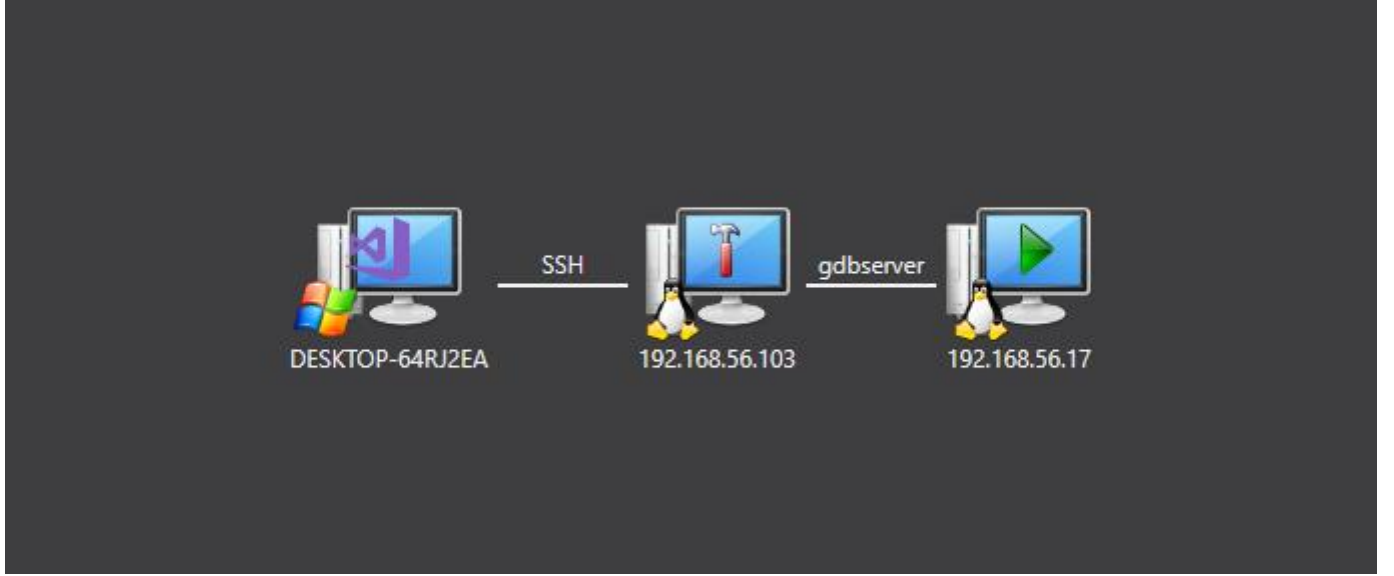
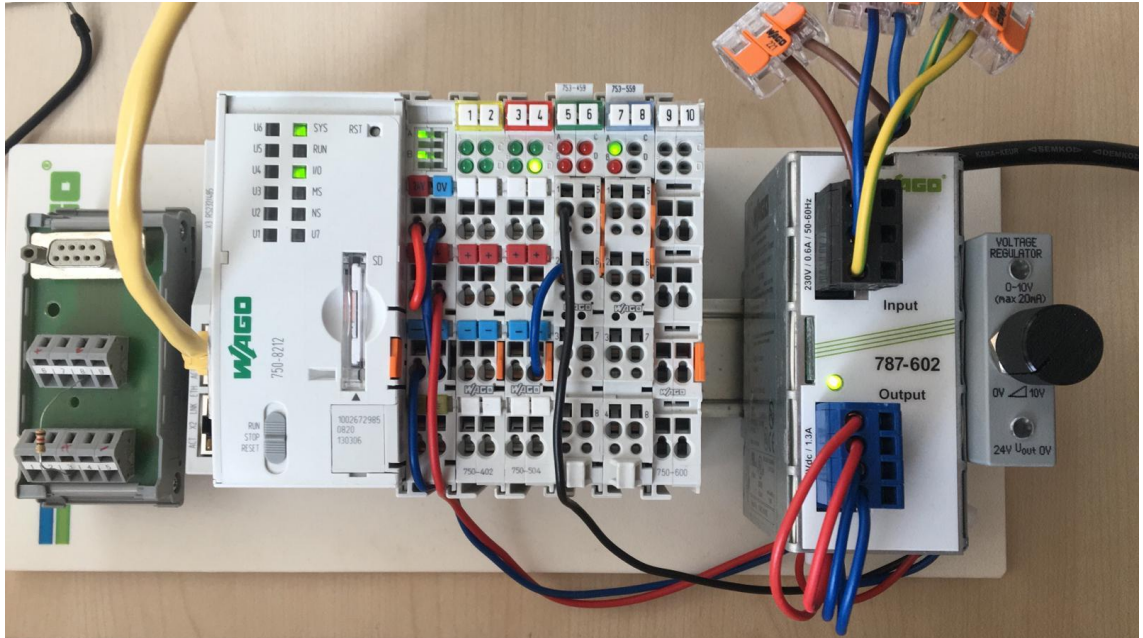


Visual GDB ile Wago Linux Geliştirme Uygulama Notu



Bu doküman aşağıdaki gibi bir demo ile hazırlanmıştır.

- WAGO 750-8212 PFC
- WAGO 750-402 Dijital Giriş Modülü
- WAGO 750-504 Dijital Çıkış Modülü
- WAGO 753-459 Analog Giriş Modülü
- WAGO 753-559 Analog Çıkış Modülü
- WAGO 750-600 Sonlandırma Modülü
- 0-10V Voltaj Regülatör
- WAGO 787-602 Güç Kaynağı



1. Geliştirme Ortamının Hazırlanması

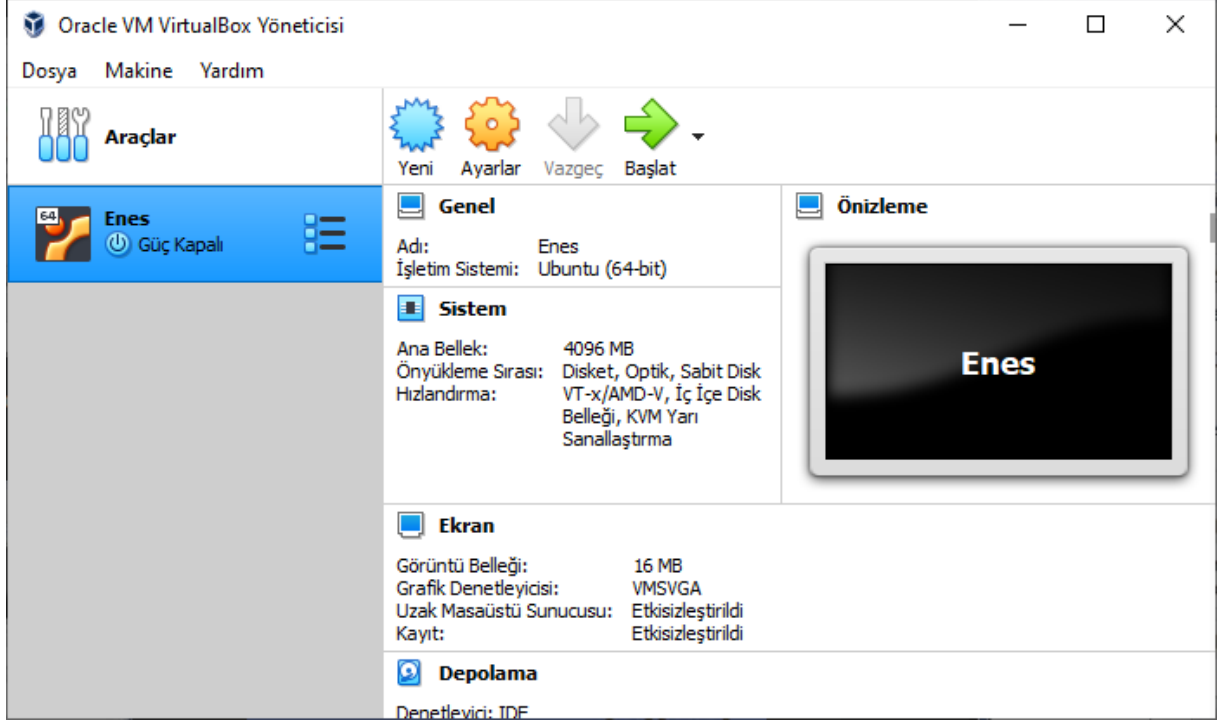
1.1 Sanal Bilgisayar Kurulumu

1.1.1 Virtual Box Kurulumu

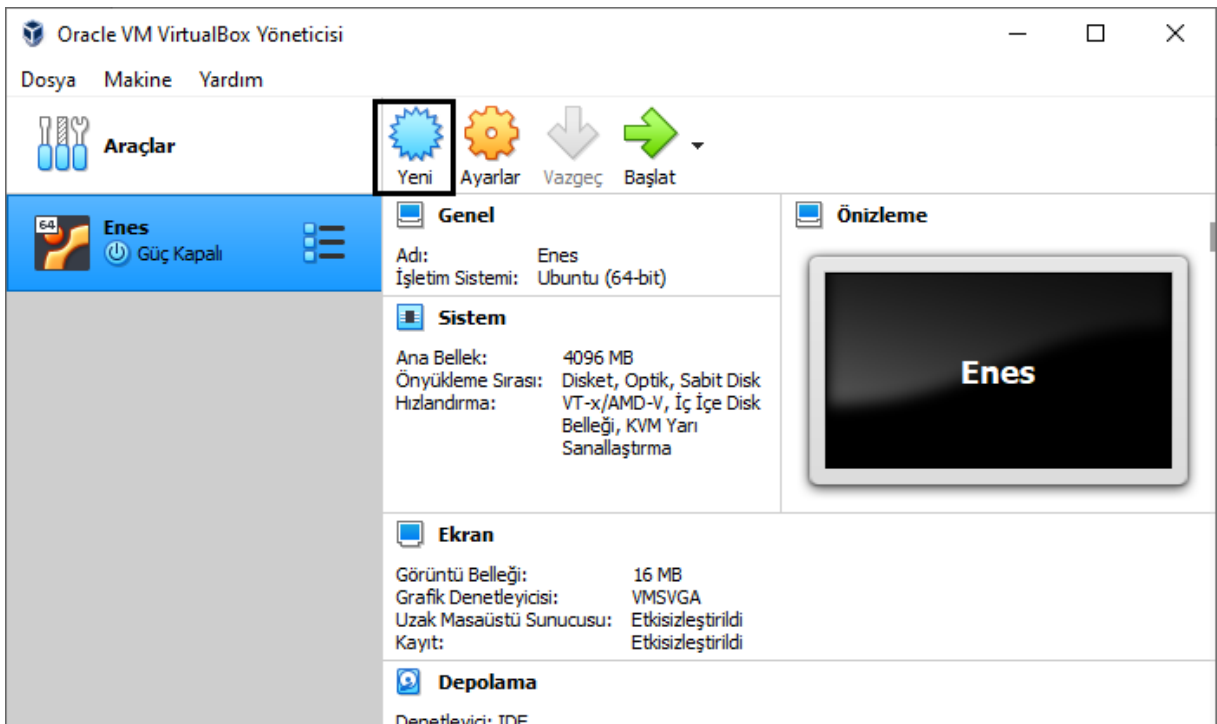
Virtual Box sanal işletim sistemi kurmanızı ve kullanmanızı sağlayan açık kaynaklı bir yazılımdır.

<https://www.virtualbox.org/wiki/Downloads> adresini kullanarak işletim sisteminize uygun olan sürümünü indirebilirsiniz. Bu dokümantasyonda Windows tabanlı bir sistem için VirtualBox 6.1.14 sürümünün 64 bit versiyonu kullanılmıştır. Standart bir kurulum yaparak kullanmaya başlayabilirsiniz.

1.1.2 Sanal İşletim Sistemi Oluşturma



Başarılı bir kurulum yaptıysanız programı başlattığınızda sizi buna benzer bir ekran karşılayacaktır. Eğer ilk kez kullanıyorsanız muhtemelen biraz daha boş bir sayfayla karşılaşacaksınız. Vakit kaybetmeden bir işletim sistemi oluşturmanız en mantıklı hareket olacaktır.



Yeni sekmesine tıklayıp bir sanal işletim sistemi oluşturmaya başlıyoruz. Makinenin kurulacağı klasörü değiştirip herhangi bir disk olarak seçebilirsiniz.

? ×


← Sanal Makine Oluştur

Adı ve işletim sistemi

Lütfen yeni sanal makine için açıklayıcı bir ad ve hedef klasör seçin ve yüklemek niyetinde olduğunuz işletim sistemi türünü seçin. Seçtiğiniz ad bu makineyi tanımlamak için VirtualBox içerisinde kullanılacaktır.

Adı:

Makine Klasörü:

Türü: 

Sürüm:

Sırada bellek miktarını seçmek var. Minimum 2048 MB kullanmanızı gereklidir. Bu dökümantasyon için 4096 MB bir bellekle çalışıldı. Bellek miktarınızı ne kadar arttırabilirseniz sanal makineniz size o kadar zaman kazandıracaktır. Yine de VirtualBox uygulamasının sıradaki şekilde gösterilen kırmızı sınırlarını aşmak bilgisayarınızı bir miktar sıkıntıya sokabilir, dikkatli olmanızı tavsiye ederim.

? ×

← Sanal Makine Oluştur

Bellek boyutu

Sanal makineye ayrılması için megabayt olarak bellek (RAM) miktarını seçin.

Önerilen bellek boyutu **1024 MB**'tır.

MB

4 MB 8192 MB

Belleğimizi de ayırdığımıza göre sıra sabit disk alanı var. Burada en az 25 GB boyutunda bir alan ihtiyacımız var. Yeterli alana sahipseniz 50 GB ayırmak daha iyi olacaktır. Yüksek boyutlu veri transferinin gerekeceği zamanları olabilir.

← Sanal Makine Oluştur

Sabit disk

Eğer isterseniz yeni makineye sanal bir sabit disk ekleyebilirsiniz. Ya yeni bir sabit sürücü dosyası oluşturabilirsiniz ya da listeden veya klasör simgesini kullanarak başka bir yerden birini seçebilirsiniz.

Eğer daha karışık depolama ayarlamasına ihtiyacınız varsa bu adımı atlayabilir ve makine bir kere oluşturulduktan sonra makine ayarlarından değişiklikleri yapabilirsiniz.

Sabit disk için önerilen boyut **32,00 GB**.

- ☐ Sanal bir sabit disk ekleme
- ☒ Şimdi sanal bir sabit disk oluştur
- ☐ Varolan sanal bir sabit disk dosyası kullan

enes.vdi (Normal, 50,00 GB)

Oluştur

İptal

← Sanal Sabit Disk Oluştur

Sabit disk dosyası türü

Lütfen yeni sanal sabit disk için kullanmak istediğiniz dosyanın türünü seçin. Eğer diğer sanallaştırma yazılımları ile kullanmaya ihtiyacınız yoksa bu ayarı değiştirmeden bırakabilirsiniz.

- ☒ VDI (VirtualBox Disk Kalıbı)
- ☐ VHD (Sanal Sabit Disk)
- ☐ VMDK (Sanal Makine Diski)

Uzman Kipi

İleri

İptal

← Sanal Sabit Disk Oluştur

Fiziksel sabit diskte depolama

Lütfen yeni sanal sabit disk dosyasının kullanılmasına göre (değişken olarak ayrılan) büyüyüp büyümemesini ya da en fazla boyutunda (sabitlenmiş boyut) oluşturulup oluşturulmamasını seçin.

Değişken olarak ayrılan sabit disk dosyası yalnızca fiziksel sabit sürücünüzdeki alanı doldurarak (en fazla **sabitlenmiş boyuta** kadar) kullanacak olmasına rağmen alan serbest kaldığında otomatik olarak tekrar küçülmeyecektir.

Sabitlenmiş boyutlu sabit disk dosyasını oluşturmak bazı sistemlerde uzun sürebilir ama kullanması çoğu kez en hızlı olandır.

- ☒ Değişken olarak ayrılan
☐ Sabitlenmiş boyut

İleri

İptal

← Sanal Sabit Disk Oluştur

Dosya yeri ve boyutu

Lütfen aşağıdaki kutuya yeni sanal sabit disk dosyasının adını yazın ya da dosyanın içinde oluşturulacağı farklı bir klasörü seçmek için klasör simgesine tıklayın.

C:\Users\enesf\VirtualBox VMs\Wago\Wago.vdi



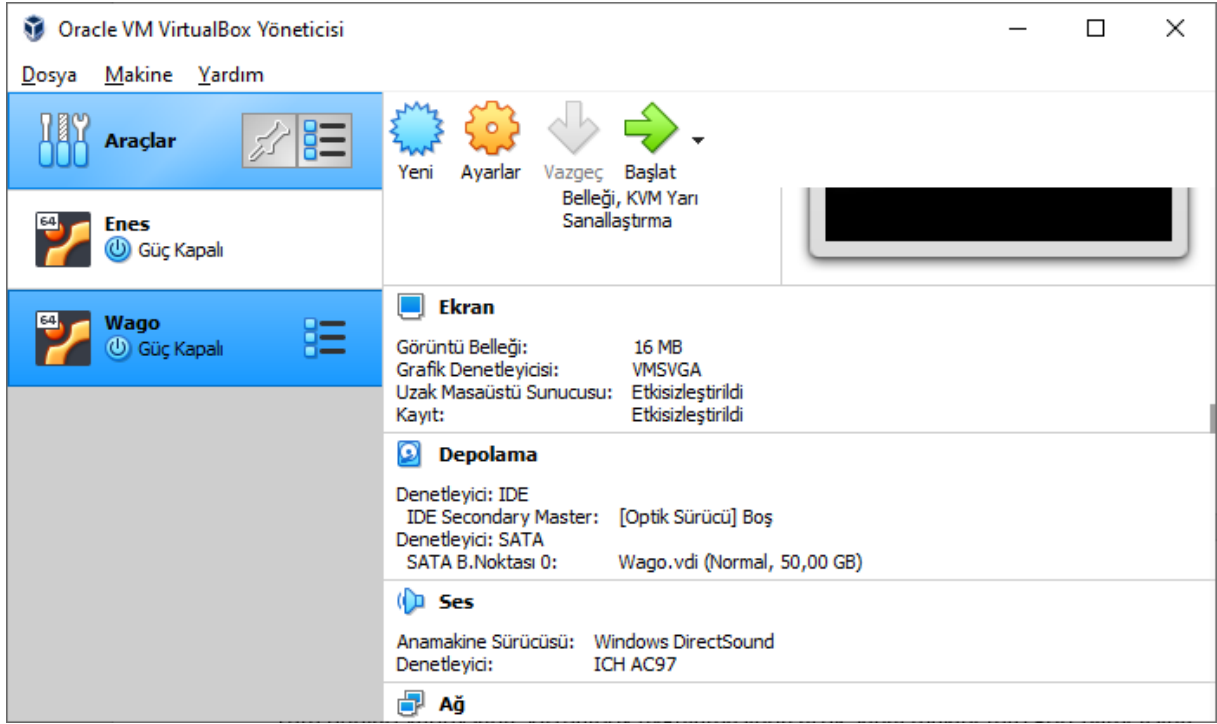
Megabayt olarak sanal sabit diskin boyutunu seçin. Bu boyut sabit disktaki depolanabilecek bir sanal makine dosya verisinin miktarını sınırlandırır.



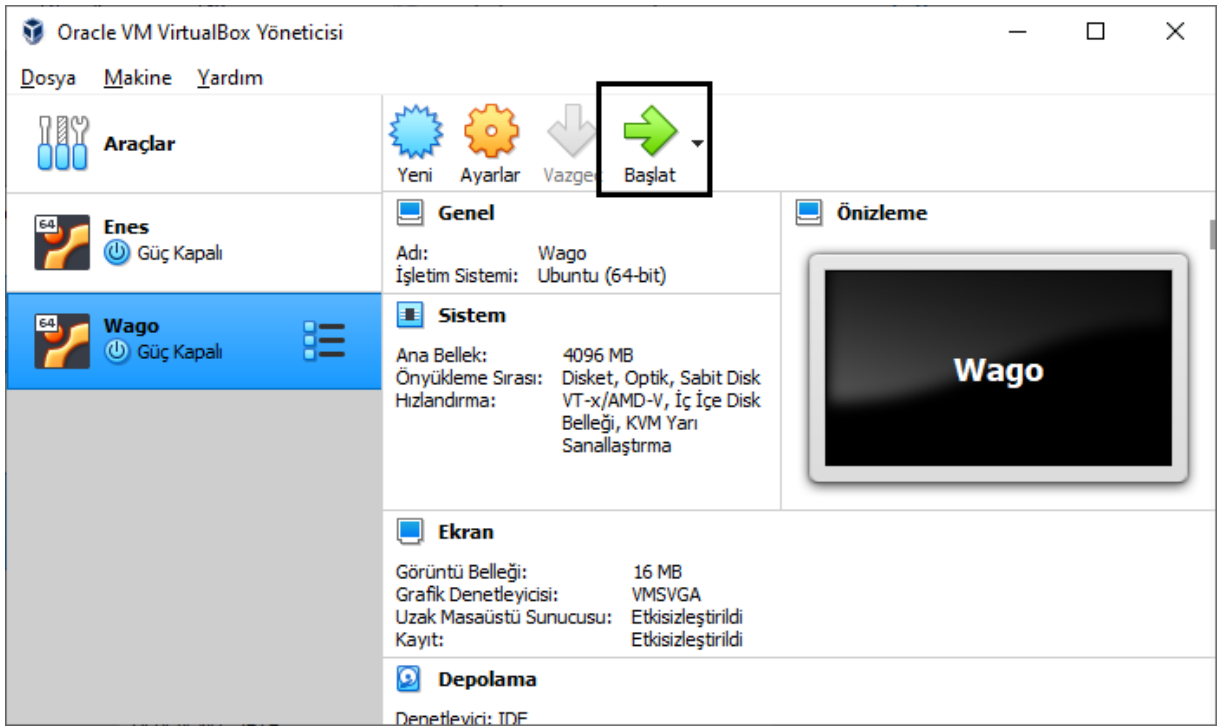
Oluştur

İptal

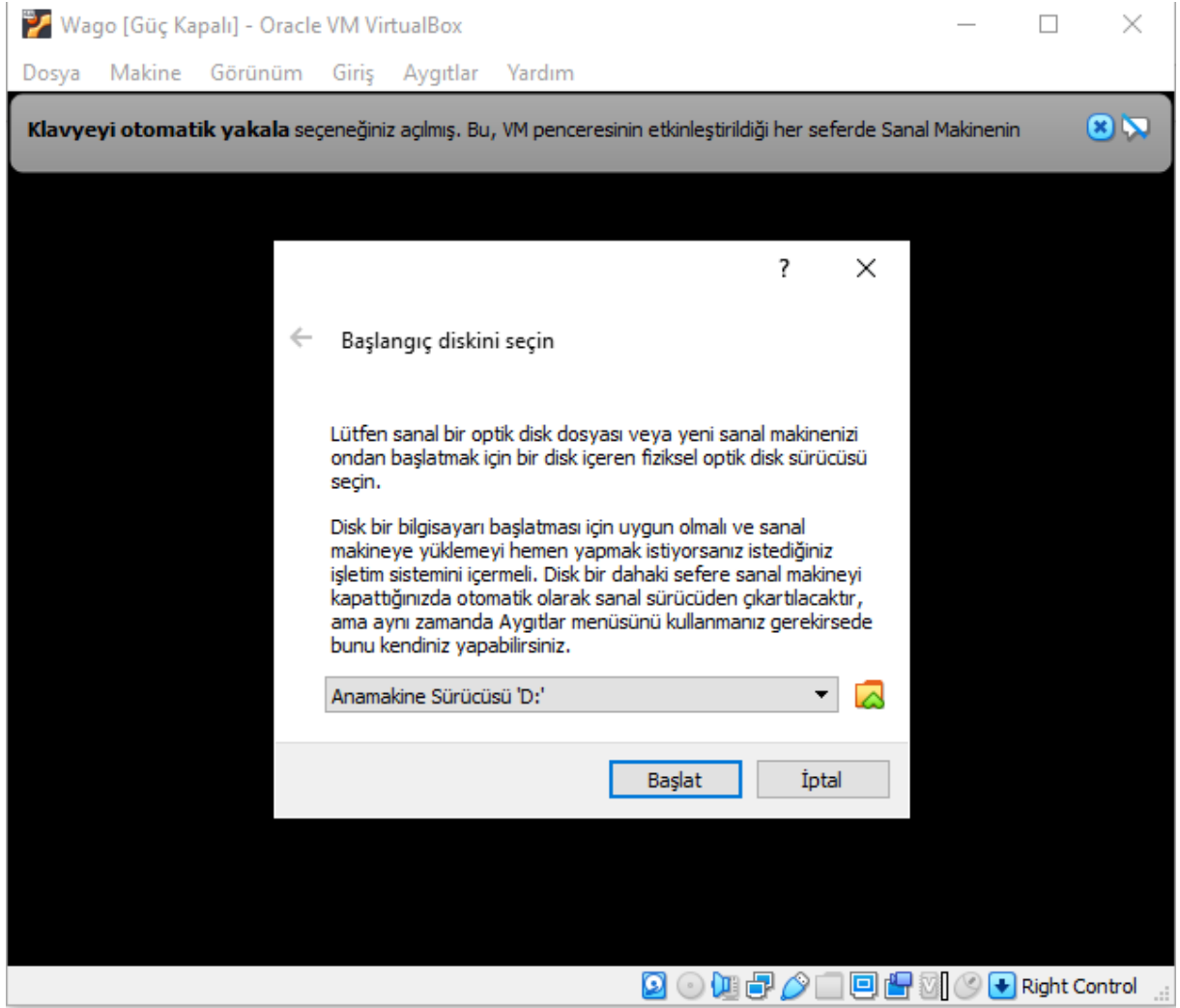
Tüm bunları yaptıysanız VirtualBox uygulamasında artık sanal makinemizi görebilmelisiniz.



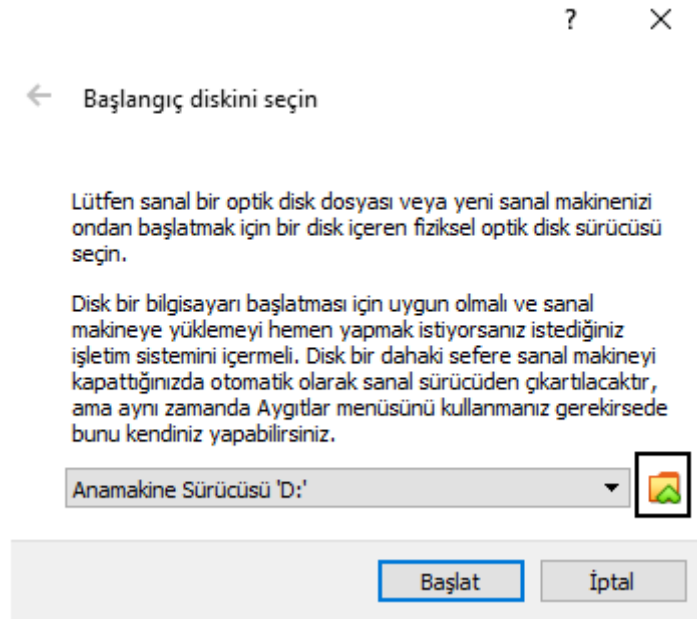
Ve artık çalıştırma zamanı.

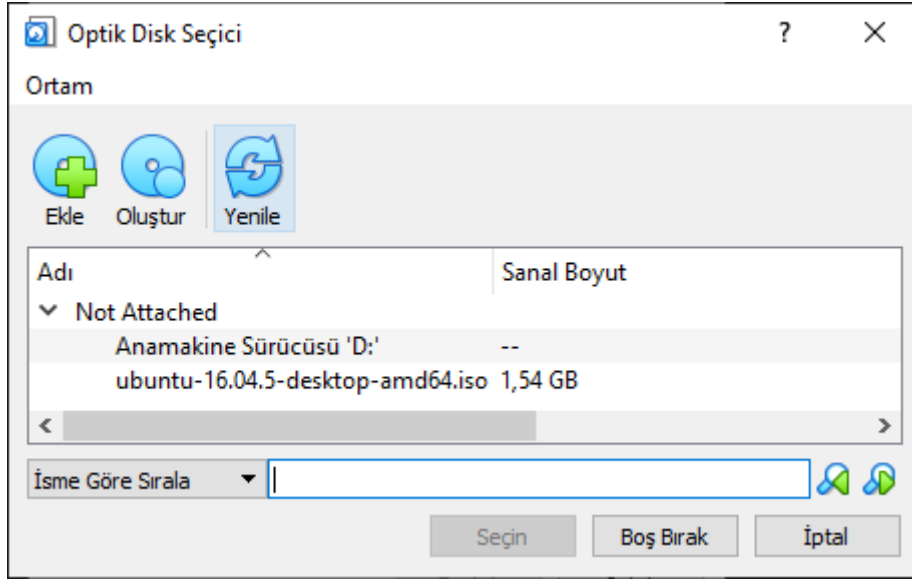


Şimdi bir sanal makinenize bir işletim sistemi kurmanız gerekmektedir. Bu dokümantasyonun hazırlanmasında Ubuntu Desktop 16.04.5 işletim sistemi kullanıldı. Kullandığınız veya aşına olduğunuz bir başka sürüm varsa kullanabilirsiniz. Büyük ihtimalle bir sıkıntı çıkmayacaktır. <http://old-releases.ubuntu.com/releases/16.04.5/ubuntu-16.04.5-desktop-amd64.iso> bu adres size kullanılan sürümü indirmenize yardımcı olacaktır. İmaj dosyasını indirdikten sonra VirtualBox uygulamasına geri dönüp yarım kalan işimizi tamamlayacağız.

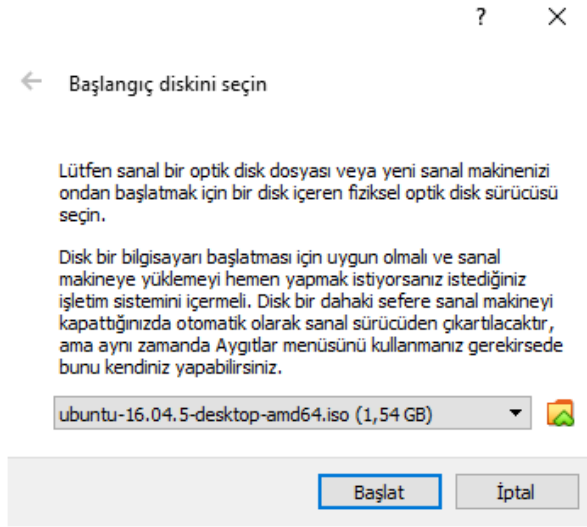


Başlat demeden önce indirdiğimiz dosyayı seçiyoruz. Açılan menüden ekle dedikten sonra indirdiğimiz .iso dosyasını buluyoruz.

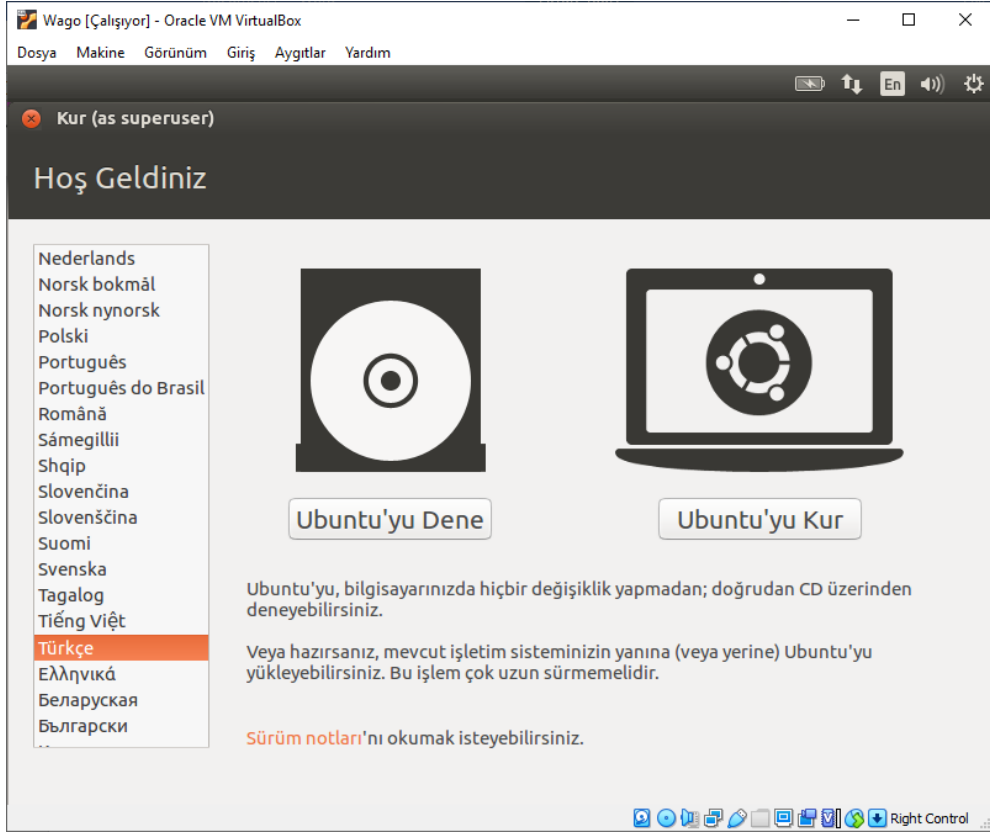




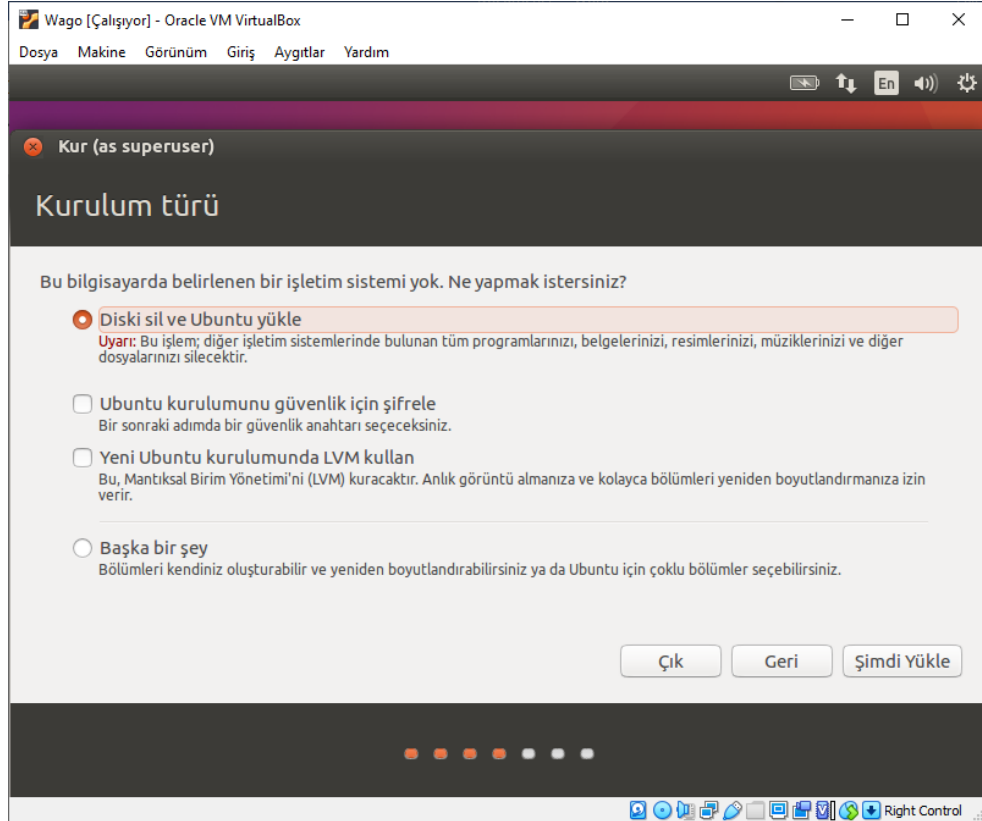
Ve başlat.



Karşınıza aşağıdaki ekran geldiyse her şey doğru gidiyor demektir. Sol menüden dil seçip Ubuntu'yu Kur diyoruz. Kurarken istediğiniz seçenekleri değiştirebilirsiniz. Eğer güncellemeleri kurulum esnasında yüklemesiniz daha hızlı bir kurulum gerçekleştirebilirsiniz.



Açılan menüde diski sil ve Ubuntu yükle seçeneğini uygulayın. Uyarılar sizi bir miktar korkutabilir, ama devam et seçeneğini seçin. Unutmayın, artık sanal bir makinedesiniz ve kendi kişisel bilgisayarınızla hiçbir bağıınız kalmadı. Yeni açılan pencerelerde saatimizi ve klavyemizi ayarlayıp devam ediyoruz.



Nihayet kendimizi tanıttığımız bölüme geliyoruz. Sanal işletim sistemimize bir isim verebilirsiniz. İstedığınız bir parolayı da ekleyip devam et diyoruz.

Wago [Çalışıyor] - Oracle VM VirtualBox

Dosya Makine Görünüm Giriş Aygıtlar Yardım

Kur (as superuser)

Kimsiniz?

Adınız: Wago ✓

Bilgisayarınızın adı: wago-VirtualBox ✓
Bu ad, diğer bilgisayarlarla kurulan iletişim esnasında kullanılır.

Bir kullanıcı adı seçin: wago ✓

Bir parola seçin: Kısa parola

Parolanızı doğrulayın: ✓

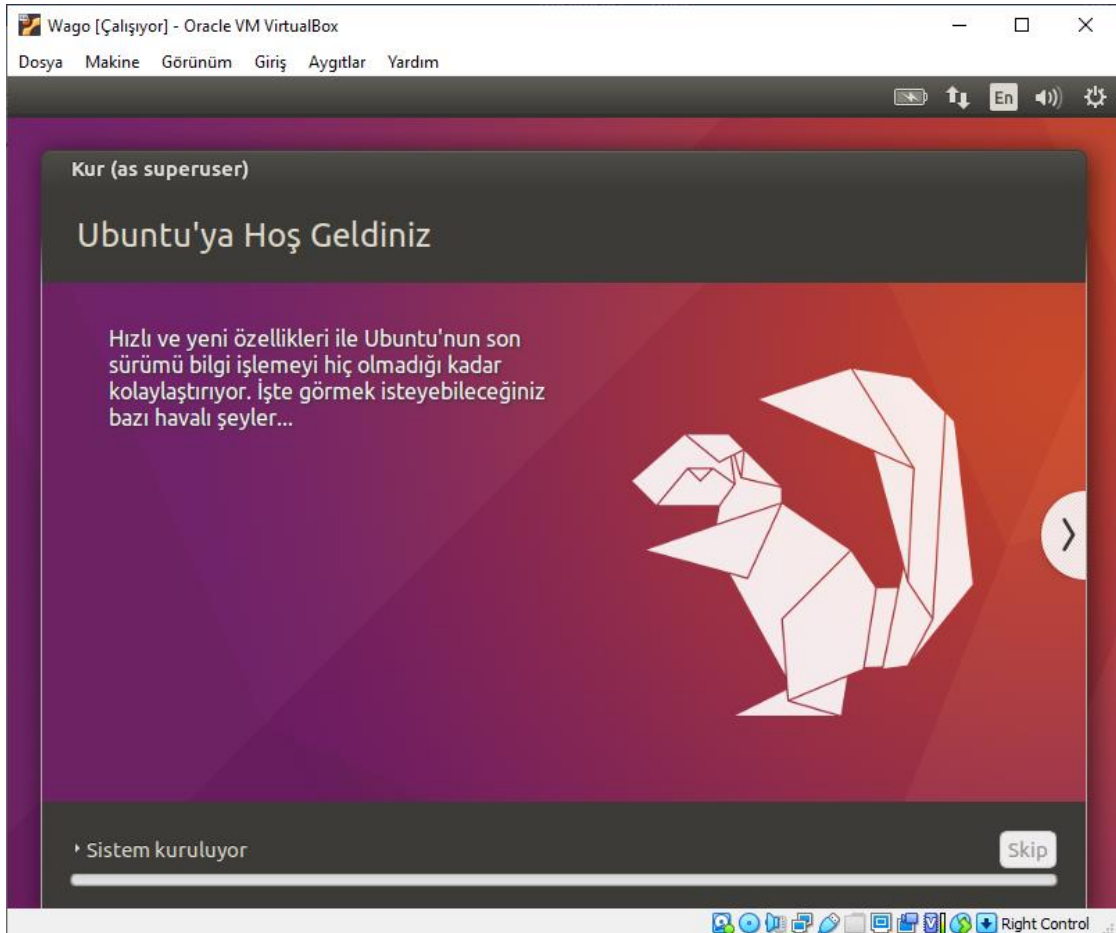
☐ Otomatik giriş yap

☒ Giriş yapmak için parola iste

☐ Ev dizinimi şifrele

Geri Devam Et

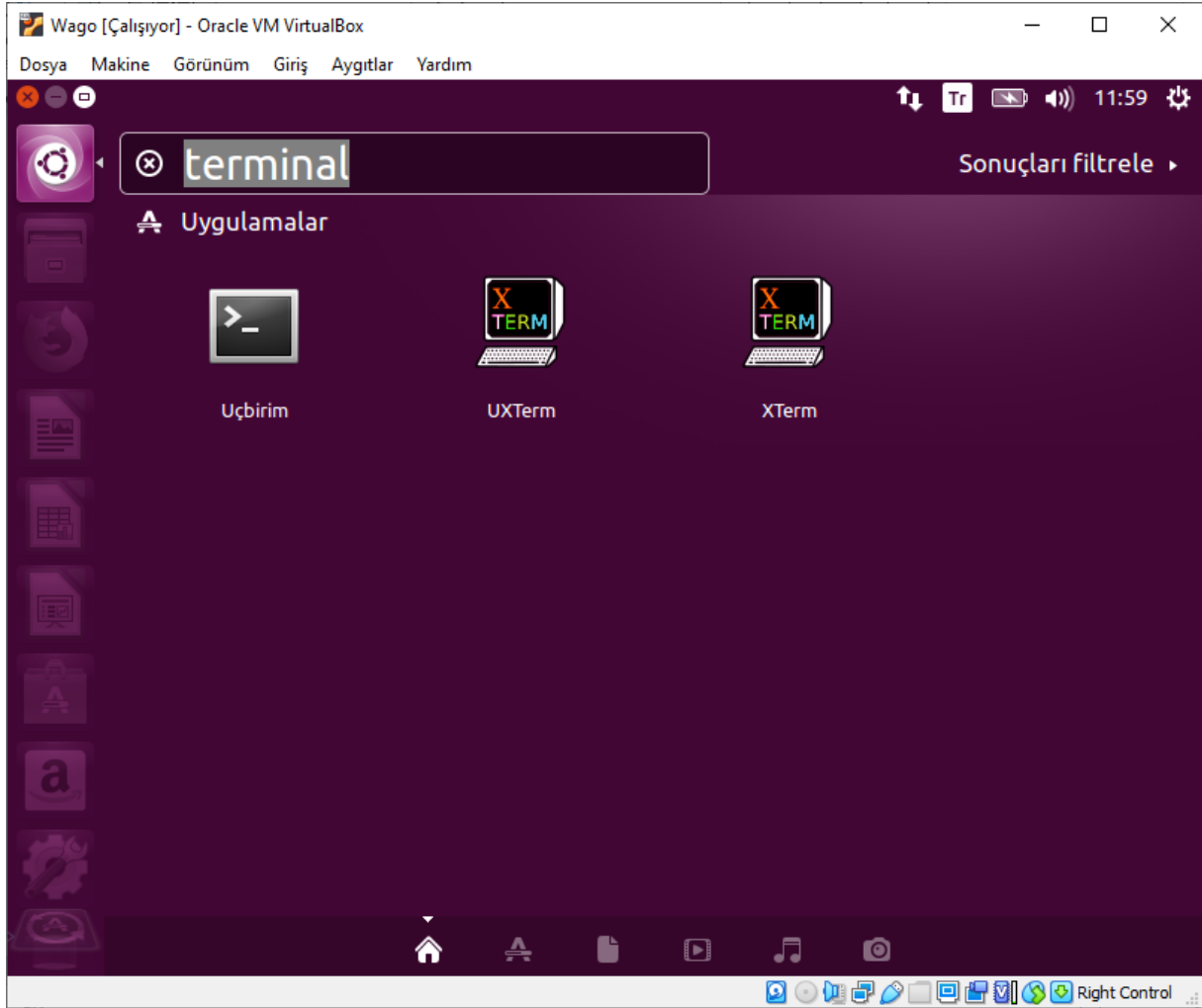
Biraz bekledikten sonra Ubuntu işletim sistemine sahip sanal makinemize kavuşmuş olacağız.

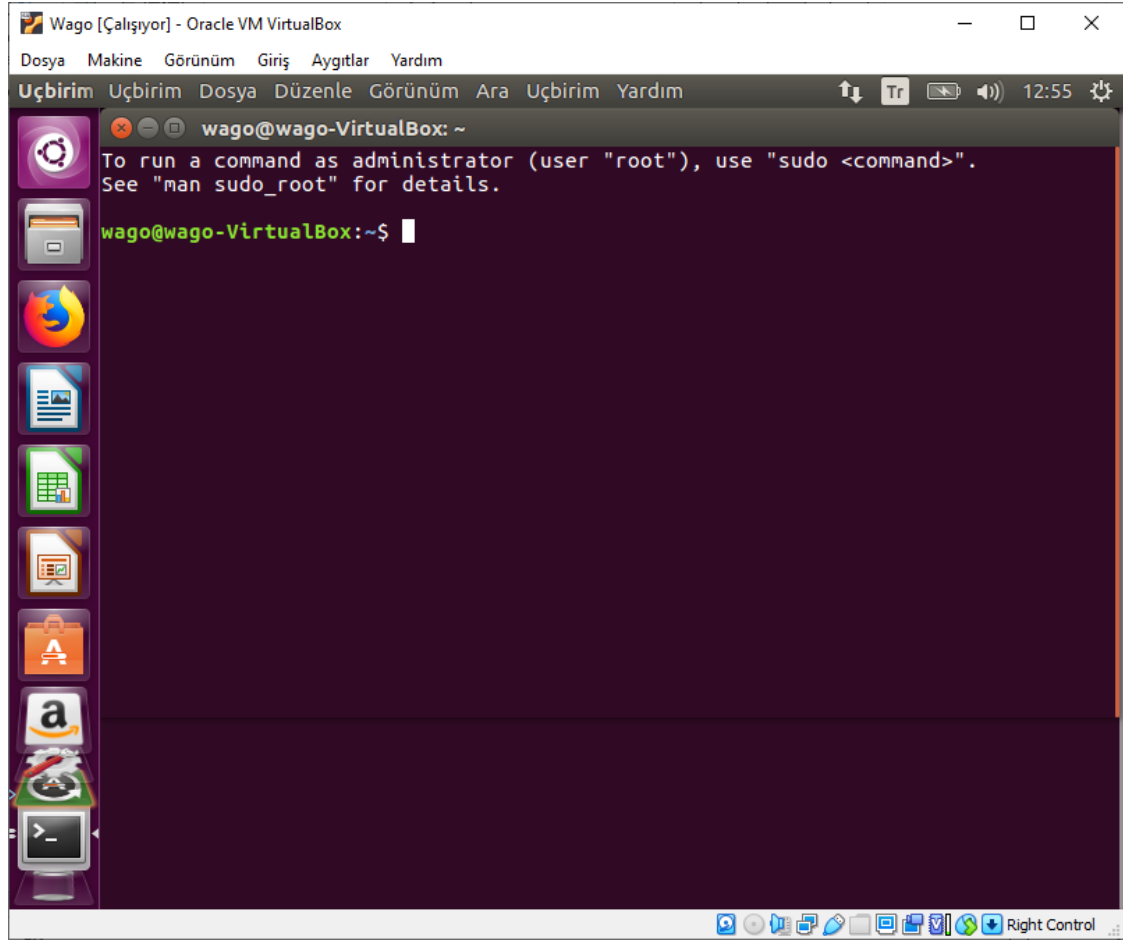


Kurulum bittikten sonra sistemimizi yeniden başlatıyoruz. Yeniden başlatmadan sonra ekranda birkaç “Error” yazısı görebilirsiniz. Endişelenecek bir şey yok biraz bekleyince giriş ekranı gelecektir. Eğer gelmezse sistemi kapatıp tekrar açabilirsiniz. Artık sanal makinemiz hazır.

1.2 WAGO-PFC-SDK Kurulumu

Bundan sonraki tüm komutlarımızı az önce kurduğumuz sanal işletim sistemimizin terminalinde gerçekleştireceğiz. Bu aşama en önemli kısım diyebiliriz. Bu adımları dikkatli izlemeniz gerekmektedir, aksi takdirde temiz bir kurulum elde edemeyebilirsiniz. Bu kurulum daha sonraki tüm geliştirmelerinizde kullanacağınız dosyayı elde etmenize yarayacaktır. İlk olarak terminalimizi açıyoruz, uçbirim adıyla karşınıza çıkacaktır.





1.2.1 Başlamadan Önce

Bu kısım <https://github.com/WAGO/pfc-firmware-sdk> adresinden alınmıştır. Daha detaylı bilgi için ziyaret edebilirsiniz.

1.2.1.1 GIT Yüklemesi

Terminalimize aşağıdaki komutu kopyalayıp yapıştırıyoruz. Eğer bu doküman bilgisayarınızın kendi sisteminde açıksa kopyayı terminale yapıştırmanız zor olabilir. Hatırlayın, artık iki farklı bilgisayara sahipsiniz. Dokümanı Ubuntu sisteminizde açarsanız işler sizin için son derece kolaylaşacaktır. Eğer hazırsanız komutları çalıştırmaya devam edelim. Bu komut sayesinde GIT uygulamasını sistemimize kuracağız. Komutları kopyalarken baştaki sembolleri seçmemeniz gerekmektedir.

```
>sudo apt install git
```

1.2.1.2 GIT Large File Support Yüklemesi

Bu adımlar önerilen sistem için hazırlanmıştır. Daha detaylı bilgi için <https://github.com/git-lfs/git-lfs/wiki/Installation> adresini ziyaret edebilirsiniz.

```
>sudo apt install software-properties-common curl
>sudo add-apt-repository ppa:git-core/ppa
>curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
Detected operating system as Ubuntu/xenial.
Checking for curl...
Detected curl...
Checking for gpg...
Detected gpg...
Running apt update... done.
Installing apt-transport-https... done.
Installing /etc/apt/sources.list.d/github_git-lfs.list...done.
Importing packagecloud gpg key... done.
Running apt update... done.

The repository is setup! You can now install packages.
>sudo apt install git-lfs
>git lfs install
```

Artık repository yani depoyu bilgisayarımıza klonlayabiliriz.

```
>cd ~
>mkdir -p wago/ptxproj/
>cd wago/ptxproj/
>git clone https://github.com/WAGO/pfc-firmware-sdk.git .
```

Sondaki nokta komutun önemli bir parçası, terminale yazarken unutmamaya çalışmalısınız.

1.2.2 “Linaro Cross Tool Chain” Yüklemesi

1.2.2.1 Git ile Linaro Alet Zincirinin Klonlanması

```
>sudo mkdir -p /opt/LINARO.Toolchain-2017.10/
>sudo git clone http://www.github.com/wago/gcc-linaro.toolchain-2017-precompiled.git
/opt/LINARO.Toolchain-2017.10/
```

1.2.3 “ptxdist” Kurulumu

1.2.3.1 Gerekli Paketlerin Kurulumu

Ubuntu 16.04 için bazı paketlere ihtiyacımız var. Bunları kurmak için aşağıdaki komutları tek tek çalıştırıyoruz. Eğer gözünüze fazla geldiyse bir alt komuta bakmalısınız.

```
>sudo apt install libncurses5-dev
>sudo apt install gawk
>sudo apt install flex
>sudo apt install bison
>sudo apt install texinfo
>sudo apt install python-dev
>sudo apt install g++
>sudo apt install dialog
>sudo apt install lzop      #used to build kernel image, ./configure did not check if
installed
>sudo apt install autoconf  #used to build kernel image, ./configure did not check if
installed
>sudo apt install libtool   #used to build kernel image, ./configure did not check if
installed
>sudo apt install xmlstarlet #used to build led_server package, ./configure did not check if
installed
>sudo apt install xsltproc  #used to build led_server package, ./configure did not check if
installed
>sudo apt install doxygen   #used to build modular-config-tools package, ./configure did not
check if installed
>sudo apt install autopoint #used to build libmodbus_tglx package
```

İsterseniz tek taşla tüm kuşları vurabilirsiniz. Alttaki komutun üstteki ile aynı etkiye sahip olduğuna emin olabilirsiniz.

```
>sudo apt install libncurses5-dev gawk flex bison texinfo python-dev g++ dialog lzop autoconf
libtool xmlstarlet xsltproc doxygen autopoint
```

1.2.3.2 Build Tool Kurulumu

```
>git clone http://github.com/wago/ptxdist.git ~/ptxdist
```

1.2.3.3 Build Tool’unun Konfigüre Edilmesi

```
>cd ~/ptxdist
>./configure
```

Bu komuttan sonra çıktınız böyle gözüküyorsa işler harika demektir:

```
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking for ptxdist patches... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
```

....

```
checking for gunzip... /bin/gunzip
checking for mktemp... /bin/mktemp
checking for wget... /usr/bin/wget
checking find version... 4.4.2
checking for gmake... no
checking for gnumake... no
checking for make... /usr/bin/make
checking for file... /usr/bin/file
checking for msgfmt... /usr/bin/msgfmt
checking for gcc... /usr/bin/gcc
checking for python2.7... /usr/bin/python2.7
checking whether /usr/bin/python2.7 finds distutils... yes
checking whether python development files are present... yes
checking for patch... /usr/bin/patch
checking whether /usr/bin/patch will work... yes
```

```
configure: creating ./config.status
config.status: creating Makefile
```

```
ptxdist version 2017.11.1 configured.
Using '/usr/local' for installation prefix.
```

Report bugs to ptxdist@pengutronix.de

Eğer değilse önceki komutları gözden geçirmelisiniz.

1.2.3.4 Build Ortamının Build Edilmesi

```
>cd ~/ptxdist
>make
```

Çıktımız böyle gözükmeli:

```
building conf and mconf ...
make[1]: Betrete Verzeichnis '/home/wago/Downloads/ptxdist-2013.03.0/scripts/kconfig'
...
gcc -g -O2 -DCURSES_LOC="<curses.h>" -DKBUILD_NO_NLS -DPACKAGE="ptxdist" -
DCONFIG_="PTXCONF" -c nconf.gui.c -o nconf.gui.o
gcc nconf.o zconf.tab.o nconf.gui.o -o nconf -lnurses -lmenu -lpanel
make[1]: Verlasse Verzeichnis '/home/wago/Downloads/ptxdist-2013.03.0/scripts/kconfig'
done.
preparing PTXdist environment ... done
```

1.2.3.5 Build Ortamının Yüklenmesi

```
>cd ~/ptxdist
>sudo make install
```

1.2.4 Proje Çevresinin Konfigüre Edilmesi

1.2.4.1 Yazılım Ayarının Seçilmesi

```
>cd ~/wago/ptxproj  
>ptxdist select configs/wago-pfcXXX/ptxconfig_generic  
info: selected ptxconfig:  
      'configs/wago-pfcXXX/ptxconfig_generic'
```

1.2.4.2 Donanım Platformunun Seçilmesi

```
>cd ~/wago/ptxproj  
>ptxdist platform configs/wago-pfcXXX/platformconfig  
info: selected platformconfig:  
      'configs/wago-pfcXXX/platformconfig'  
  
info: insufficient information in your platformconfig file  
      please use 'ptxdist toolchain </path/to/toolchain>' to select your toolchain
```

1.2.4.3 Kullanılacak Toolchain seçilmesi

```
>cd ~/wago/ptxproj  
>ptxdist toolchain /opt/LINARO.Toolchain-2017.10/arm-linux-gnueabihf/bin/  
found and using toolchain:  
'ptxdist toolchain /opt/LINARO.Toolchain-2017.10/arm-linux-gnueabihf/bin/'
```

1.2.4.4 Ana Menu

```
>cd ~/wago/ptxproj  
>ptxdist menu
```

Açıldıysa “[Exit]” ile çıkıyoruz

1.2.4.5 Menu Config

```
>cd ~/wago/ptxproj  
>ptxdist menuconfig
```

Tekrardan “[Exit]” ile çıkış yapıyoruz.

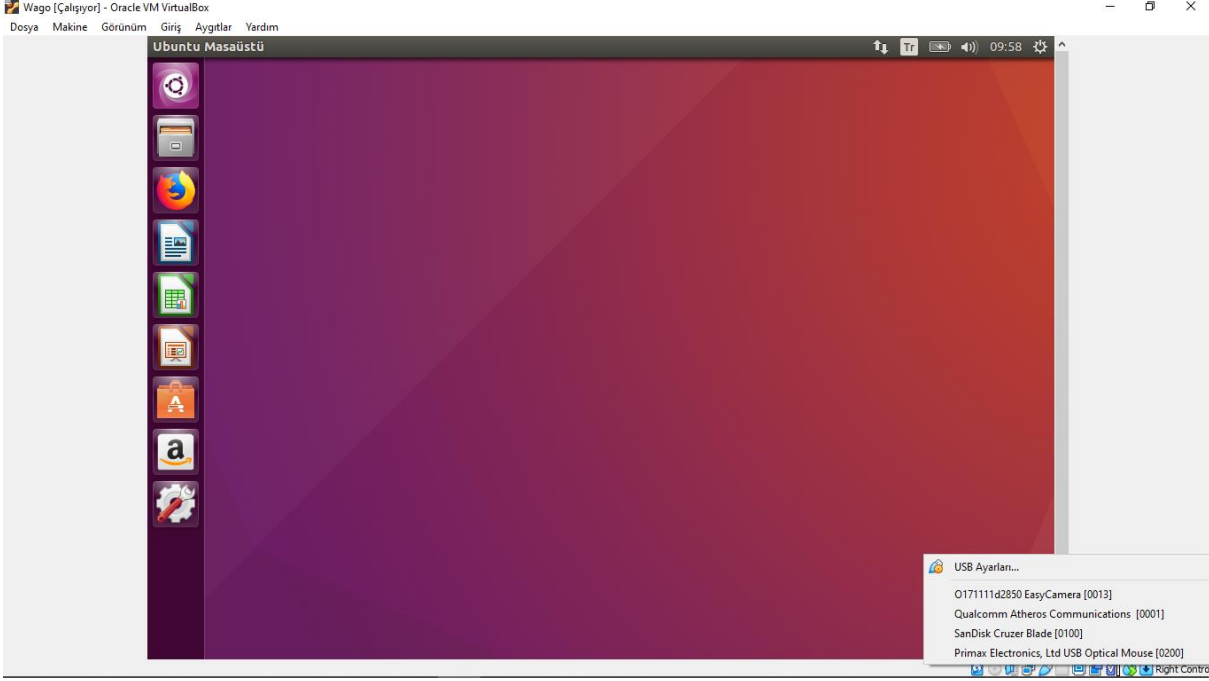
1.2.5 “sdhhd.img” Dosyasının Elde Edilmesi

En önemli kısma gelmiş bulunmaktayız. Şimdiden sizi uyarmalıyım eğer mesai saatlerinin sonundaysanız ve bilgisayarınızı kapatmanız gerekiyorsa bu komutu sakın uygulamayın. Eğer bolca vaktiniz varsa kendinize bir kahve yapıp sevdiğiniz diziyi açmaya hazırlanabilirsiniz. Bu kısım 30 dakika ile birkaç saat arasında sürebiliyor. Kahve hazırsa komutu başlatabilirsiniz.

```
>cd ~/wago/ptxproj  
>ptxdist images
```

Komut sona erdikten sonra “sd.hdimg” dosyasını ~/wago/ptxproj/platform-wago-pfcXXX/images/ klasöründe bulabilmeniz gerekiyor. Eğer dosyayı bulduysanız geliştirme ortamını oluşturmuş bulunmaktasınız. İşin büyük kısmı sizin için artık bitti sayılır.

Şimdi bir flash bellek yardımı ile image dosyamızı Windows işletim sistemine geçiriyoruz.



Sağ alttaki USB simgesine sağ tıklayıp flash belleğinizi seçerseniz artık Ubuntu işletim sistemine sahip sanal makinenizde kullanabilirsiniz.

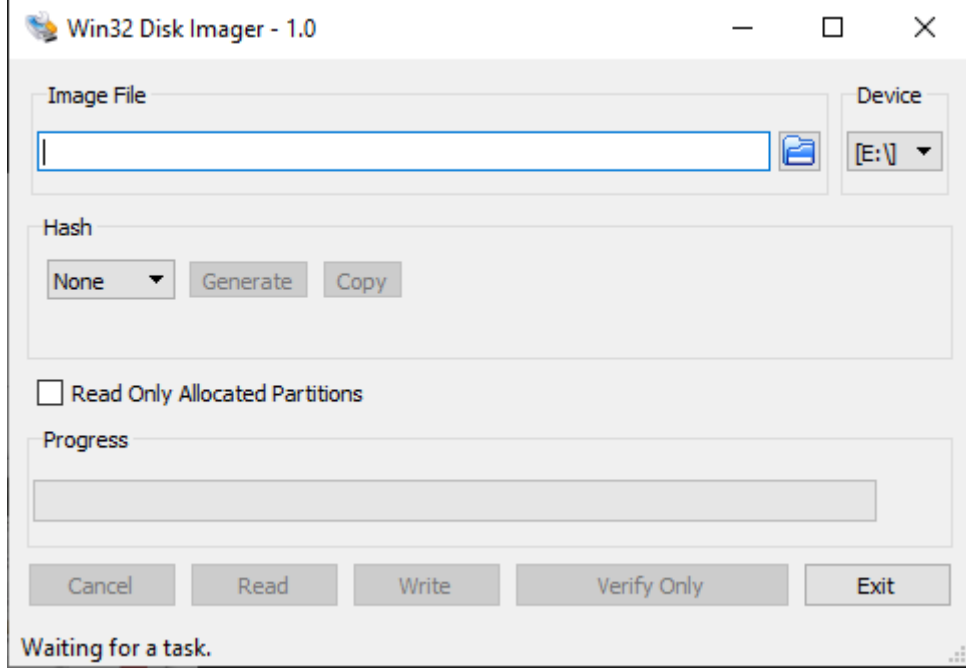
Dosya aktarımını tamamladıktan sonra flash belleği bir kez çıkarıp takarsanız tekrardan Windows işletim sisteminde kullanmaya başlayabilirsiniz. Aktardığınız dosyayı Windows sisteminize kopyalayın, şimdi ise ufak bir değişiklik yapma zamanı.



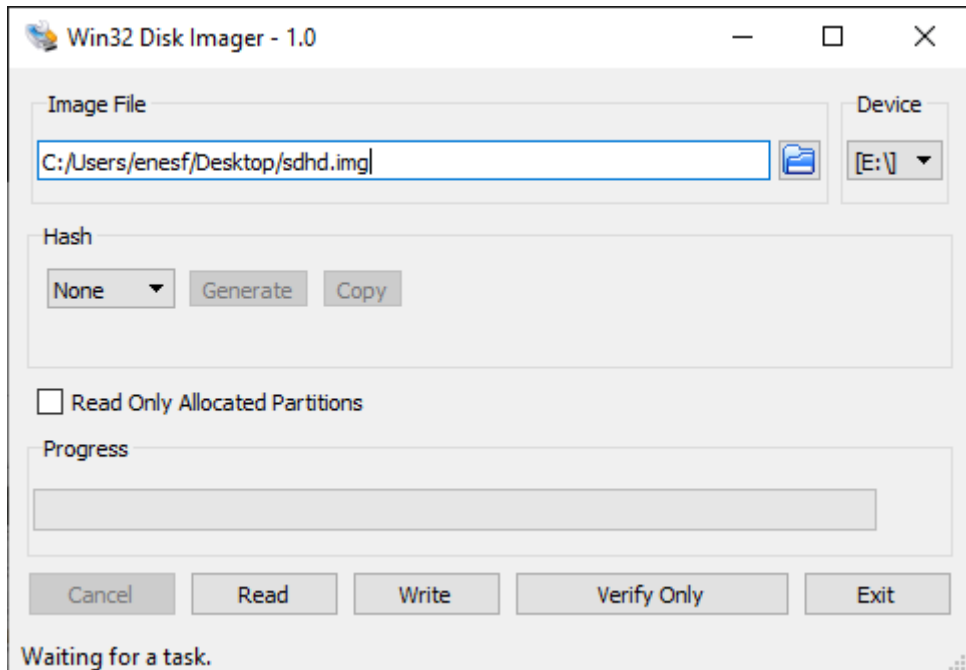
Dosyanızı yeniden adlandırıp nokta işaretini bir miktar sağa taşımanız gerekiyor. Şimdi sanal bilgisayarınızı kapatabilirsiniz.

1.2.5 “sdhd.img” Dosyasını Sd Karta Yazdırma

Bu aşamada tahmin edersiniz ki ilk olarak bir adet “SD Kart” bulmamız gerekiyor. Daha sonra bilgisayarınıza bir şekilde bağlamalısınız, çeşitli adaptörler ve dönüştürücüler kullanabilirsiniz. Diğer aşama olarak bilgisayarımıza “Win32 Disk Imager” kurulumu yapmamız gerekiyor. <https://sourceforge.net/projects/win32diskimager/> bağlantısından direkt olarak indirme yapabilirsiniz. Standart bir kurulumla “Win32 Disk Imager” programını elde edebileceksiniz. Kurulumdan sonra ise yüklediğimiz programı açıyoruz.



Klasör işaretine tıklayarak daha önce yeniden adlandırdığımız “sdhd.img” dosyasını seçiyoruz. Ardından bağladığınız SD kart hangi adı taşıyorsa (E:\, D:\ vb.) “Device” kısmından onu da seçiyoruz. Daha sonra ise “Write” sekmesini seçip bir süre bekliyoruz.



Burada işimiz bitti. Tek yapmanız gereken PFC'nizi kapatın. SD kartı takıp tekrardan açın. Artık geliştirme ortamınız hazır.

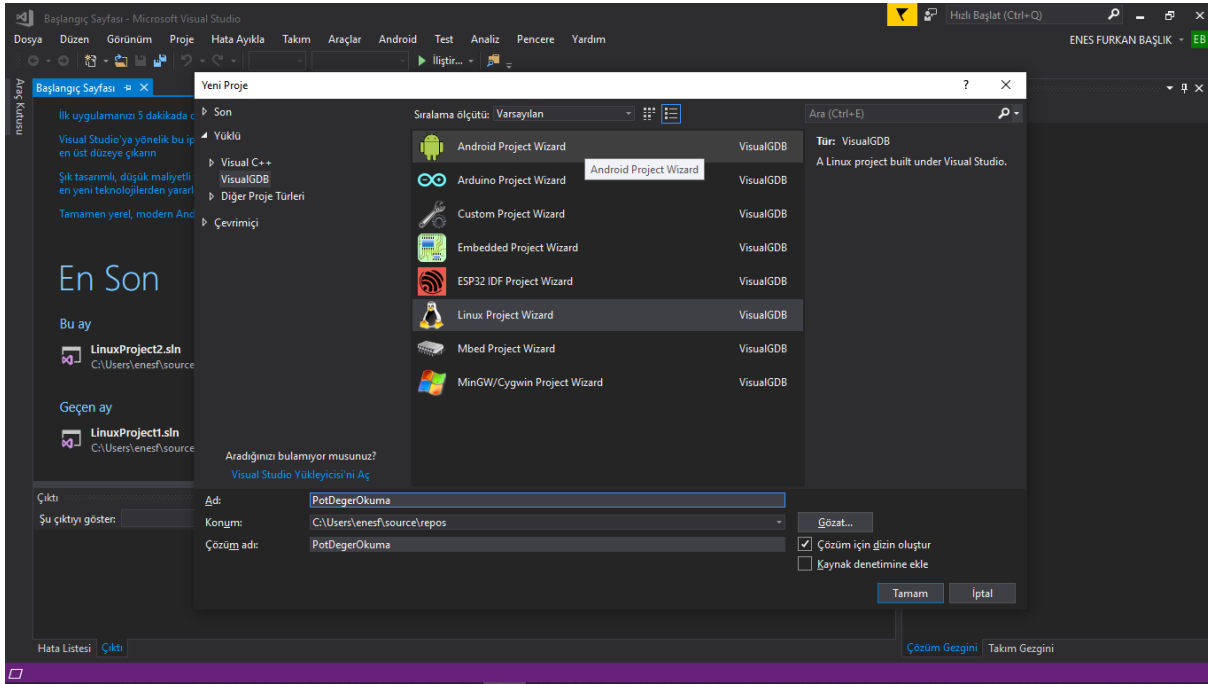
2. Visual GDB ile Visual Studio Kullanımı

2.1 Visual GDB Kurulumu

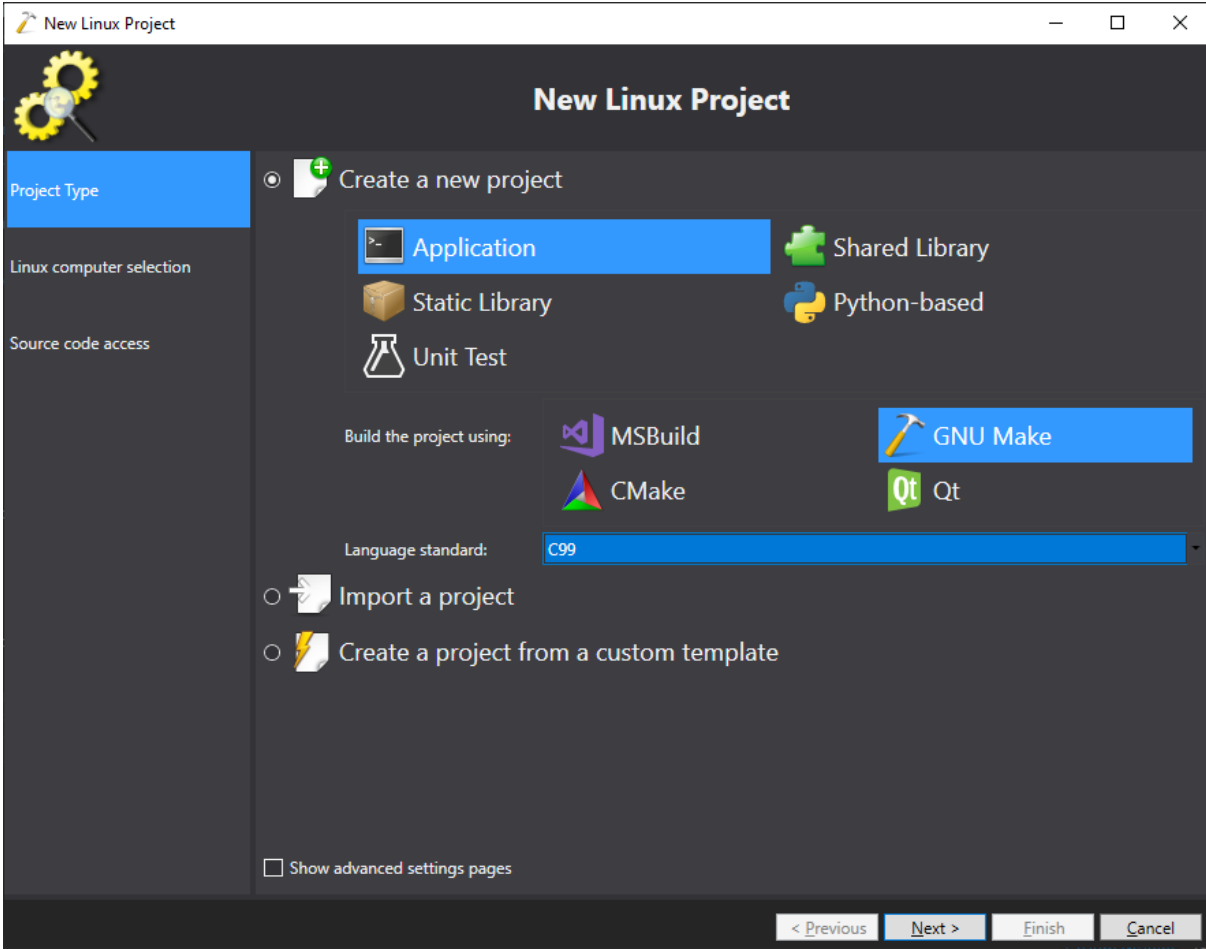
VisualGDB; Gömülü, Linux ve Android platformları için C/C++ desteği veren bir Visual Studio eklentisidir. <https://visualgdb.com/download/> bu bağlantıdan programı elde edebilirsiniz. Fakat bunun 30 günlük bir tam sürüm denemesi olduğunu unutmayın. Geliştirmeye devam edecekseniz programı satın almanız gerekmektedir. Ayrıca Visual Studio'ya sahip değilseniz onu da yüklemeniz gerekmektedir. Ben bu döküman için "Visual Studio 2017" sürümünü kullandım. Siz sahip değilseniz "Visual Studio Community Edition" programını ücretsiz olarak yükleyebilirsiniz. Visual GDB ve Visual Studio hazırısa artık geliştirmeye hazırız demektir. Visual GDB kurulumunu verdiğim bağlantıdaki adımları takip ederek kurduğunuzu varsayıyorum.

2.2 Visual Studio ile Proje Oluşturma

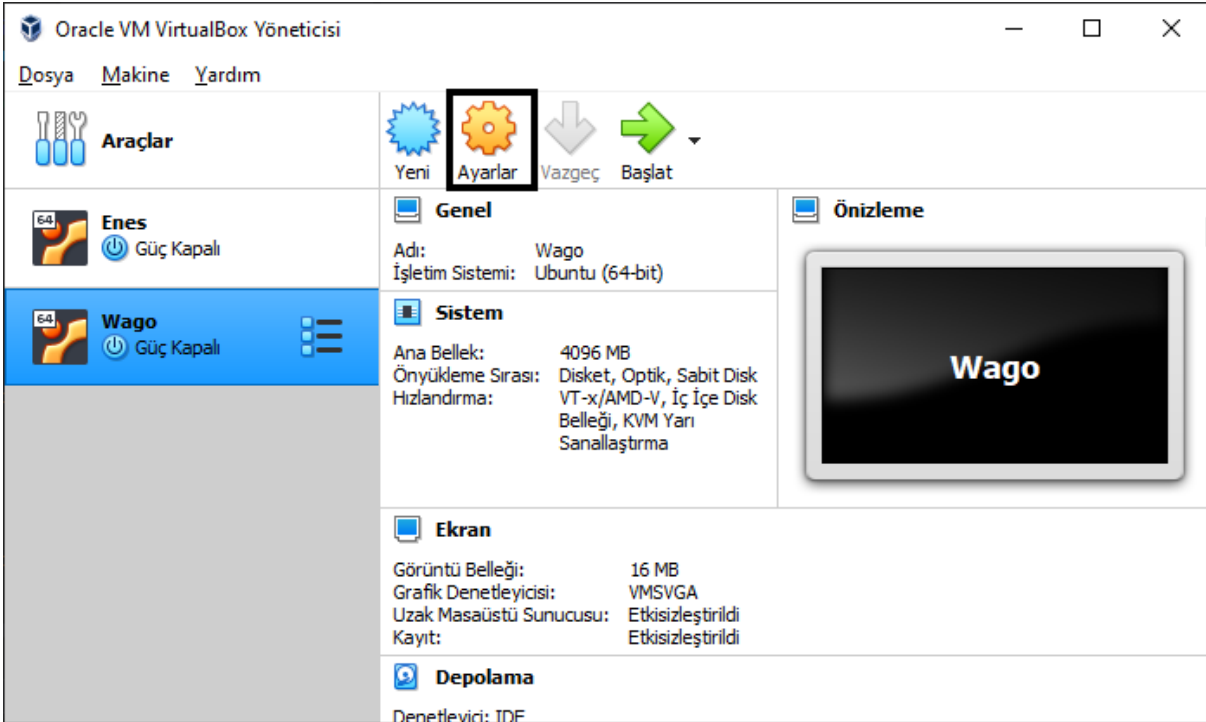
Visual Studio programını başlatın. Yeni bir proje oluştur diyerek daha önce kurduğumuz Visual GDB eklentisi ile bir proje oluşturacağız. Bu kısımda "Linux Project Wizard" seçerek ilerliyoruz. Ben bu dökümanda analog bir değer okuyacağım için adını "PotDegerOkuma" olarak kullandım. Siz dilediğiniz gibi devam edebilirsiniz. Karşınıza bir "pop-up" çıkarsa "OK" seçeneğine tıklayarak devam edin.



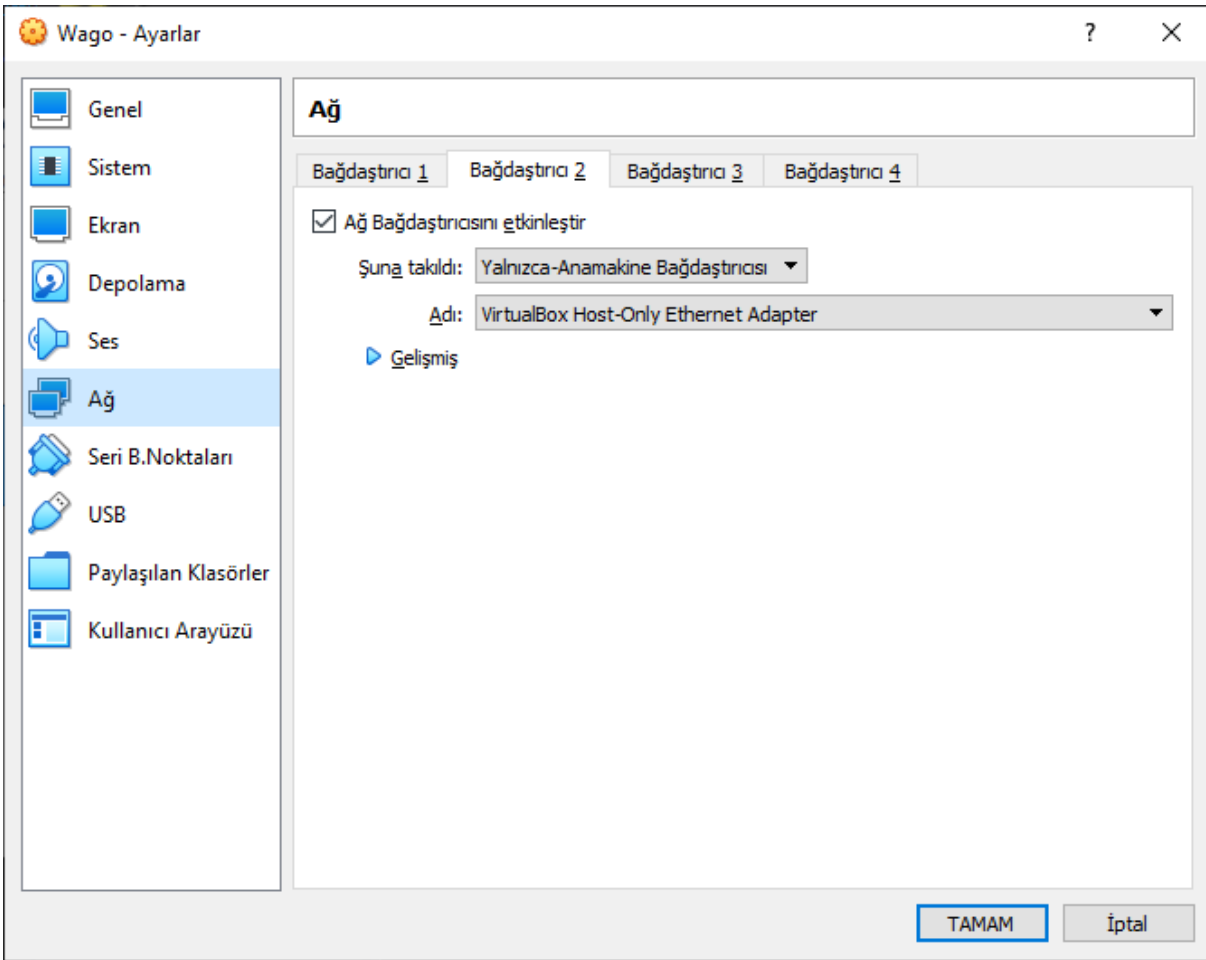
Daha sonra ayarlarımızı aşağıdaki gibi seçiyoruz ve "next" diyoruz.



Şimdi sırada kurduğumuz sanal bilgisayara “SSH” bağlantısı yapmak için bir “IP” gerekli olduğunu fark ediyoruz. Bunu bulmak için öncelikle birkaç küçük ayar yapıyoruz. Virtual Box programımızı tekrar açıyoruz. Ardından sanal bilgisayarımızı seçerek “Ayarlar” sekmesine giriyoruz.



Ayarlara girdiğimizde önce sol menüden “Ağ” sekmesine tıklıyoruz. Daha sonra “Bağdaştırıcı 1” sekmesine hiç dokunmadan “Bağdaştırıcı 2” sekmesine giriyoruz ve ayarları aşağıdaki gibi yapıyoruz. Ardından “Tamam” diyerek çıkış yapıyoruz. Şimdi sanal bilgisayarımızı tekrar başlatalım.



Açılan sanal makinemizde tekrar terminali açıyoruz ve aşağıdaki komutu giriyoruz.

```
>ifconfig
```

Daha sonra şöyle bir çıktı almanız gerekmekte:

```
wago@wago-VirtualBox: ~  
wago@wago-VirtualBox:~$ ifconfig  
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:82:bc:86  
        inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0  
        inet6 addr: fe80::aed4:49fe:775d:5740/64  Scope:Link  
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
        RX packets:26 errors:0 dropped:0 overruns:0 frame:0  
        TX packets:80 errors:0 dropped:0 overruns:0 carrier:0  
        collisions:0 txqueuelen:1000  
        RX bytes:8355 (8.3 KB)  TX bytes:8630 (8.6 KB)  
  
enp0s8  Link encap:Ethernet  HWaddr 08:00:27:8b:cf:39  
        inet addr:192.168.56.103  Bcast:192.168.56.255  Mask:255.255.255.0  
        inet6 addr: fe80::f8d3:c7d3:1a1e:808d/64  Scope:Link  
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
        RX packets:14 errors:0 dropped:0 overruns:0 frame:0  
        TX packets:53 errors:0 dropped:0 overruns:0 carrier:0  
        collisions:0 txqueuelen:1000  
        RX bytes:1971 (1.9 KB)  TX bytes:6799 (6.7 KB)  
  
lo      Link encap:Local Loopback  
        inet addr:127.0.0.1  Mask:255.0.0.0  
        inet6 addr: ::1/128  Scope:Sunucu  
        UP LOOPBACK RUNNING  MTU:65536  Metric:1  
        RX packets:212 errors:0 dropped:0 overruns:0 frame:0  
        TX packets:212 errors:0 dropped:0 overruns:0 carrier:0  
        collisions:0 txqueuelen:1000  
        RX bytes:15825 (15.8 KB)  TX bytes:15825 (15.8 KB)
```


Burada “enp0s8” kısmında yazan “inet addr” kısmı bizim SSH ile bağlanacağımız adres. İsterseniz aklınızda tutun, isterseniz bir yere yazın. Daha sonra bu adrese ihtiyacımız olacak. Şimdi belki dikkatinizi çekmiştir, adres “56” tabanında. Bu ne demek oluyor diye sorabilirsiniz. Bunun anlamı bizim sistemimizin de “56” tabanında çalışması gerekiyor demek. Sizde farklı bir adres çıkabilir, kendinize göre şekillendirebilirsiniz. Şimdi ise yapmamız gereken “SSH” bağlantısını oluşturmak. Öncelikle aşağıdaki komutu uyguluyoruz.

```
>sudo apt-get install openssh-server
```


Eğer bu da tamamlandıysa tekrar Visual Studio’ya geri dönebiliriz. Remote Computer satırının sonundaki aşağı yön okuna tıklayıp “Create New SSH Connection” kısmını seçiyoruz. Aşağıdaki gibi ayarları yapıp devam ediyoruz. Dikkatli olmanızda fayda var, kendi sanal bilgisayarımın adını kullanıyorum, kendi IP adresimi kullanıyorum ve tabi ki kendi parolamı kullanıyorum. Bunların hepsini sanal makineyi kurduğum sırada oluşturmuştum. Bunları siz kendi sisteminize göre değiştirebilirsiniz.

Create a New SSH Connection

Setup new SSH connection

 Host Name:

User Name:

 Authentication method

☒ Password:

☒ Setup public key authentication (more secure than saving the password)

☐ Public key in Windows key store (associated with your user account): Auto RSA DSA

☐ OpenSSH key

☐ Override default key location:

☐ Passphrase:

☐ Use HTTP CONNECT proxy:

☐ Enable zlib compression (recommended for slow connections)

☐ Emulate SCP file transfers with 'cat' (required for Dropbear SSH server with no SCP support)

Transfer folders using: On-the-fly TAR File-by-file SCP (slow)

☐ Enable keep-alive packets every: seconds

Create Cancel

Eğer bir pop-up çıkarsa "Save" diyerek devam edebilirsiniz. Şimdi sıra geldi PFC bağlantımıza. Wago PFC'ler genel olarak "192.168.1.17" şeklinde adreslendirilir. Eğer sizde farklıysa veya bilmiyorsanız resetleme yöntemine başvurabilirsiniz ya da servis kablosu ile güncel adresi öğrenebilirsiniz.

The image shows a screenshot of the WAGO Ethernet Settings application. A "Search device" dialog box is open, displaying the search results for the IP address 192.168.10.17. The dialog box has a title bar with "WAGO Ethernet Settings" and a close button. The main area is divided into two sections: "Address Range" and "Device List".

Address Range:

- from: 192 . 168 . 10 . 1
- to: 192 . 168 . 10 . 254
- ☐ Resolve network names
- ☐ Show all devices in network
- ☒ Use ping
- Search** button

Device List:


IP address	Item number ^	MAC address
✓ 192.168.10.17	750-8212	00:30:DE:47:1A:D0


At the bottom of the dialog box are "Apply" and "Cancel" buttons.


The background application window shows the "WAGO" logo, a "Settings" button, and a status bar at the bottom with the text "Ready" and the IP address "192.168.10.17".


Benim PFC adresim “192.168.10.17” şeklinde kalmış, şimdi bunu “56” tabanına çekmem gerekiyor ki aynı tabanda çalışabilsinler. Bu demek oluyor ki “10” yazan kısmı “56” olarak değiştirmeliyim.


750-8212 - WAGO Ethernet Settings





**750-8212**
WAGO 750-8212 PFC200 G2 2ETH RS


Open


Save

Read

Write

Restart

Factory Settings


Settings

IdentificationNetworkPLCStatus

Parameter	Edit	Currently used
Address Source	Static Configuration	Static Configuration
IP address	192.168.56.17	192.168.10.17
Subnet Mask	255.255.255.0	255.255.255.0
Gateway	192.168.10.1	192.168.10.1
Preferred DNS-Server	192.168.10.1	192.168.10.1
Alternative DNS-Server	0.0.0.0	0.0.0.0
Time server	0.0.0.0	not available
Hostname		PFC200V3-471AD0
Domain name	localdomain.lan	localdomain.lan

Interface X1


Interface X2

Run WBM

Interfaces

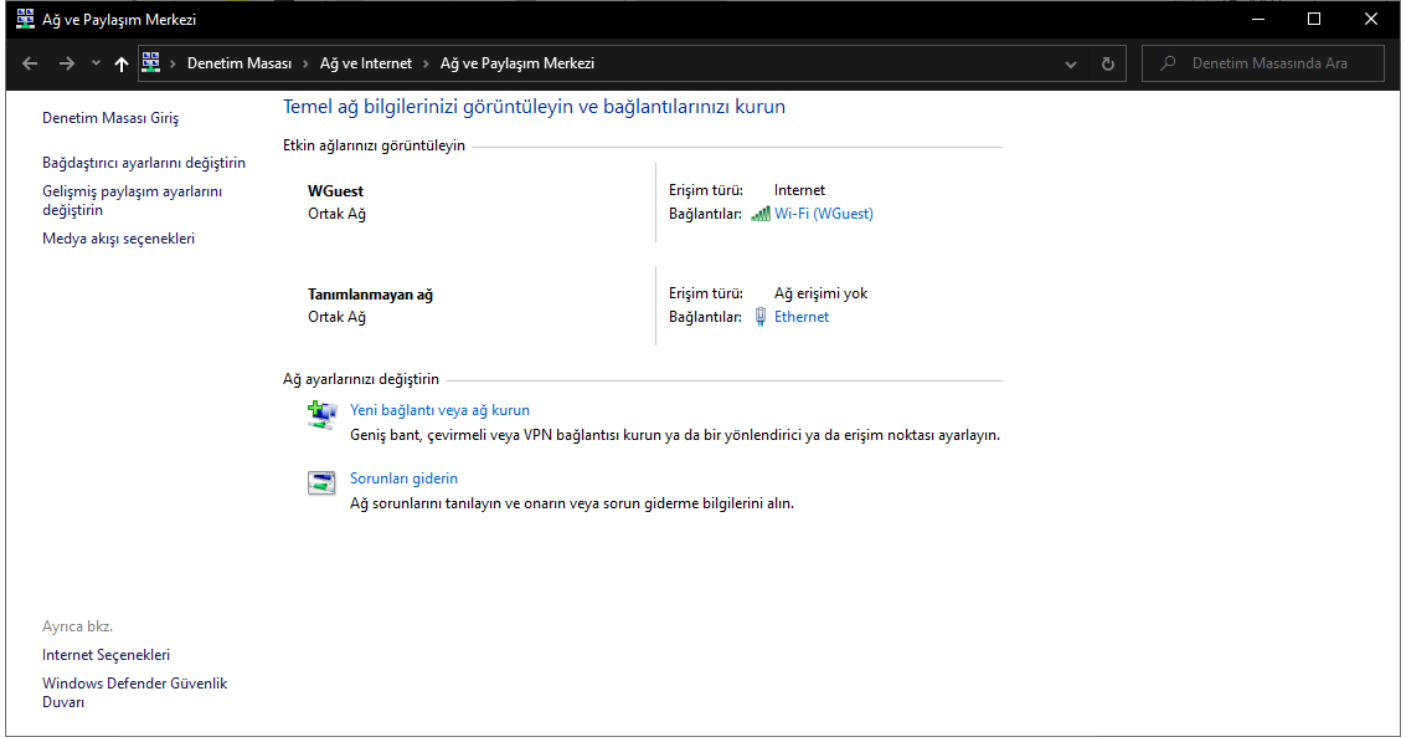
☐ Switched

☒ Separated

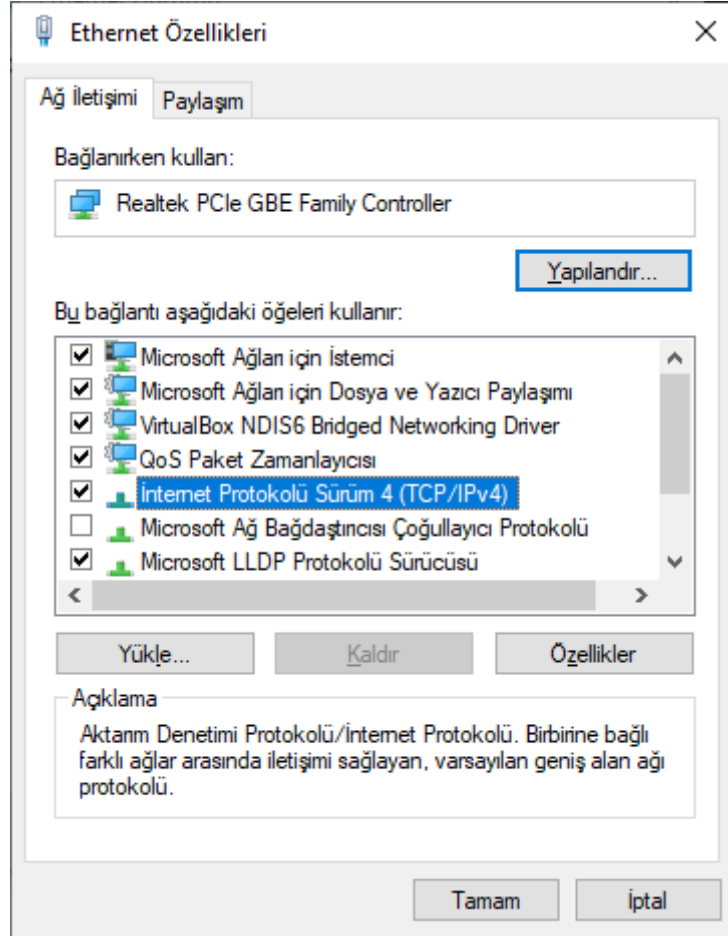
Ready192.168.10.17

Bunu yaptıktan sonra “Write” diyoruz. Çıkan uyarıya “Yes” dedikten sonra devam edebiliriz. Daha sonra bağlantı hatası aldınız değil mi? Eğer aldıysanız işler harika demektir.

Şimdi son basamak olarak kendi bilgisayarımızı da “56” tabanına çekiyoruz. Bilgisayarınızın “Ağ ve Paylaşım Merkezi” bunu yapacağımız yer.



Bu kısımda Ethernet kısmına bir kez tıklıyoruz. Karşımıza çıkan pencereden “Özellikler” kısmına tıklayarak giriş yapıyoruz. Mavi ile işaretli kısma iki kez tıklarak bir pencere daha açıyoruz.



Ayarlarımızı ařağıdaki gibi yapıyoruz. Siz “50” yerine başka bir sayı yazabilirsiniz ama diğerk cihazlarla çakışmamasına dikkat etmeniz gerekiyor. Bunu da hallettiğimize göre Visual Studio’ya keskin bir dönüş zamanı geldi.

İnternet Protokolü Sürüm 4 (TCP/IPv4) Özellikleri

Genel

Ağınız destekliyorsa, IP ayarlarının otomatik olarak atanmasını sağlayabilirsiniz. Aksi halde, IP ayarlarınız için ağ yöneticinize başvurmanız gerekir.

☐ Otomatik olarak bir IP adresi al

☒ Ařağıdaki IP adresini kullan:

IP adresi:

192 . 168 . 56 . 50

Alt ağ maskesi:

255 . 255 . 255 . 0

Varsayılan ağ geçidi:

. . .

☐ DNS sunucu adresini otomatik olarak al

☒ Ařağıdaki DNS sunucu adreslerini kullan:

Tercih edilen DNS sunucusu:

. . .

Diğerk DNS Sunucusu:

. . .


☐ Çıkarken ayarları doğrula


Gelişmiş...

Tamam


İptal

Şimdi ise aşağıdaki satırda en sağdaki küçük anahtara tıklayarak bir sonraki pencereyi açıyoruz. Açılan pencerede ayarları aşağıdaki gibi değiştirmeniz gerekmektedir. Eğer en başından beri dökümanın kopyası şeklinde ilerliyorsanız daha aşağıda size kopyalamanız için bu bilgileri vermiş olacağım. Eğer sanal bilgisayarınızın adını “wago” olarak değil başka bir isim olarak seçtiyseniz /home/ sekmesinden sonraki kısmı kendi sanal bilgisayarınızın adı olacak şekilde değiştirebilirsiniz. Dilerseniz yandaki klasör işaretleri ile arayabilirsiniz. Tercih sizin.

Remote toolchain:  Default GCC-based toolchain on 192.168.56.103

 Edit Toolchain







Toolchain summary:

Unique Toolchain ID:
 Unique ID will be used to locate this toolchain on other computers.

Location:

Name:

Locations of individual tools:

gcc:	<input type="text" value="/home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-gcc"/>	
g++:	<input type="text" value="/home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-g++"/>	
gdb:	<input type="text" value="/home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-gdb"/>	
ar:	<input type="text" value="ar"/>	
objcopy:	<input type="text" value="objcopy"/>	
GNU Make:	<input type="text" value="make"/>	

Advanced settings:

Environment Setup File:

Extra PATH directories:

CMake executable:

gcc: /home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-gcc

g++: /home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-g++

gdb: /home/wago/wago/ptxproj/selected_toolchain/arm-linux-gnueabi-hf-gdb


extra path directories: /home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target

Şimdi sırada PFC için ayarlar var. Yine “Create New SSH Connection” diyoruz. Ayarları aşağıdaki gibi yapıp yola devam edeceğiz. Parola değiştirilmediyse “wago” olarak girmeniz gerekiyor. “Setup public key authentication” tikini kaldırmanızı tavsiye ederim. Daha sonra “Create” seçeneğine tıklıyoruz.


Create a New SSH Connection

— □ ×


Setup new SSH connection



Host Name: 192.168.56.17



User Name: root



Authentication method


☒ Password:

☐ Setup public key authentication (more secure than saving the password)

☐ Public key in Windows key store (associated with your user account):

Auto RSA DSA

☐ OpenSSH key

☐ Override default key location: 

☐ Passphrase:

☐ Use HTTP CONNECT proxy:

☐ Enable zlib compression (recommended for slow connections)

☐ Emulate SCP file transfers with 'cat' (required for Dropbear SSH server with no SCP support)

Transfer folders using:

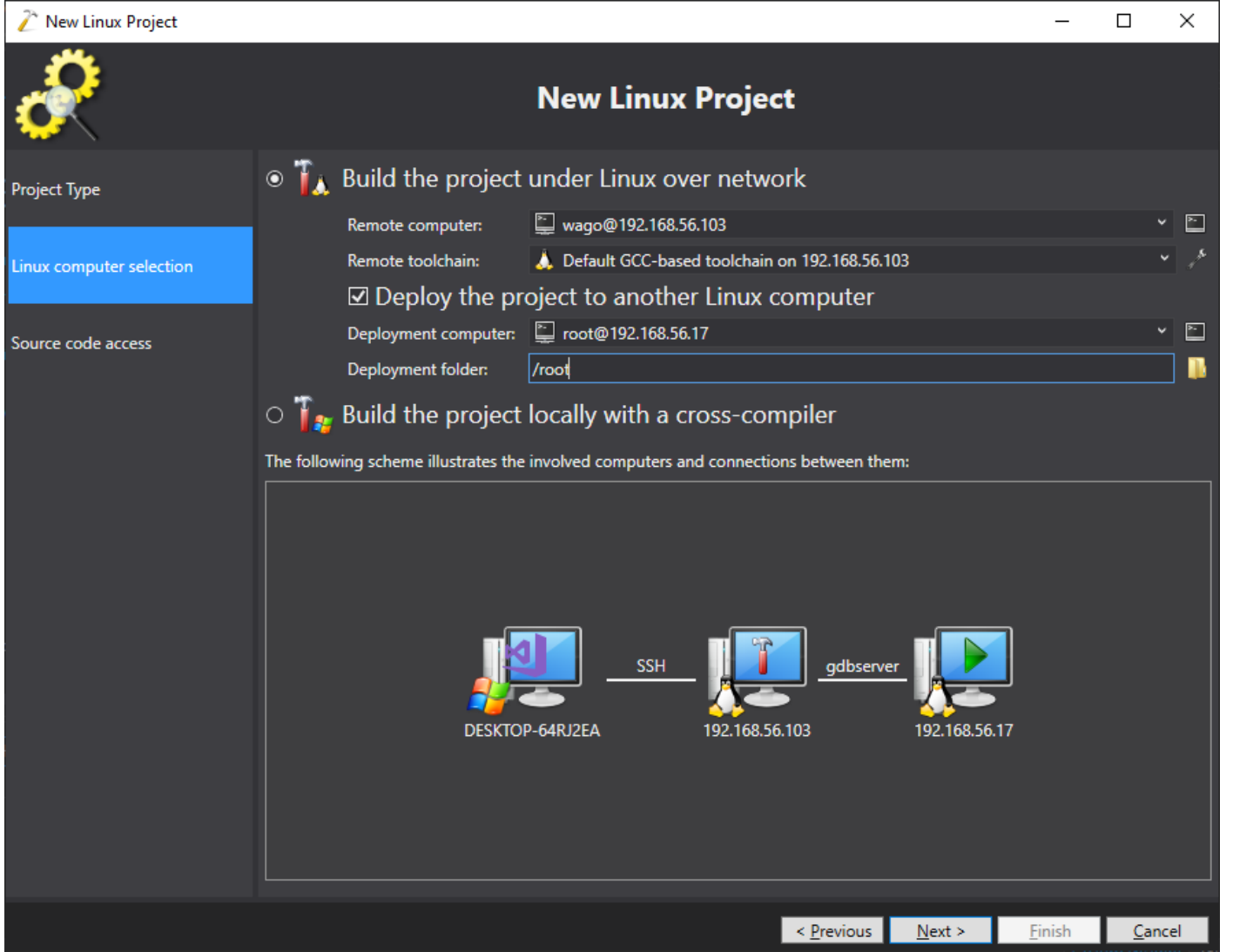
On-the-fly TAR File-by-file SCP (slow)

☐ Enable keep-alive packets every: seconds

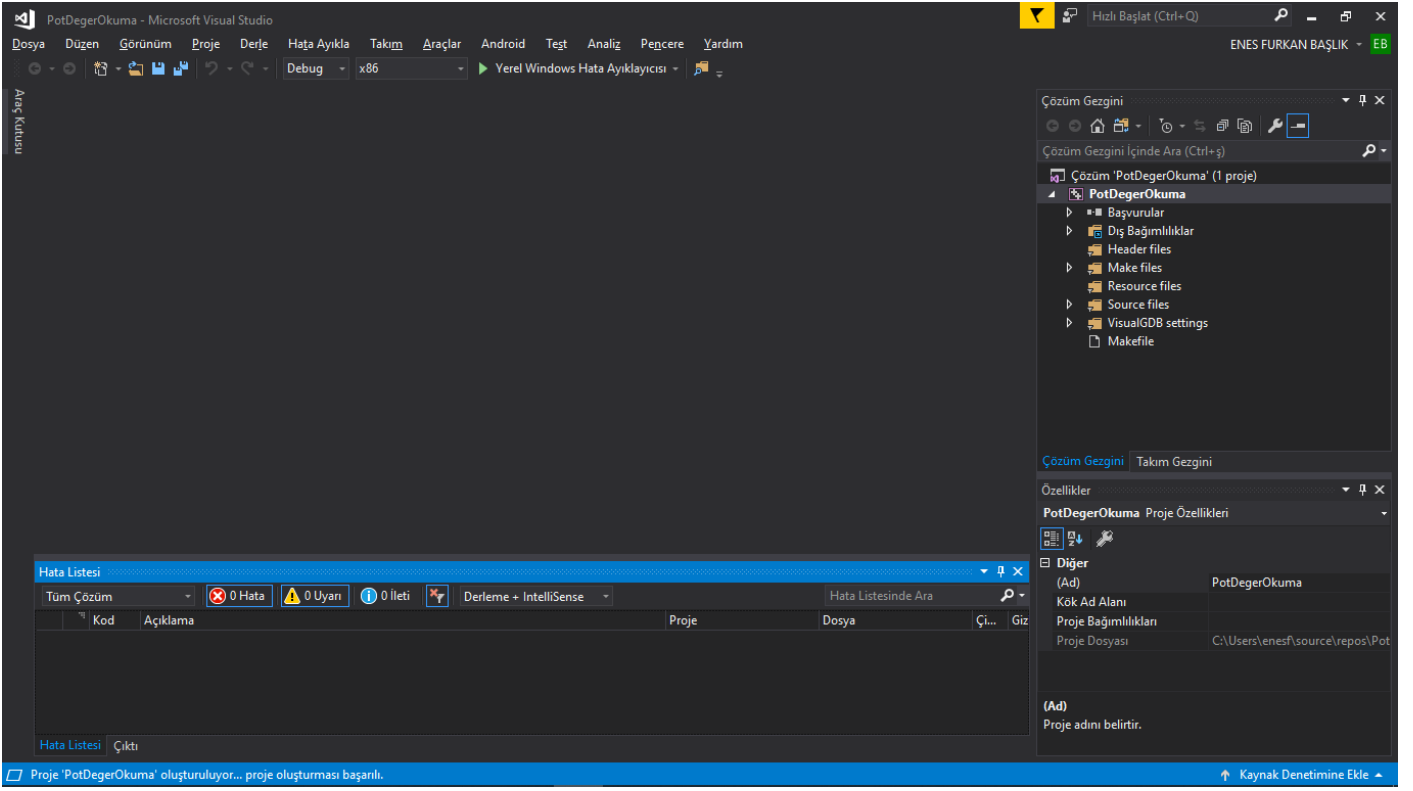
Create

Cancel

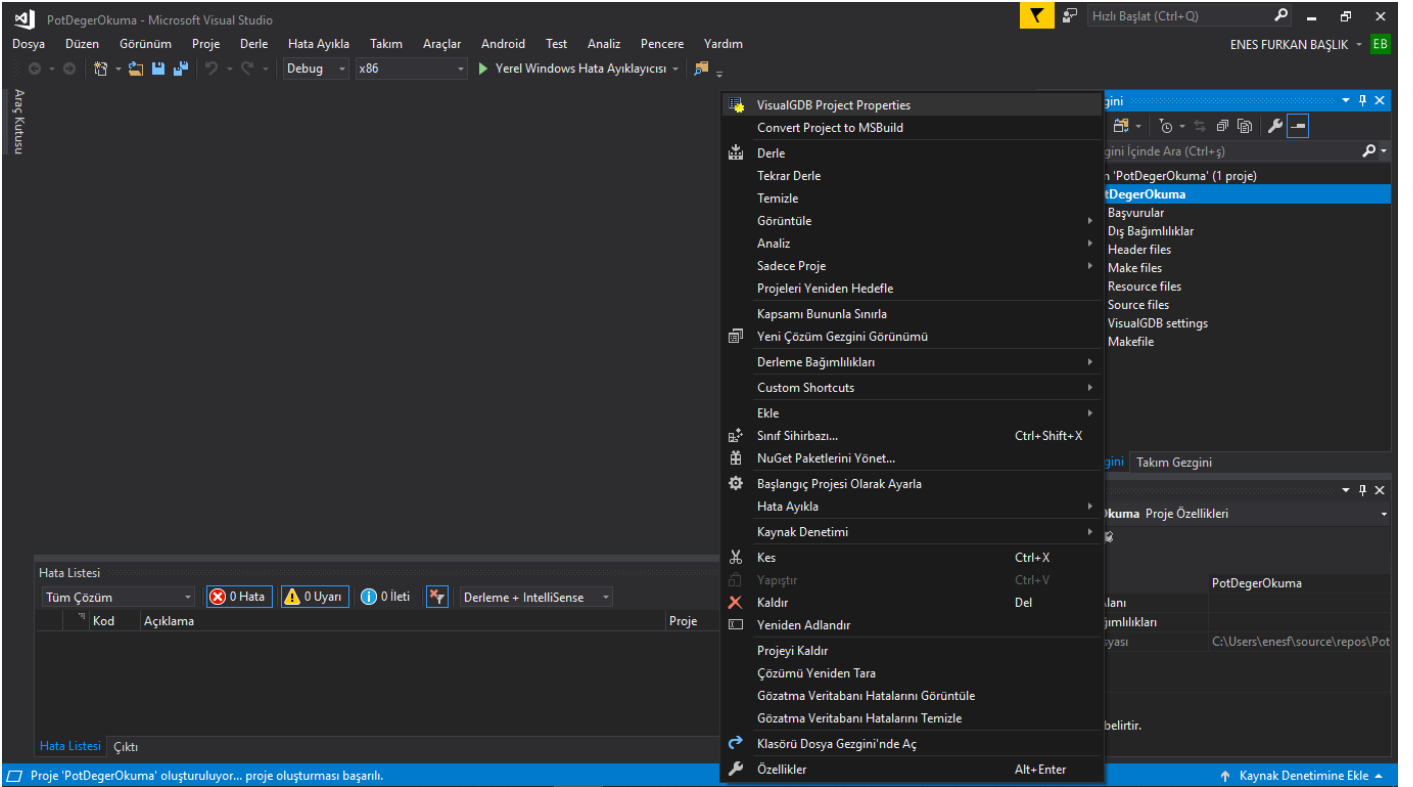
En sonunda aşağıdaki gibi bir görüntü elde etmeniz gerekiyor. “Deployment folder” kısmını “/root” olarak düzeltmeyi unutmayın.

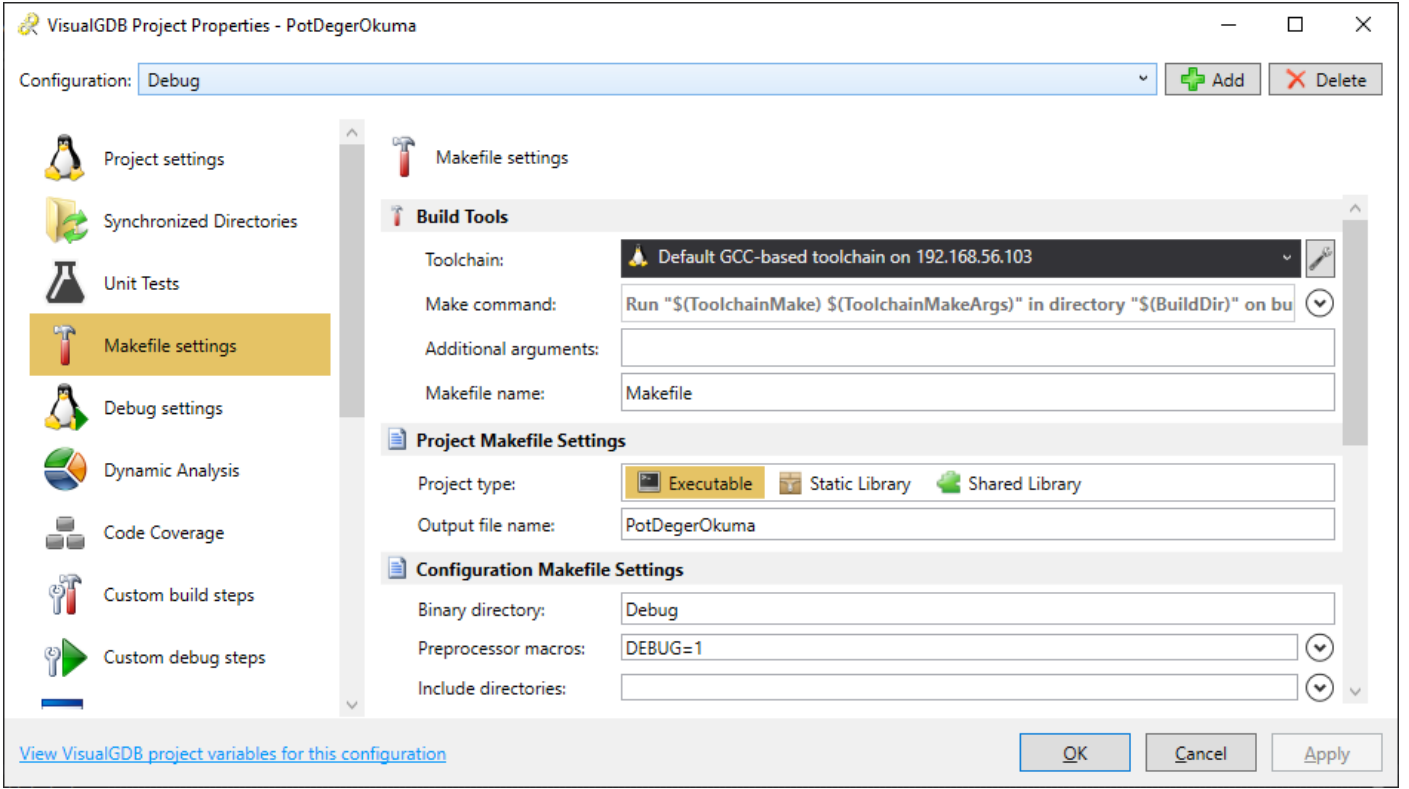


Tiklerinizi sıra sıra gördüyseniz sizin için işler hala yolunda. Karşınıza pencereler çıkabilir. “OK” diyoruz ve devam ediyoruz. Aşağıdaki ekran geldiğinde hiçbir şeye dokunmadan “Finish” diyoruz. En sonunda “C” programlama yeteneklerinizi göstereceğiniz alana geldik. Şimdi kendi örneğim için bir “Makefile” oluşturacağım.

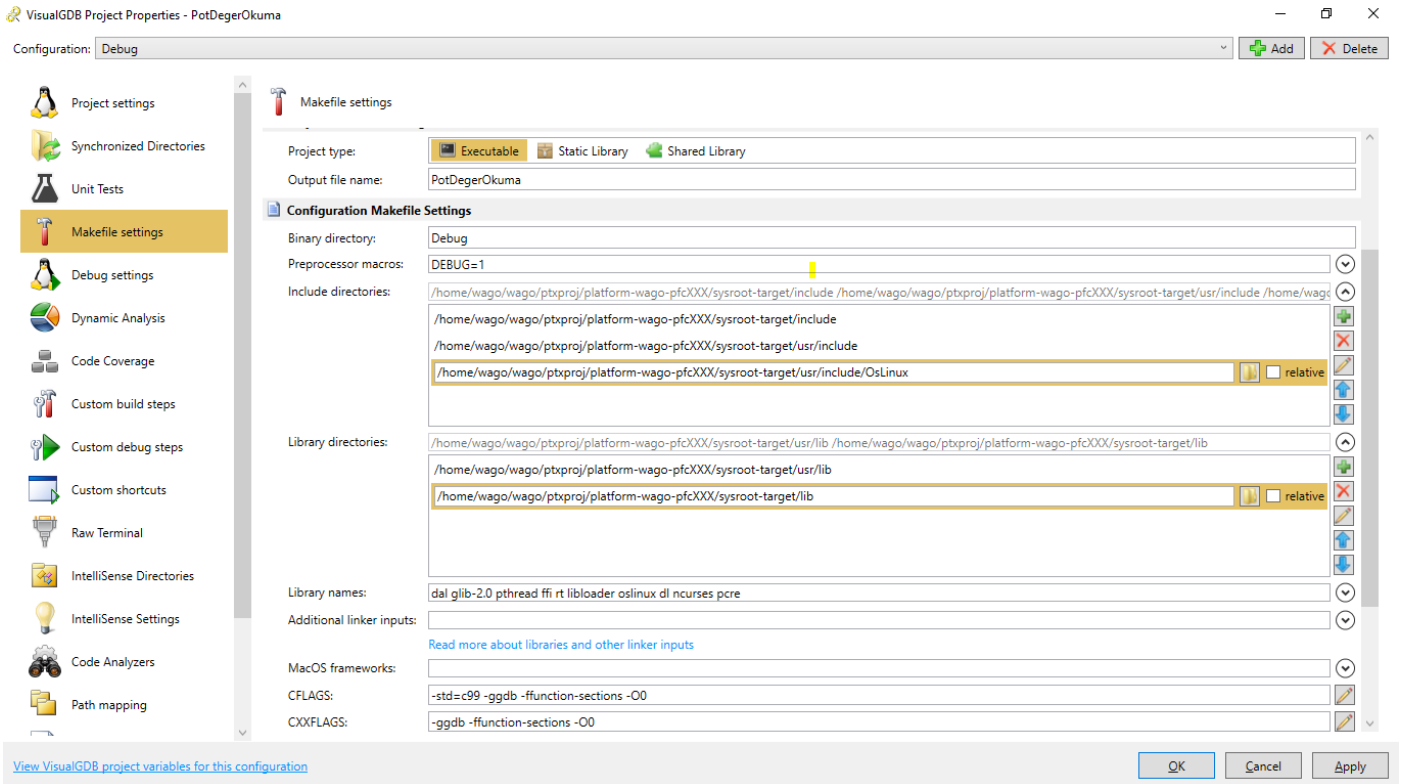


Projenizin adına sağ tıklayıp “VisualGDB Project Properties” seçeneğine tıklıyoruz ve “Makefile Settings” sekmesine giriyoruz.





Ekleme istediğiniz kütüphaneleri ve dosyaları bu kısımda seçiyoruz. Eğer dokümana sadık kaldıysanız sizin için alttaki görselden sonra yapıştırabilmeniz için dosya konumlarını paylaşacağım. Eğer farklı isimlere sahipseniz klasörleri yandaki klasör işaretine tıklayarak aramanız gerekmektedir. Yine de çok farklı yerlerde olmayacağı için kolayca bulabilirsiniz. Ayrıca relative sekmelerinin tikini kaldırmanızı tavsiye ederim. Tüm bunları tamamladıktan sonra “OK” diyoruz.



Include directories:

```
/home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target/include
```

```
/home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target/usr/include
```

```
/home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target/usr/include/OSLinux
```

Library directories:

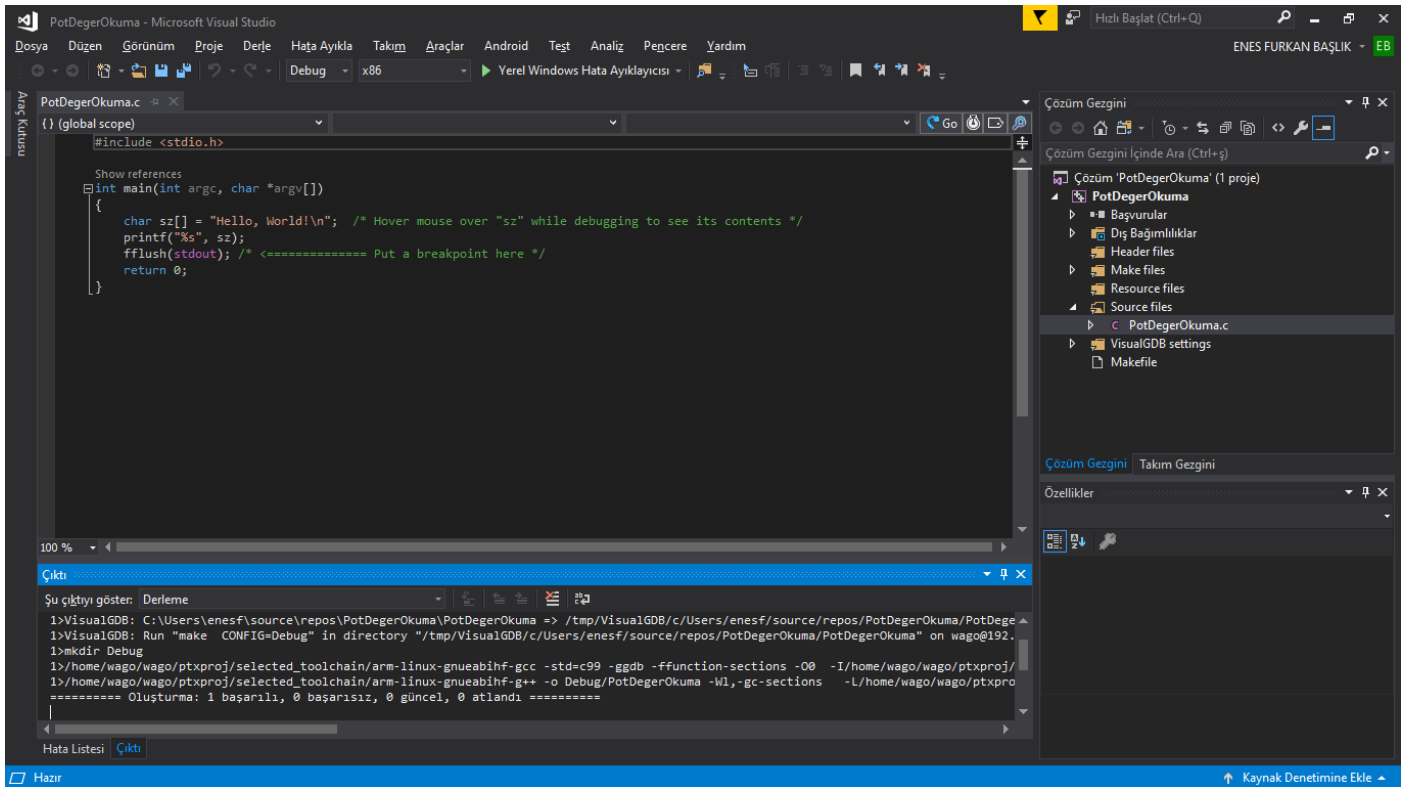
```
/home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target/usr/lib
```

```
/home/wago/wago/ptxproj/platform-wago-pfcXXX/sysroot-target/lib
```

Library names:

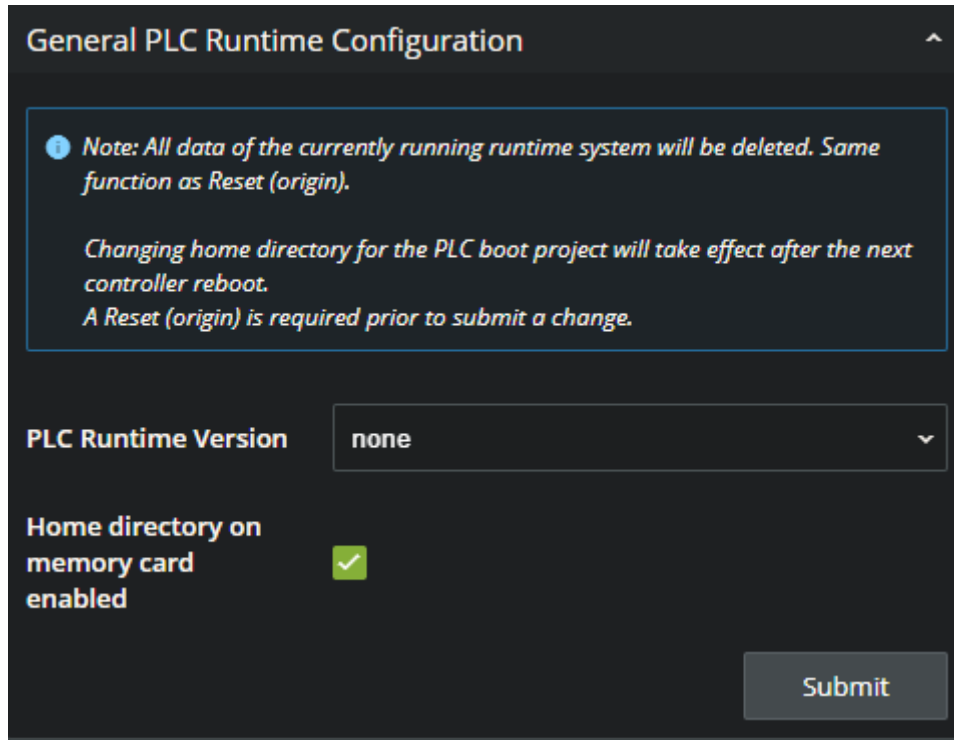
```
dal glib-2.0 pthread ffi rt libloader oslinux dl ncurses pcre
```

Sırada ne var dediğinizi duyar gibiyim. Bundan sonrası size ait diyebilirim. İstedığınız bir “C” kodunu yazıp PFC içerisinde çalıştırabilirsiniz. Örnek olması adına bir adet okuma ve yazma işlemi yapan kod paylaşacağım. Ayrıca ADI/DAL Kütüphanesinin kullanımı ile ilgili dokümanı da klasörün içine ekliyorum. Karşınızda şu an aşağıdaki gibi bir ekran olmalı.



ÖNEMLİ NOTLAR:

PLC Runtime Version “none” olmalı. Aksi takdirde “No KBUS device found” hatası alırsınız. PFC’nin “IP” adresini tarayıcınıza girdiğinizde karşınıza bir “Web-Management” kısmı çıkacaktır. Varsayılan olarak kullanıcı adı ve şifresi sırasıyla “admin” ve “wago” olarak ayarlanmıştır. Bu kısımda aşağıdaki ayarlamayı yapabilirsiniz.



The image shows a 'General PLC Runtime Configuration' window. At the top, there is a note: 'Note: All data of the currently running runtime system will be deleted. Same function as Reset (origin). Changing home directory for the PLC boot project will take effect after the next controller reboot. A Reset (origin) is required prior to submit a change.' Below the note, there are two settings: 'PLC Runtime Version' set to 'none' and 'Home directory on memory card enabled' with a green checkmark. A 'Submit' button is at the bottom right.

General PLC Runtime Configuration

Note: All data of the currently running runtime system will be deleted. Same function as Reset (origin).
Changing home directory for the PLC boot project will take effect after the next controller reboot.
A Reset (origin) is required prior to submit a change.

PLC Runtime Version none

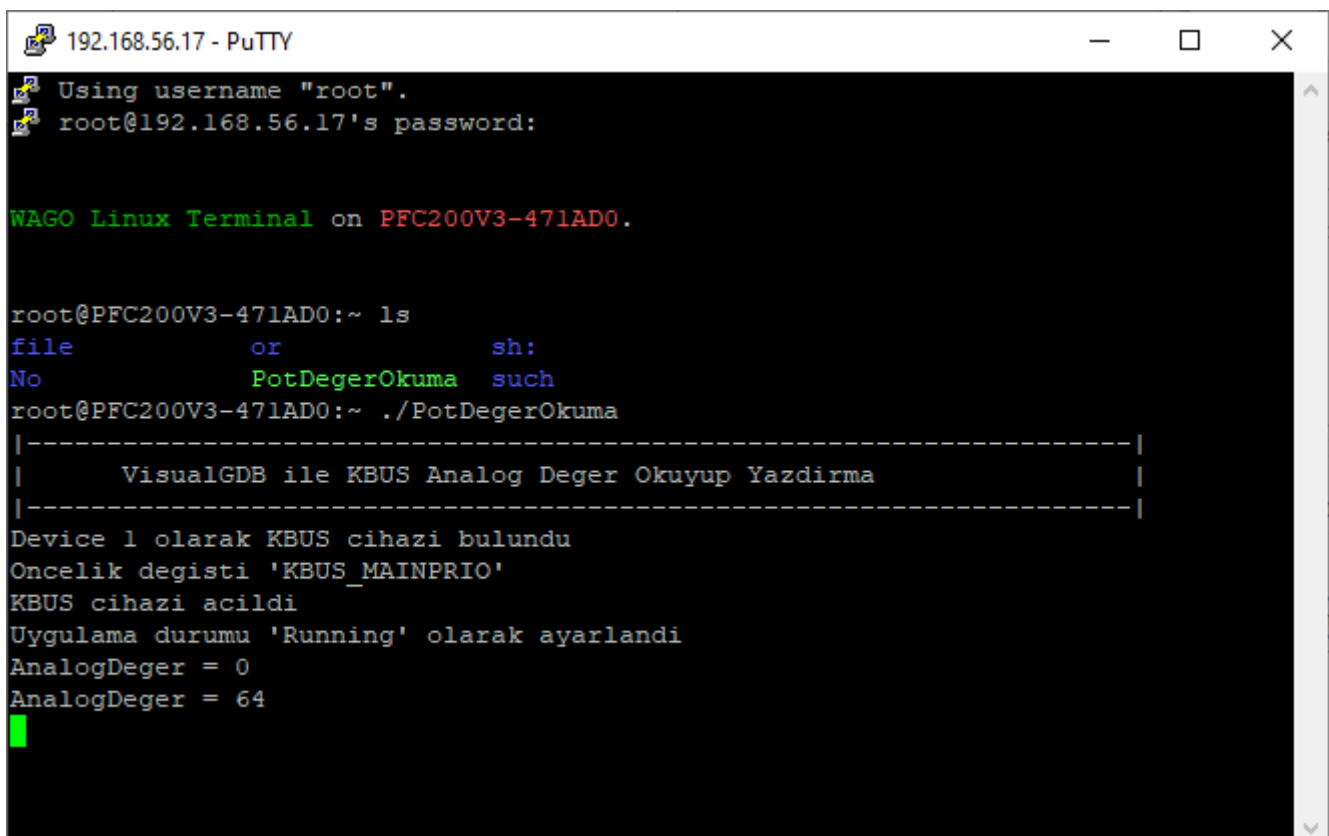
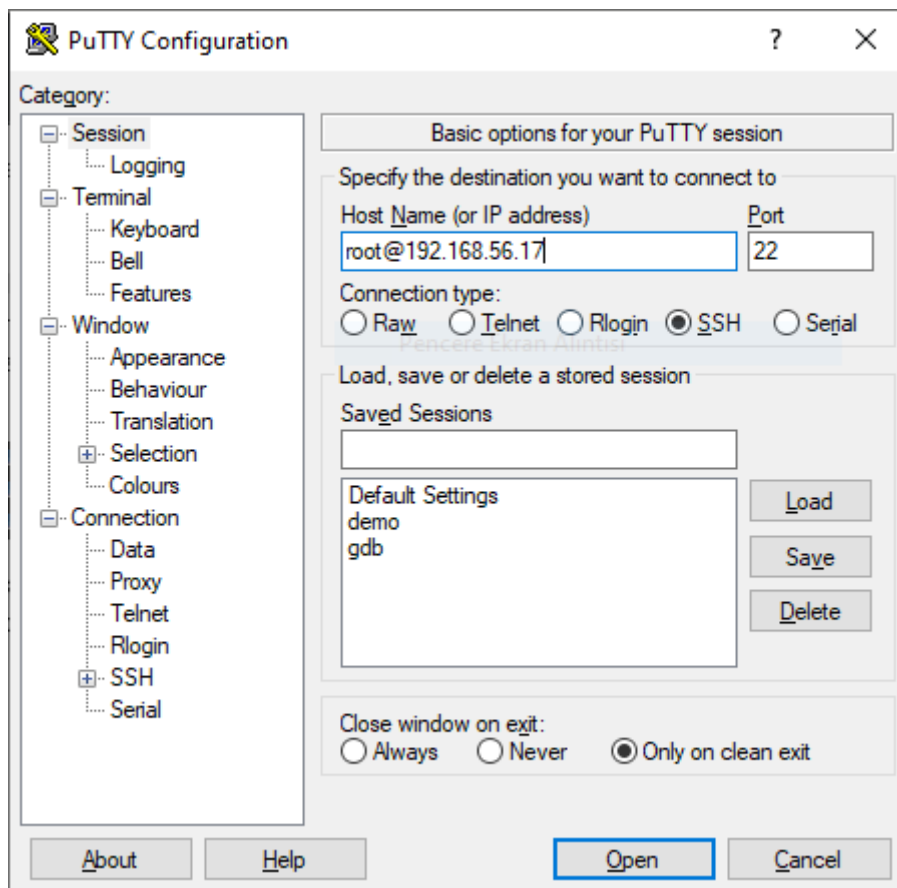
Home directory on memory card enabled ☒

Submit

PFC adreslemelerini yaparken sırasıyla analoglar öncelikle yapar. Bu da demek oluyor ki eğer analog ve dijital inputları bir arada kullanmak istiyorsanız dijital inputlarınızdan veri okumak için veya dijital çıkışlarınızdan çıkış vermek için adresinizi ötelemeniz gerekmektedir. Aşağıdaki örnekte kullandığım sette bir adet “753-889” analog çıkış kullandığım için bir miktar ötelemek zorunda kaldığımı fark etmelisiniz. 4 kanallı bir modül olduğu, ayrıca her kanal 2 byte alan kapladığı için 8 byte öteleyerek yazdırmaya başlıyoruz. Aşağıdaki örnekte 8 sayısı öteleme sayıysyken, 1 sayısı ise yazdırılmak istenen byte sayısını temsil ediyor. Bu işlemleri yaparken izlediğiniz yolu girişleri okurken de izlemelisiniz.

```
adi->WriteStart(kbusDeviceId, taskId); // PD-cikis verisini kilitle  
adi->WriteBytes(kbusDeviceId, taskId, 8, 1, (uint8_t*)&pd_out[0]); // yaz  
adi->WriteEnd(kbusDeviceId, taskId); // PD-cikis verisinin kilidini kaldır
```

Bir kez build ettiğiniz koda tekrar erişmek için bütün bu işlemleri tekrardan yapmanıza gerek yoktur. SSH bağlantısı yapabileceğiniz bir programla (PuTTY vb.) tekrar çalıştırabilirsiniz.



- Visual Studio
- VisualGDB
- WAGO Ethernet Settings