

## APPLICATION NOTE

```
// Create an instant camera object with the first
Camera_t camera( CTIFactory::GetInstance().Creat

// Register an image event handler that accesses
camera.RegisterImageEventHandler( new CSampleImage
Ownership_TakeOwnership);

// Open the camera.
camera.Open();
```

### How to build pylon applications for ARM

Version: 01    Language: 000 (English)

Release Date: 31 January 2014

# How to build pylon applications for ARM

## Table of Contents

<b>1</b>	<b>Introduction.....</b>	<b>2</b>
<b>2</b>	<b>Steps.....</b>	<b>2</b>

## 1 Introduction

This document explains how pylon applications can be built on a given Linux distribution and be run on ARM based machines afterwards.

The procedures described in this document assume you are using a MiraBox Development Kit (<http://www.globalscaletechnologies.com/>), an Oracle VM VirtualBox (<https://www.virtualbox.org/>), a Windows 7 (32 or 64bit) machine and a Linux ISO image of a current Linux distribution such as Ubuntu (12 or higher) or Fedora (12 or higher).

It is also assumed that on your Windows PC you are using one network adapter connected to internet and there is at least another one network adapter (Gigabit Ethernet) for connecting with the MiraBox.

## 2 Steps

### 1. Installation

- a. We first recommend downloading and installing the Oracle VM VirtualBox on your Windows 7 machine.
- b. Now create a new virtual machine using a Linux ISO image and make sure you have an internet connection on your Linux virtual machine. In our case an Ubuntu 12.04LTS x86 image was used.

c. Download the current pylon for Linux and pylon for Linux ARM version. The procedures described in this document assume you are using pylon v3.2.0.

The pylon software can be downloaded under:

<http://www.baslerweb.com/>

d. On your Windows PC download/install the MiraBox User Guide, the serial communication tool “putty.exe” and the Prolific-USB-to-Serial-Comm-Port driver “2KXPVDock.exe”, (<http://www.globalscaletechnologies.com/t-downloads.aspx> ) and follow the instruction in order to establish a connection to the MiraBox (Chapters 1 and 2).

For your convenience you may also download all necessary tools from the Basler FTP server:

<ftp://Pylon4Linux-ro:h50UZgkl@ftp.baslerweb.com/ARM-Cross-Toolchain/>

## 2. Setup

a. After the debugging console was started (i.e. through “putty.exe”) and you logged in using the following login information:

Login: root

Password: nosoup4u

for test purposes you may type in the following command in order to display the network configuration of your MiraBox:

```
# ifconfig
```



```
root
Password:
Last login: Tue Jan 28 12:45:34 UTC 2014 from 192.168.3.2 on pts/1
Linux mirabox-debian 2.6.35.9 #12 Thu Aug 23 22:13:28 EDT 2012 armv7l

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@mirabox-debian:~# ifconfig
eth0      Link encap:Ethernet  HWaddr f0:ad:4e:01:a4:27
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:1894 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1294 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:512
          RX bytes:228600 (223.2 KiB)  TX bytes:2733859 (2.6 MiB)
          Interrupt:8

eth1      Link encap:Ethernet  HWaddr f0:ad:4e:01:a4:28
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:512
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:10

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:26 errors:0 dropped:0 overruns:0 frame:0
          TX packets:26 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1800 (1.7 KiB)  TX bytes:1800 (1.7 KiB)

uap0      Link encap:Ethernet  HWaddr 94:db:c9:d2:d8:92
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@mirabox-debian:~#
```

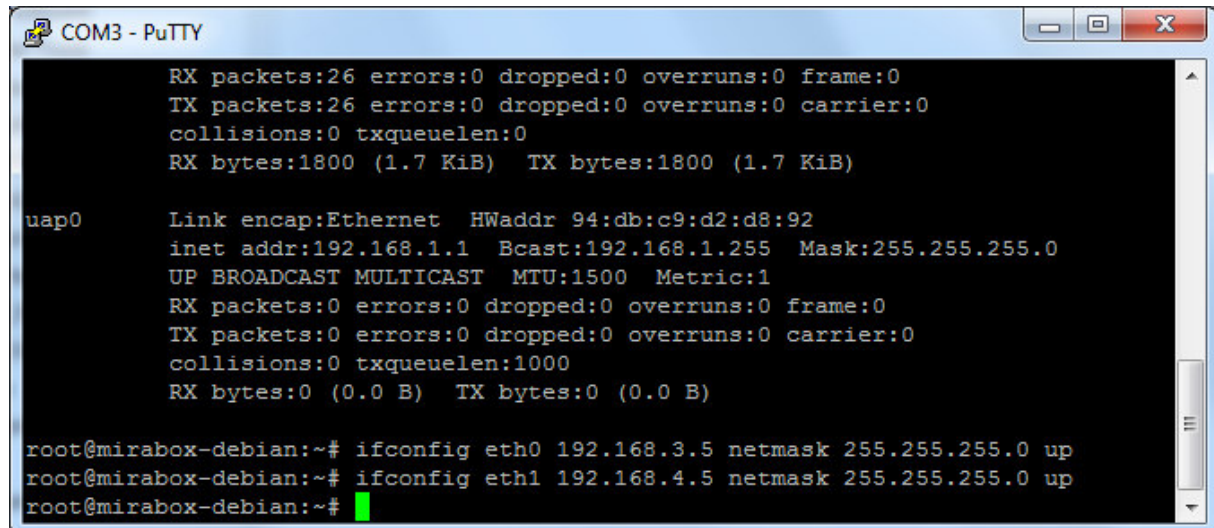
b. As the MiraBox supports two built-in Gigabit Ethernet network adapters we have to configure them in different subnets using e.g. Class C IP addresses in order to avoid any IP address conflicts and:

- 1) in order to be able to establish a connection between the MiraBox and the Windows/Linux machine using the first port (eth0)
- 2) to establish a connection between the MiraBox and a Basler GigE Vision camera on the second port (eth1).

In order to configure the adapters in different subnets execute the following lines:

```
# ifconfig eth0 192.168.3.5 netmask 255.255.255.0 up
```

```
# ifconfig eth1 192.168.4.5 netmask 255.255.255.0 up
```



```
COM3 - PuTTY

RX packets:26 errors:0 dropped:0 overruns:0 frame:0
TX packets:26 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:1800 (1.7 KiB) TX bytes:1800 (1.7 KiB)

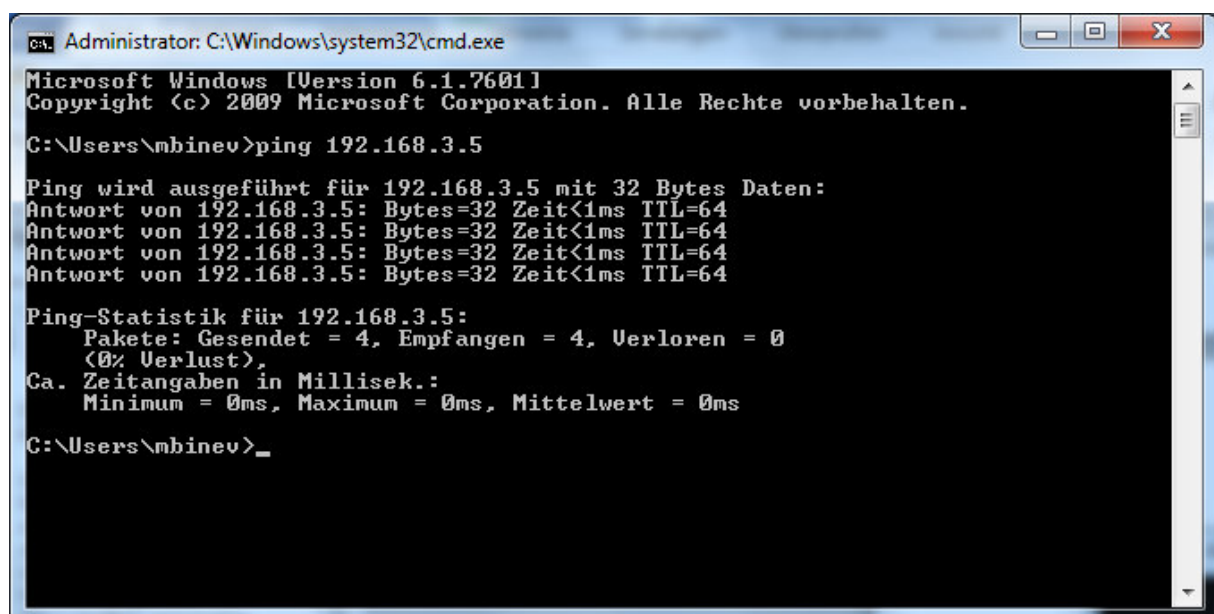
uap0    Link encap:Ethernet  HWaddr 94:db:c9:d2:d8:92
        inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
        UP BROADCAST MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@mirabox-debian:~# ifconfig eth0 192.168.3.5 netmask 255.255.255.0 up
root@mirabox-debian:~# ifconfig eth1 192.168.4.5 netmask 255.255.255.0 up
root@mirabox-debian:~#
```

c. Now connect Port Nr.1 (eth0) of the MiraBox (the one next to the power connector) to the free adapter port of your Windows PC. This document assumes that you have already configured this port in the same subnet as eth0, e.g. IP: 192.168.3.2, Subnet Mask: 255.255.255.0.

For test purposes you may run the Windows Command Prompt (cmd) and type in the following command line in order to check if the connection was established successfully:

```
# ping 192.168.3.5
```



```
Administrator: C:\Windows\system32\cmd.exe

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\mbinev>ping 192.168.3.5

Ping wird ausgeführt für 192.168.3.5 mit 32 Bytes Daten:
Antwort von 192.168.3.5: Bytes=32 Zeit<1ms TTL=64
Antwort von 192.168.3.5: Bytes=32 Zeit<1ms TTL=64
Antwort von 192.168.3.5: Bytes=32 Zeit<1ms TTL=64
Antwort von 192.168.3.5: Bytes=32 Zeit<1ms TTL=64

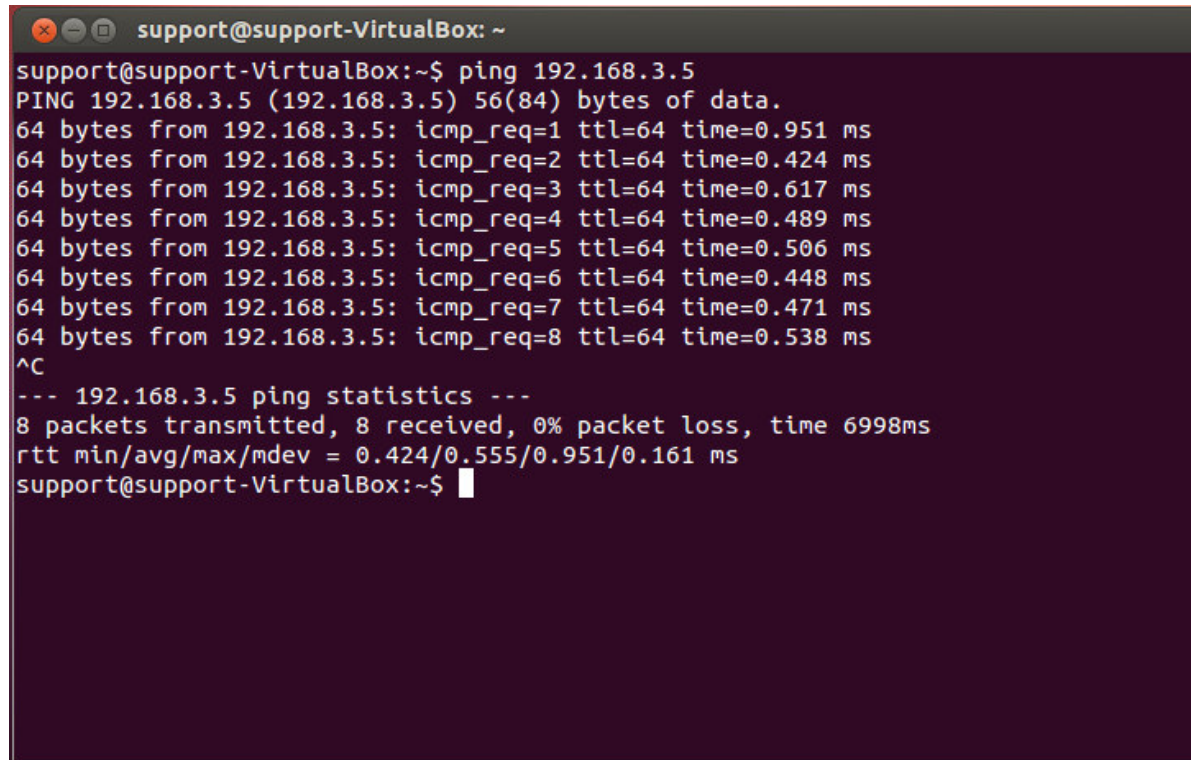
Ping-Statistik für 192.168.3.5:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0
            (0% Verlust),
    Ca. Zeitangaben in Millisek.:
        Minimum = 0ms, Maximum = 0ms, Mittelwert = 0ms

C:\Users\mbinev>_
```



d. Now go to your Linux machine that is running on the Oracle VM VirtualBox and run a terminal (console).

For test purposes you may type in the same command as described in the previous step in order to check if there is a connection between your Linux machine and the MiraBox.



```
support@support-VirtualBox: ~  
support@support-VirtualBox:~$ ping 192.168.3.5  
PING 192.168.3.5 (192.168.3.5) 56(84) bytes of data.  
64 bytes from 192.168.3.5: icmp_req=1 ttl=64 time=0.951 ms  
64 bytes from 192.168.3.5: icmp_req=2 ttl=64 time=0.424 ms  
64 bytes from 192.168.3.5: icmp_req=3 ttl=64 time=0.617 ms  
64 bytes from 192.168.3.5: icmp_req=4 ttl=64 time=0.489 ms  
64 bytes from 192.168.3.5: icmp_req=5 ttl=64 time=0.506 ms  
64 bytes from 192.168.3.5: icmp_req=6 ttl=64 time=0.448 ms  
64 bytes from 192.168.3.5: icmp_req=7 ttl=64 time=0.471 ms  
64 bytes from 192.168.3.5: icmp_req=8 ttl=64 time=0.538 ms  
^C  
--- 192.168.3.5 ping statistics ---  
8 packets transmitted, 8 received, 0% packet loss, time 6998ms  
rtt min/avg/max/mdev = 0.424/0.555/0.951/0.161 ms  
support@support-VirtualBox:~$
```

e. For test purposes unpack the pylon for Linux package (nor ARM) and refer to the INSTALL text file in order to install pylon correctly (in our case the package was unpacked on the Desktop):

```
# cd Desktop/pylon-3.2.0-x86
```

Set the pylon environment variables:

```
# source ./pylon3/bin/pylon-setup-env.sh pylon3
```

For test purposes, make sure you can build one the pylon SDK samples:

```
# cd Sampes/Grab
```

```
# make
```

```

support@support-VirtualBox: ~/Desktop/pylon-3.2.0-x86/Samples/Grab
support@support-VirtualBox:~$ cd Desktop/pylon-
pylon-2.3.3-1337-bininst/ pylon-3.2.0-x86/
pylon-3.2.0-ARM/          pylon-3.2.1-x86/
support@support-VirtualBox:~$ cd Desktop/pylon-3.2.0-x86/
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86$ ls
doc  INSTALL  INSTALL~  pylon3  pylonSDK-x86.tar.gz  README  Samples
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86$ source ./pylon3/bin/pylon-
setup-env.sh pylon3
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86$ cd Samples/Grab
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86/Samples/Grab$ make
g++ -I/home/support/Desktop/pylon-3.2.0-x86/pylon3/genicam/library/CPP/include -
I/home/support/Desktop/pylon-3.2.0-x86/pylon3/include -DUSE_GIGE -c -o Grab.o G
rab.cpp
g++ -L/home/support/Desktop/pylon-3.2.0-x86/pylon3/lib -L/home/support/Desktop/p
ylon-3.2.0-x86/pylon3/genicam/bin/Linux32_i86 -L/home/support/Desktop/pylon-3.2.
0-x86/pylon3/genicam/bin/Linux32_i86/GenApi/Generic -Wl,-E -o Grab Grab.o -lpylo
nbase -lGenApi_gcc40_v2_3 -lGCBASE_gcc40_v2_3 -lLog_gcc40_v2_3 -lMathParser_gcc4
0_v2_3 -lXerces-C_gcc40_v2_7 -llog4cpp_gcc40_v2_3
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86/Samples/Grab$
support@support-VirtualBox:~/Desktop/pylon-3.2.0-x86/Samples/Grab$ █

```

f. Now unpack the **pylon for Linux ARM** package and refer to the INSTALL text file in order to install pylon correctly (in our case the package was unpacked on the Desktop):

```
# cd ~/Desktop/pylon-3.2.0-ARM
```

Set the pylon environment variables:

```
# source ./pylon3/bin/pylon-setup-env.sh pylon3
```

For test purposes you may try to build one of the ARM samples now:

```
# cd Samples/Grab
```

```
# make
```



```

support@support-VirtualBox: ~/Desktop/pylon-3.2.0-ARM/Samples/Grab
support@support-VirtualBox:~$ cd Desktop/pylon-3.2.0-ARM/
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ ls
doc  INSTALL  pylon3  pylonSDK-ARM.tar.gz  README  Samples
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ source ./pylon3/bin/pylon-
setup-env.sh pylon3
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ cd Samples/Grab
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ make
g++ -I/home/support/Desktop/pylon-3.2.0-ARM/pylon3/genicam/library/CPP/include -
I/home/support/Desktop/pylon-3.2.0-ARM/pylon3/include -DUSE_GIGE -c -o Grab.o G
rab.cpp
g++ -L/home/support/Desktop/pylon-3.2.0-ARM/pylon3/lib -L/home/support/Desktop/p
ylon-3.2.0-ARM/pylon3/genicam/bin/Linux32_ARM -L/home/support/Desktop/pylon-3.2.
0-ARM/pylon3/genicam/bin/Linux32_ARM/GenApi/Generic -Wl,-E -o Grab Grab.o -lpylo
nbase -lGenApi_gcc43_v2_3 -lGCBBase_gcc43_v2_3 -lLog_gcc43_v2_3 -lMathParser_gcc4
3_v2_3 -lXerces-C_gcc43_v2_7 -llog4cpp_gcc43_v2_3
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
lib/libpylonbase.so when searching for -lpylonbase
/usr/bin/ld: cannot find -lpylonbase
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/libGenApi_gcc43_v2_3.so when searching for -lGenApi_gcc4
3_v2_3
/usr/bin/ld: cannot find -lGenApi_gcc43_v2_3
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/libGCBBase_gcc43_v2_3.so when searching for -lGCBBase_gcc4
3_v2_3
/usr/bin/ld: cannot find -lGCBBase_gcc43_v2_3
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/libLog_gcc43_v2_3.so when searching for -lLog_gcc43_v2_3
/usr/bin/ld: cannot find -lLog_gcc43_v2_3
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/libMathParser_gcc43_v2_3.so when searching for -lMathPar
ser_gcc43_v2_3
/usr/bin/ld: cannot find -lMathParser_gcc43_v2_3
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
lib/libXerces-C_gcc43_v2_7.so when searching for -lXerces-C_gcc43_v2_7
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/GenApi/Generic/libXerces-C_gcc43_v2_7.so when searching
for -lXerces-C_gcc43_v2_7
/usr/bin/ld: cannot find -lXerces-C_gcc43_v2_7
/usr/bin/ld: skipping incompatible /home/support/Desktop/pylon-3.2.0-ARM/pylon3/
genicam/bin/Linux32_ARM/liblog4cpp_gcc43_v2_3.so when searching for -llog4cpp_gc
c43_v2_3
/usr/bin/ld: cannot find -llog4cpp_gcc43_v2_3
collect2: ld returned 1 exit status
make: *** [Grab] Error 1
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$

```

In this case you will fail as the normal Linux compiler tries to build x86 binaries with pylon libraries built for ARM.

g. In order to be able to build ARM samples/application, which will be run on ARM machines eventually, you would need an ARM Cross Toolchain.

In this document we assume that you are using the following toolchain available on the Basler FTP server:



<ftp://Pylon4Linux-ro:h50UZgkl@ftp.baslerweb.com/ARM-Cross-Toolchain/arm-marvell-linux-gnueabi-vfp-fixed.tar.bz2>

h. Download and unpack the toolchain under the pylon ARM folder. In order to use the cross toolchain compiler instead of the Linux compiler, export the following environment variable:

```
# export CXX=~/.Desktop/pylon-3.2.0-ARM/arm-marvell-linux-gnueabi-vfp-fixed/bin/arm-marvell-linux-gnueabi-g++
```

Now move to the given sample again and remove the already available object file before compiling again.

```
# cd Samples/Grab
```

```
# rm *.o
```

Eventually you should successfully build a binary (Grab) that could be run on the MiraBox then:

```
# make
```

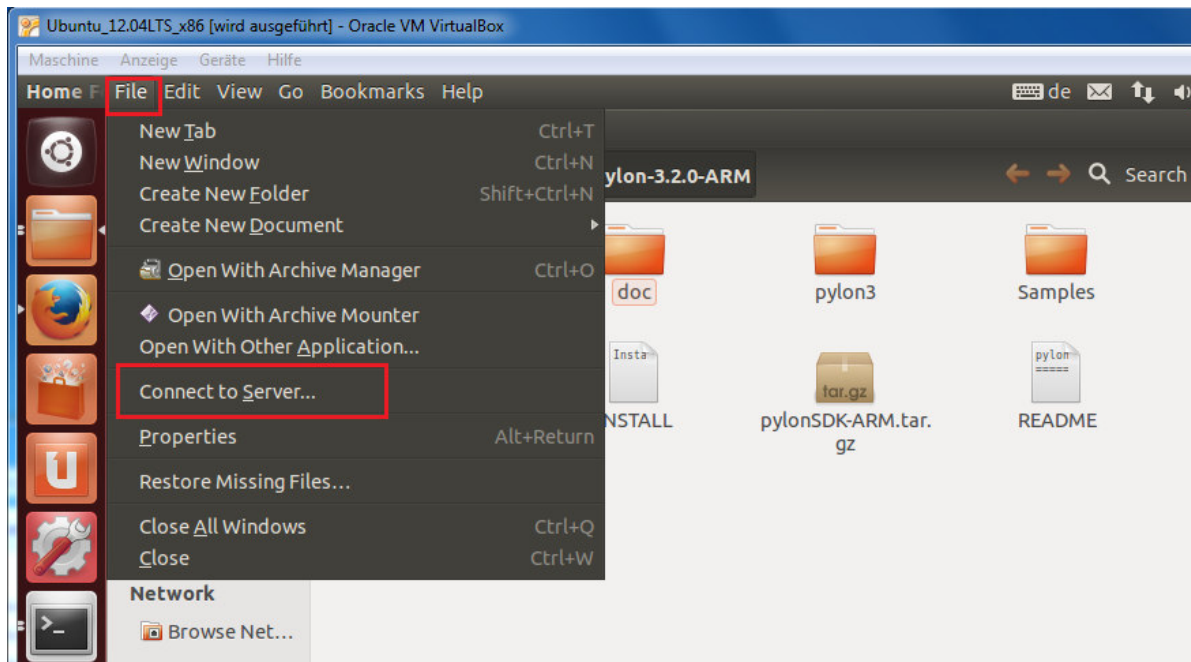
```
# ls
```

```

support@support-VirtualBox: ~/Desktop/pylon-3.2.0-ARM/Samples/Grab
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ cd ../../
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ ls
arm-marvell-linux-gnueabi-vfp-fixed  INSTALL  README
arm-marvell-linux-gnueabi-vfp-fixed.tar.bz2  pylon3  Samples
doc  pylonSDK-ARM.tar.gz
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ export CXX=~/Desktop/pylon-3.2.0-ARM/arm-marvell-linux-gnueabi-vfp-fixed/bin/arm-marvell-linux-gnueabi-g++
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ export CXX=~/Desktop/pylon-3.2.0-ARM/arm-marvell-linux-gnueabi-vfp-fixed/bin/arm-marvell-linux-gnueabi-g++
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ make
make: *** No targets specified and no makefile found. Stop.
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM$ cd Samples/Grab
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ ls
Grab.cpp  Grab.o  Makefile
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ rm *.o
rm: cannot remove `*.o': No such file or directory
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ rm *.o
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ make
/home/support/Desktop/pylon-3.2.0-ARM/arm-marvell-linux-gnueabi-vfp-fixed/bin/arm-marvell-linux-gnueabi-g++ -I/home/support/Desktop/pylon-3.2.0-ARM/pylon3/genicam/library/CPP/include -I/home/support/Desktop/pylon-3.2.0-ARM/pylon3/include -DUSE_GIGE -c -o Grab.o Grab.cpp
In file included from /home/support/Desktop/pylon-3.2.0-ARM/pylon3/include/pylon/PylonIncludes.h:48,
                 from Grab.cpp:17:
/home/support/Desktop/pylon-3.2.0-ARM/pylon3/include/pylon/PylonBase.h:62: warning: 'cdecl' attribute directive ignored
/home/support/Desktop/pylon-3.2.0-ARM/pylon3/include/pylon/PylonBase.h:71: warning: 'cdecl' attribute directive ignored
In file included from /home/support/Desktop/pylon-3.2.0-ARM/pylon3/include/pylon/PylonIncludes.h:48,
                 from Grab.cpp:17:
/home/support/Desktop/pylon-3.2.0-ARM/pylon3/include/pylon/PylonBase.h:78: warning: 'cdecl' attribute directive ignored
/home/support/Desktop/pylon-3.2.0-ARM/arm-marvell-linux-gnueabi-vfp-fixed/bin/arm-marvell-linux-gnueabi-g++ -L/home/support/Desktop/pylon-3.2.0-ARM/pylon3/lib -L/home/support/Desktop/pylon-3.2.0-ARM/pylon3/genicam/bin/Linux32_ARM -L/home/support/Desktop/pylon-3.2.0-ARM/pylon3/genicam/bin/Linux32_ARM/GenApi/Generic -Wl,-E -o Grab Grab.o -lpylonbase -lGenApi_gcc43_v2_3 -lGCBASE_gcc43_v2_3 -lLog_gcc43_v2_3 -lMathParser_gcc43_v2_3 -lXerces-C_gcc43_v2_7 -llog4cpp_gcc43_v2_3
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ ls
Grab  Grab.cpp  Grab.o  Makefile

```

i. In this document we are going to use the SSH server in order to copy the pylon for Linux ARM package and the Grab binary file to the MiraBox:

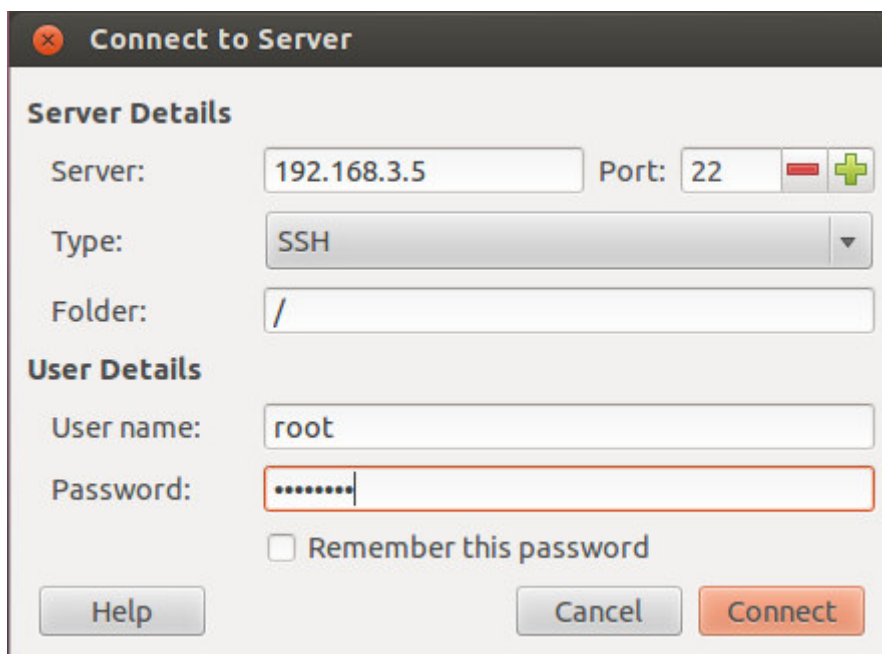


As a Server we are going to use the IP address of eth0, which we preconfigured already.

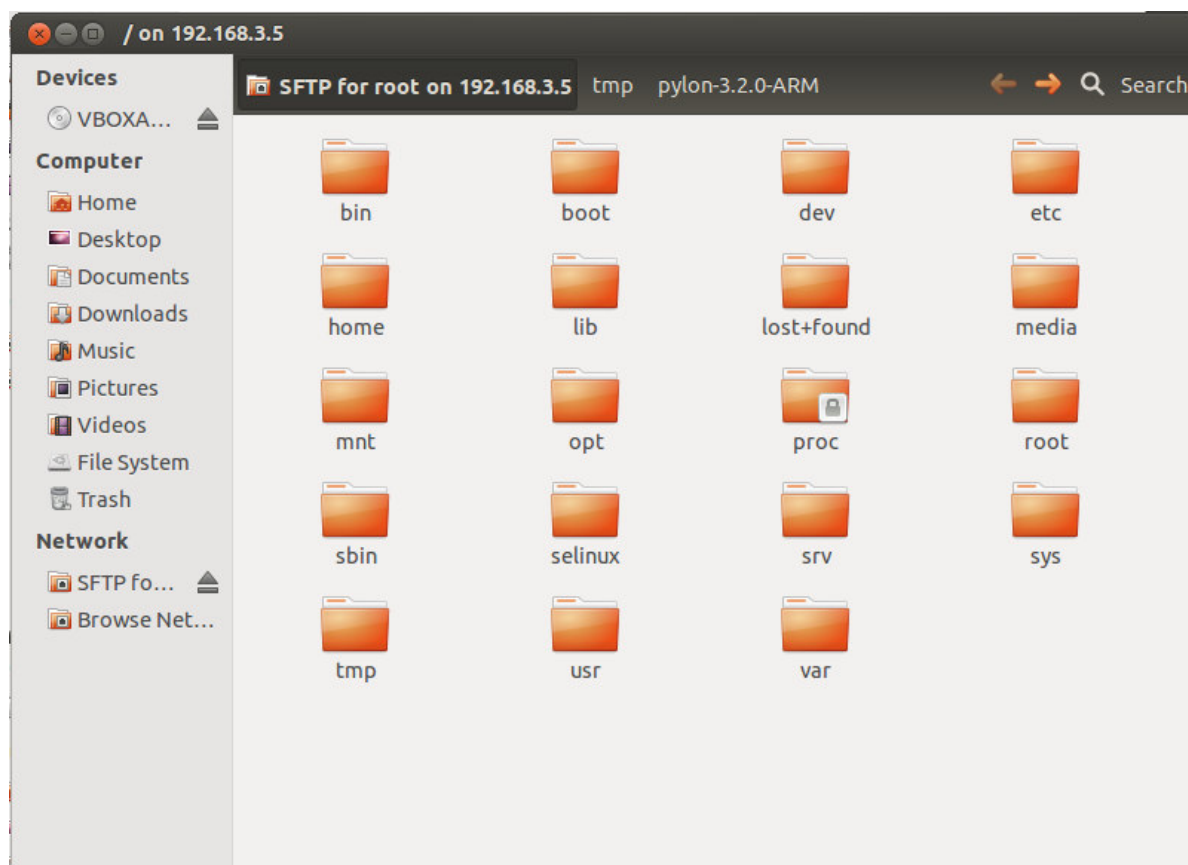
As a Type we are going to use SSH and the User Details are as follows:

User Name: root

Password: nosoup4u

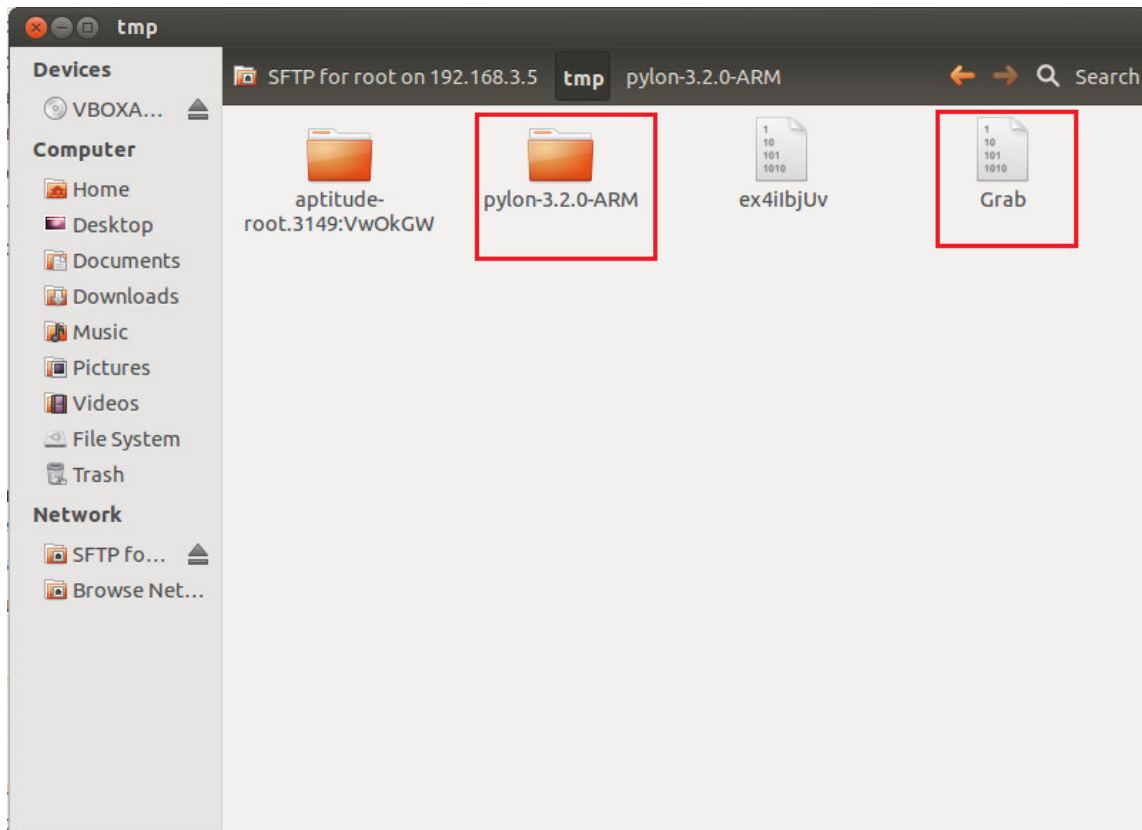


Click Connect in order to connect with the MiraBox:



- j. Go to the /tmp folder and copy the pylon-3.2.0-ARM package and the binary Grab file from your Linux machine here:





k. Connect to the MiraBox from the Linux terminal by typing in:

```
# ssh root@192.168.3.5
```

password: nosoup4u

Change the directory and go to /tmp:

```
# cd ../tmp
```

Run the Grab binary in order to check if it runs fine:

```
# ./Grab
```

As there is no camera connected to eth1 yet, you should get a notification that no device was available:

```
support@support-VirtualBox: ~/Desktop/pylon-3.2.0-ARM/Samples/Grab
support@support-VirtualBox:~/Desktop/pylon-3.2.0-ARM/Samples/Grab$ ssh root@192.168.3.5
root@192.168.3.5's password:
Linux mirabox-debian 2.6.35.9 #12 Thu Aug 23 22:13:28 EDT 2012 armv7l

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jan 29 08:36:04 2014 from 192.168.3.2
root@mirabox-debian:~# cd ..
root@mirabox-debian:/# ls
bin    dev    home  lost+found  mnt  proc  sbin    srv  tmp  var
boot  etc    lib   media      opt  root  selinux sys  usr
root@mirabox-debian:/# cd tmp/
root@mirabox-debian:/tmp# ls
aptitude-root.3149:VwOkGW  ex4iIbjUv  Grab  pylon-3.2.0-ARM
root@mirabox-debian:/tmp# source ./pylon-3.2.0-ARM/pylon3/bin/pylon-setup-env.sh
pylon-3.2.0-ARM/pylon3/
root@mirabox-debian:/tmp# ./Grab
An exception occurred.
No device is available or no device contains the provided device info properties
.

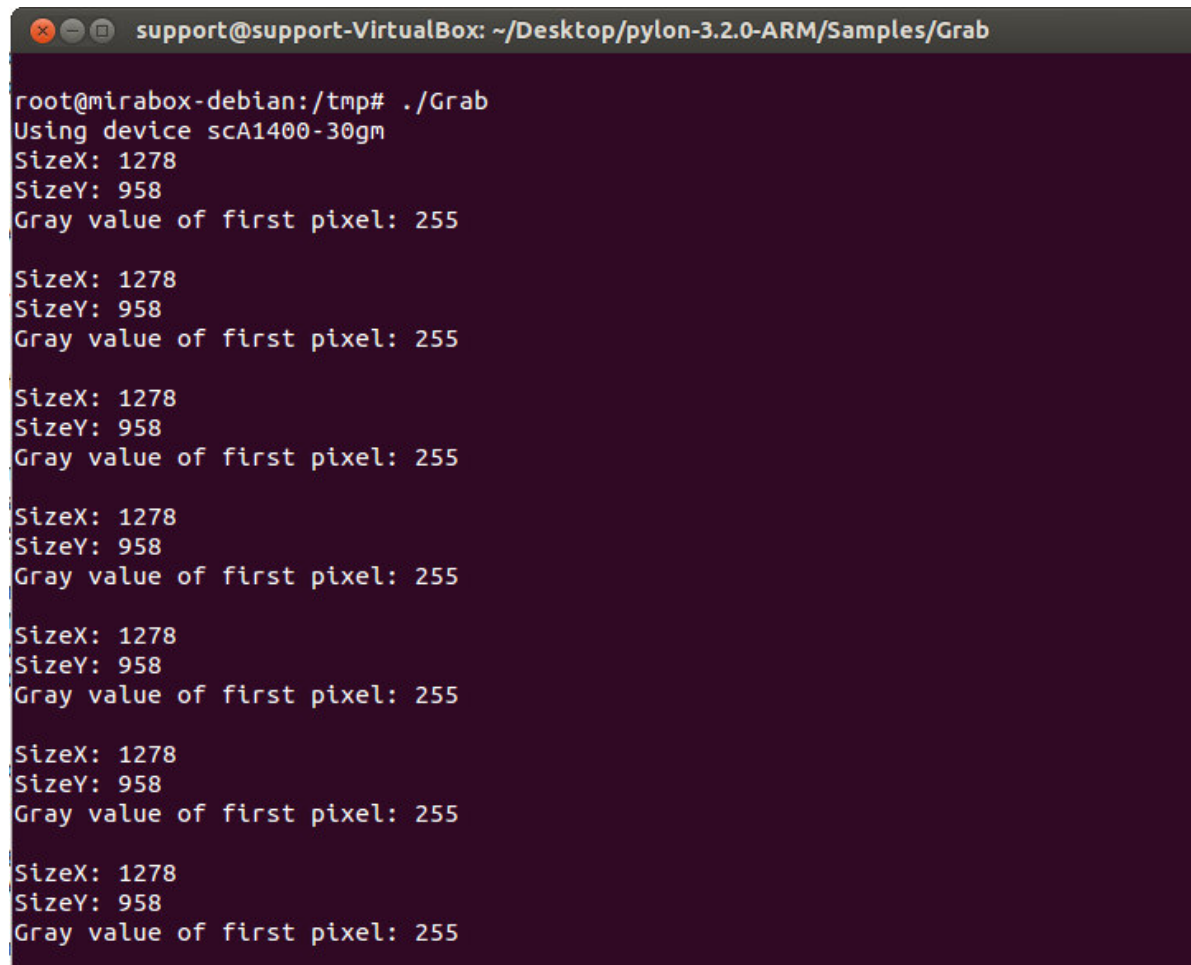
Press Enter to exit.
root@mirabox-debian:/tmp#
```

l. In order to test the Grab binary in combination with a GigE Vision camera, connect your Basler camera to the second Ethernet port (eth1) of the MiraBox. This document assumes that you have already preconfigured the camera within the subnet of eth1, e.g. IP: 192.168.4.123, Subnet Mask: 255.255.255.0.

You can do that either on your Windows or Linux machine by using the pylon IP Configurator tool.

m. After the camera was connected you may run the Grab binary again:

```
# ./Grab
```

A terminal window titled 'support@support-VirtualBox: ~/Desktop/pylon-3.2.0-ARM/Samples/Grab' shows the execution of a script. The prompt is 'root@mirabox-debian:/tmp# ./Grab'. The output consists of several identical blocks of text: 'Using device scA1400-30gm', 'SizeX: 1278', 'SizeY: 958', and 'Gray value of first pixel: 255'. There are five such blocks in total, indicating a loop or multiple acquisitions.

```
support@support-VirtualBox: ~/Desktop/pylon-3.2.0-ARM/Samples/Grab
root@mirabox-debian:/tmp# ./Grab
Using device scA1400-30gm
SizeX: 1278
SizeY: 958
Gray value of first pixel: 255

SizeX: 1278
SizeY: 958
Gray value of first pixel: 255

SizeX: 1278
SizeY: 958
Gray value of first pixel: 255

SizeX: 1278
SizeY: 958
Gray value of first pixel: 255

SizeX: 1278
SizeY: 958
Gray value of first pixel: 255
```

n. Pay attention that if it takes too long (40-50 seconds or longer) for the camera to get opened and the image acquisition started, you may refer to the INSTALL text file within the pylon for Linux package (chapter Environment Variables) in order to take account of the following:

The GENICAM\_CACHE\_V2\_3 environment variable must point to a folder where the application can write, e.g.,

```
# mkdir -p $HOME/genicam_xml_cache
```

```
# export GENICAM_CACHE_V2_3=$HOME/genicam_xml_cache
```

The directory to which the GENICAM\_CACHE\_V2\_3 variable is pointing must exist.





## Revision History

Document Number	Date	Changes
AWxxxxxxYY	31 January 2014	Initial release version of this document.

---

**Basler AG**  
**Germany, Headquarters**

Tel. +49 4102 463 500  
Fax +49 4102 463 599

[sales.europe@baslerweb.com](mailto:sales.europe@baslerweb.com)

[www.baslerweb.com](http://www.baslerweb.com)

**USA**

Tel. +1 610 280 0171  
Fax +1 610 280 7608

[sales.usa@baslerweb.com](mailto:sales.usa@baslerweb.com)

**Asia**

Tel. +65 6425 0472  
Fax +65 6425 0473

[sales.asia@baslerweb.com](mailto:sales.asia@baslerweb.com)