



Interface Raspberry-pi Board with AppCOE on Windows

Copyright (c) 2018, All Rights Reserved

MapuSoft Technologies

1301 Azalea Road

Mobile, AL 36693

support@mapusoft.com

info@mapusoft.com

www.mapusoft.com

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Copyright

The information contained herein is subject to change without notice. The materials located on the Mapusoft. ("MapuSoft") web site are protected by copyright, trademark and other forms of proprietary rights and are owned or controlled by MapuSoft or the party credited as the provider of the information.

MapuSoft retains all copyrights and other property rights in all text, graphic images, and software owned by MapuSoft and hereby authorizes you to electronically copy documents published herein solely for the purpose of reviewing the information.

You may not alter any files in this document for advertisement, or print the information contained herein, without prior written permission from MapuSoft.

MapuSoft assumes no responsibility for errors or omissions in this publication or other documents which are referenced by or linked to this publication. This publication could include technical or other inaccuracies, and not all products or services referenced herein are available in all areas. MapuSoft assumes no responsibility to you or any third party for the consequences of an error or omissions. The information on this web site is periodically updated and may change without notice.

This product includes the software with the following trademarks:

MS-DOS is a trademark of Microsoft Corporation.

Vivado® Design Suite is a trademark of Xilinx

ARM® is a registered trademark of ARM

UNIX is a trademark of X/Open.

IBM PC is a trademark of International Business Machines, Inc.

Nucleus PLUS and Nucleus NET are registered trademarks of Mentor Graphics Corporation.

ThreadX are registered trademarks of Mentor Graphics Corporation.

Linux is a registered trademark of Linus Torvald.

VxWorks and pSOS are registered trademarks of Wind River Systems.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Table of Contents

Chapter 1.About this Guide	8
Audience	9
How to Use This Manual	9
Document Conventions	10
Requesting Support	11
Registering a New Account	11
Submitting a Ticket.....	11
Live Support Offline	12
Documentation Feedback.....	12
Chapter 2.Introduction to AppCOE.....	13
About AppCOE.....	14
Installing AppCOE	14
Installing AppCOE License key.....	14
Launching AppCOE Workspace	15
Chapter 3.Introduction to Raspberry Package	17
AppCOE Installation Directory	18
Extracting the Raspberry Package.....	19
Details of Raspberry Package Subfolders.....	20
Launching AppCOE Raspberry Workspace.....	23
Chapter 4.Installing the ARM toolchain on windows.....	26
Installing ARM Tool Chain	27
Chapter 5.Configuring Raspberry-Pi	28
Essential Hardware Needed for Raspberry-Pi	29
Starting Raspberry-Pi.....	29
Configuring SSH server on Raspberry-Pi.....	30
Chapter 6.Configuring AppCOE for Remote Connection to Raspberry .	34
Connecting using Remote System Explorer from AppCOE.....	35
Chapter 7.Building Demo Projects on AppCOE	46
Building Demo Projects on AppCOE Host.....	47
Chapter 8.Running the Demo Projects on Raspberry-Pi	50
Setting up the Run Configuration	51
Chapter 9.Debugging the Demo Projects on Raspberry-Pi	54
Set up the Debug Configuration in AppCOE.....	55
Start Remote Debugging.....	60
10. Creating ARM Cross OS Target Application on AppCOE	63
11. Steps to Install Plugins on AppCOE	78

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Installing GNU ARM C/C++ Development Tools	79
Installing Remote Debugging Capabilities	82
12. Importing the ARM Cross OS Target Template Application on AppCOE	85
Reference.....	89
Revision History	90

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

List of Figures

Figure-1: Create a Support Ticket from AppCOE	11
Figure-2: Launching AppCOE	15
Figure-3:AppCOE Splash Screen.....	15
Figure-4: AppCOE Default Workspace.....	16
Figure-5: AppCOE installation directory	18
Figure-6: raspberry_pkg folder after extracting.	19
Figure-7: Raspberry package sub folders.....	20
Figure-8: Plugins inside the ARM_development_tools folder	20
Figure-9: ARM toolchain “raspberry-gcc4.9.2-r4.exe”	21
Figure-10: Include folders	21
Figure-11: lib folders	22
Figure-12: Workspace folder	22
Figure-13: AppCOE Raspberry Workspace for Windows Host.....	23
Figure-14: AppCOE Raspberry demos for Windows	23
Figure-15:About AppCOE	24
Figure-16:AppCOE Version	24
Figure-17: Installed Plugins on AppCOE	25
Figure-18: ARM toolchain executable file.....	27
Figure-19: Installing Raspberry-pi ARM Tool Chain	27
Figure-20:Raspberry Pi Software Configuration Tool.....	30
Figure-21: Set hostname of Raspberry Pi in Host machine	31
Figure-22: ping raspberry-pi from Host Machine	32
Figure-23: Check the Ethernet Connection by Used PuTTY Tool	32
Figure-24: Accept key for PuTTY Security Alert.....	33
Figure-25: Raspberry Login Credential from Putty Tool.....	33
Figure-26: Navigate to other Perspective.....	35
Figure-27: Select Remote System Explorer	35
Figure-28: Selecting the Connection from Remote System Explorer.....	36
Figure-29: Selecting the Linux for Remote System type.....	36
Figure-30: Define Linux System Connection information	37
Figure-31: Selecting ssh Files Configuration.....	38
Figure-32: Selecting processes.shell.linux Configuration	39
Figure-33: Selecting ssh.shells Configuration	40
Figure-34: Selecting ssh.terminals Configuration	41
Figure-35: New Connection show on Remote System Explorer	42
Figure-36: Selecting Connect	42
Figure-37: Connection status.....	44
Figure-38: Expand SFTP Files.....	44
Figure-39:Setting Username & password	43
Figure-40:Cancel the secure storage password	43
Figure-41: Check the File system	44
Figure-42:Launch Terminal	45
Figure-43: Terminal launched on Terminal pane	45
Figure-44: Raspberry demos with binaries	47
Figure-45: Build the demo	48
Figure-46: Building project	48
Figure-47: Clean the demo project	49
Figure-48: Navigate to Run Configuration	51
Figure-49: Remote application in Run configuration	51
Figure-50: Create a new launch configuration	52

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Figure-51: Select Run option from Display in favourites Menu	53
Figure-52: Console output	53
Figure-53: Navigate to Debug Configuration	55
Figure-54: Debug launch configuration	55
Figure-56: Create a .gdbinit text file	57
Figure-57: Standard Remote Create Process Launcher	57
Figure-58: Select the Remote gdb/mi Debugger	58
Figure-59: Default settings select Stop on startup at: main	58
Figure-60: Default settings for Gdbserver settings tab	59
Figure-61: Checking gdbserver installed on TargetOS	59
Figure-62: Confirm perspective switch	60
Figure-63: Remote Debugging Stop on main	60
Figure-64: Console output	61
Figure-65: Terminate Remote Debugging	61
Figure-66: Disconnect the Connection	62
Figure-67: Select a wizard	64
Figure-68: Creating the ARM Cross Target Application Project	65
Figure-69: Project Explorer view	66
Figure-70: Import file	66
Figure-71: Refresh project to view the imported files	67
Figure-72: Changing command invocation command for project	68
Figure-73: Path variable setting	68
Figure-74: Target Processor Settings	69
Figure-75: Changing Command for ARM Windows GCC Linker	69
Figure-76: Setting Preprocessor Symbols	70
Figure-77: Added Include paths	70
Figure-78: Added directory paths	71
Figure-79: Default Miscellaneous settings for ARM Windows GCC Compiler	71
Figure-80: Libraries Values	71
Figure-81: Libraries path	72
Figure-82: Process mode library	72
Figure-83: Non Process mode library	72
Figure-84: Profiler mode library	73
Figure-85: Libraries values & Search path	73
Figure-86: Additional Tools	74
Figure-87: Changing Command in ARM Windows GNU Create Flash Image	74
Figure-88: Changing Command in ARM Windows GNU Create Listing	74
Figure-89: Changing Command in ARM Windows GNU Print Size	75
Figure-90: Enable the GNU Elf parser option	75
Figure-91: Enable the GNU Compiler, Linker, Assembler Error parser	76
Figure-92: Check included directories for arm_project for Windows Host	76
Figure-93: Building vxworks_project Application Project	76
Figure-94: vxworks_project.elf Binary File	77
Figure-95: Install New Software	79
Figure-96: Add Repository from Local Sites	80
Figure-97: Selection GNU ARM C/C++ Development Support Plugins	80
Figure-98: Install Details	81
Figure-99: Accept the License	81
Figure-100: Add Repository into Available Software Window	82
Figure-101: Path of remote login Plugins	82
Figure-102: Selecting Mobile and Device Development Software Tool	83
Figure-103: Install Details	83

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Figure-104: Accept a Licenses.....	84
Figure-105: Installing Software	84
Figure-106: Restart Now	84
Figure-107: Import into C/C++ Project Pane.....	86
Figure-108: Import into C/C++ Project Pane.....	86
Figure-109: Import template project from Extracted source.	87
Figure-110: Copy template project into workspace	87
Figure-111: Building the ARM Cross Target Application Template Project	88

Chapter 1.About this Guide

This chapter contains the following topics:

- Objectives
- Audience
- How to Use This Manual
- Document Conventions**Error! Reference source not found.**Requesting Support
- Documentation Feedback

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Objectives

This manual describes the steps to use the AppCOE IDE to run & debug the RTOS application on the Raspberry-pi Target Board (ARM1176 jzf-s) from Windows Host Machine.

This manual assumes that the user having a Raspberry-pi target board running with Raspbian operating system (2017-04-10-raspbian-jessie). If you're using any other different Linux distribution on raspberry-pi target board, the configuration and other details may vary.

Audience

This manual is designed for anyone who wants to create and develop the embedded applications, projects, and also guide users to run and debug their applications on the Raspberry-pi (ARM1176jzf-s) Target Board.

This manual is intended for the following audiences:

- Customers with technical knowledge and experience with the Embedded Systems

How to Use This Manual

The organization of this document is as described below:

Using these documents user can do

- Launching Raspberry workspace
- Configuring Raspberry-pi
- Creating ARM Cross Target Application Project
- Compile a ARM Cross Target Application linking with Raspberry-pi-AppCOE Library
- Communication between AppCOE IDE and Raspberry-pi Target Board through SSH
- Deployed & run the application into Raspberry-pi Target Board by using AppCOE IDE through Remote system Explore
- Debug the application using AppCOE IDE through Remote system Explore

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Document Conventions

Table 1_1 defines the notice icons used in this manual.

Table 1_1: Notice Icons



Icon	Meaning	Description
	Informational note	Indicates important features or icons.
	Caution	Indicates a situation that might result in loss of data or software damage.

Table 1_2 : defines the Text and Syntax conventions used in this manual.

Table 1_2: Text and Syntax Conventions

Convention	Description
Bookman Old Style	Identifies Program listings and Program examples.
<i>Italic text like this</i>	Introduces important new terms. <ul style="list-style-type: none">• Identifies book names• Identifies Internet draft titles.
BOOKMAN OLD STYLE, ALL CAPS	Identifies File names.
Bookman Old Style, Bold	Identifies Interactive Command lines

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Requesting Support

Technical support is available through the MapuSoft Technologies Support Centre. If you are a customer with an active MapuSoft support contract, or covered under warranty, and need post sales technical support, you can access our tools and resources online or open a ticket at <http://www.mapusoft.com/support>.

Registering a New Account

To register:

- From <http://www.mapusoft.com/> main page, select **Support**.
- Select **Register** and enter the required details.
- After furnishing all your details, click **Submit**.

Submitting a Ticket

1. To submit a ticket:
 1. From <http://www.mapusoft.com/> main page, select **Support > Submit a Ticket**
 2. Select a department according to your problem, and click **Next**.
 3. Fill in your details and provide detailed information of your problem.
 4. Click **Submit**.

MapuSoft Support personnel will get back to you within 48 hours with a valid response.

2. To submit a ticket from AppCOE main menu, Select **Help>Create a Support Ticket** as shown inFigure-1

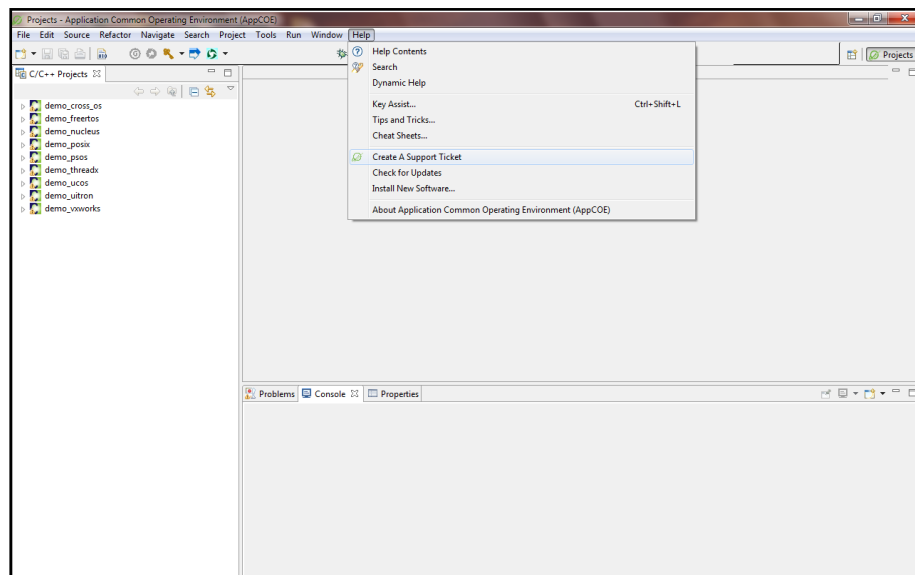


Figure-1: Create a Support Ticket from AppCOE

1. Using the Existing Email and Password for login into Mapusoft Support Suite.
2. Select the department according to your problem, and click **Next**.
3. Fill in your details and provide detailed information of your problem.
4. Click **Submit**.

MapuSoft Support personnel will get back to you within 48 hours with a valid response.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Live Support Offline

MapuSoft Technologies also provides technical support through Live Support offline.

To contact live support offline:

1. From <http://www.mapusoft.com/> main page, select **Support>Live Support Offline**.
2. Enter your personal details in the required fields. Enter a message about your technical query. One of our support personnel will get back to you as soon as possible.
3. Click **Send**.

You can reach us at our toll free number: 1-877-627-8763 for any urgent assistance.

Documentation Feedback

Send Feedback on Documentation: <http://www.mapusoft.com/support/index.php/>

Chapter 2.Introduction to AppCOE

This chapter contains the following topics:

About AppCOE

Installing AppCOE

Installing AppCOE License key

Launching AppCOE default Workspace

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

About AppCOE

AppCOE is an Eclipse based IDE. AppCOE integrates software interoperability & reuse tools like OS Changer and OS Abstractor with Eclipse's CDT to offer an IDE for developing and porting embedded applications on many host/target platforms.

With AppCOE you can perform the following actions:

- Creation of C and C++ AppCOE projects
- Porting of legacy applications
- Host development with simulation for many OS applications
- Converting Ada source code to C/C++ code
- Platform and Application profiling
- Automatic configuration of any OS Changer and OS Abstractor APIs needed by the application
- Custom configuration of OS & OS Abstractor resources needed by the application
- Custom configuration of OS Abstractor for single or multi-application development (Process Feature support)
- Optimized source code generation
- Full Source Library Package generation

[Contact MapuSoft](#) to receive the components needed for using AppCOE. The steps for using AppCOE are comprehensively described in the following pages.

Installing AppCOE

- Install AppCOE via the RTOS SIMULATOR CD given by MapuSoft Technologies.
- Please contact MapuSoft if you need help in getting the components mentioned below.
AppCOE License Key [-Request One Here](#)
- Install the Latest **AppCOE_1.6.1** on your **Windows/Linux host machine** (if not installed earlier). For more reference you refer to [Quick Installation Guide to AppCOE](#) document.

Installing AppCOE License key

Install the **license key** into AppCOE, please refer [Quick Installation Guide to AppCOE](#) document

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Launching AppCOE Workspace

- Run the **AppCOE** executable in **admin** login privilege to install. Bring up AppCOE application from AppCOE <InstallDir>, or right click the AppCOE executable shortcut from desktop and click **Run as administrator**.

Note: Do not double-click the AppCOE to open it directly.

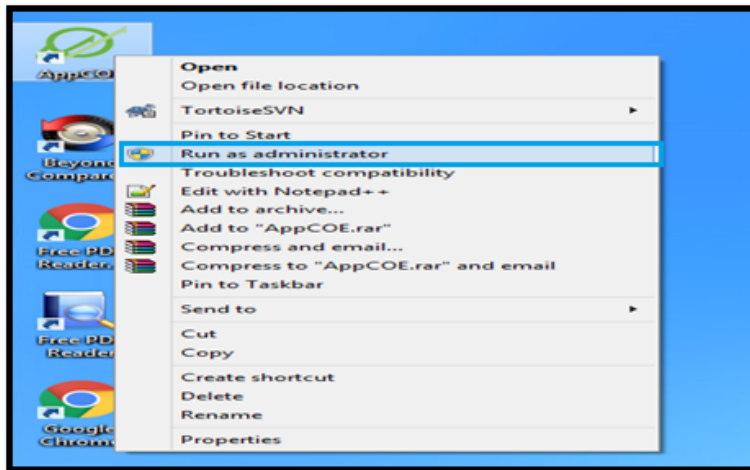


Figure-2: Launching AppCOE

- After the starting the AppCOE executable, splash screen will appear as shown in Figure-3



Figure-3: AppCOE Splash Screen

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- When AppCOE is opened, it will display the default workspace as shown in Figure-4.

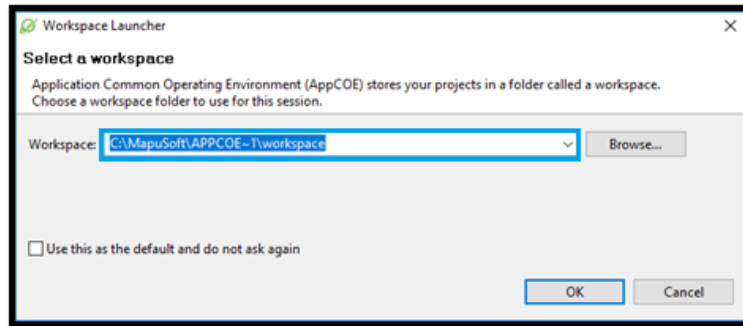


Figure-4: AppCOE Default Workspace

Chapter 3.Introduction to Raspberry Package

This chapter contains the following topics:

About AppCOE

Extracting the Raspberry Package

Details of Raspberry Package Subfolders

Launching AppCOE Raspberry Workspace

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

AppCOE Installation Directory

After installing AppCOE 1.6.1 the installation directory will be as shown in Figure-5.

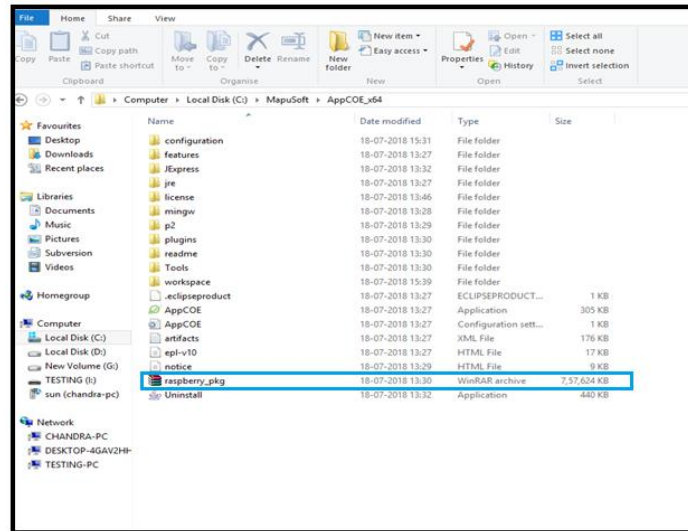


Figure-5: AppCOE installation directory

Inside the AppCOE installation directory, you will find raspberry package as a compressed rar file “**raspberry_pkg.rar**”. Hope you have installed the utility software “WinRAR” on your existing windows.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Extracting the Raspberry Package

Extract the “**raspberrry_pkg.rar**” compressed rar file inside the same AppCOE installation directory using the utility software “WinRAR”. While extracting the package through WinRAR, go for option “**extract here**”. After extraction, you will get the “**raspberrry_pkg**” folder as shown in Figure-6.

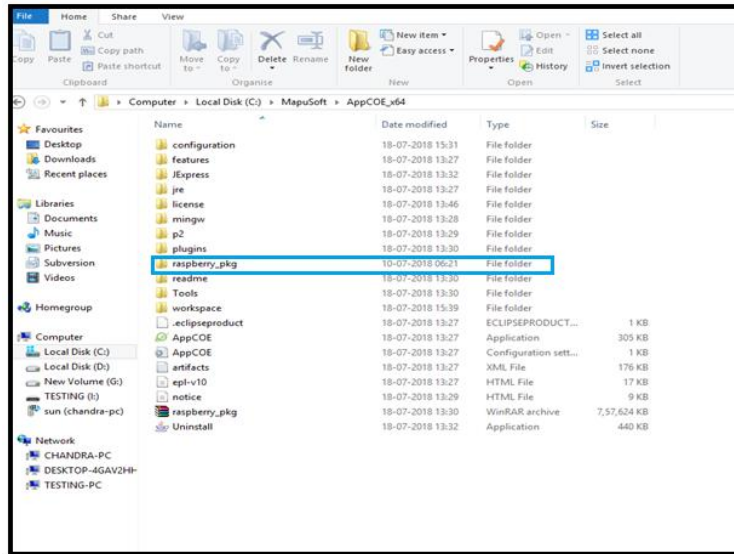


Figure-6: raspberrry_pkg folder after extracting.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Details of Raspberry Package Subfolders

Go into the raspberry package folder as shown in Figure-7.

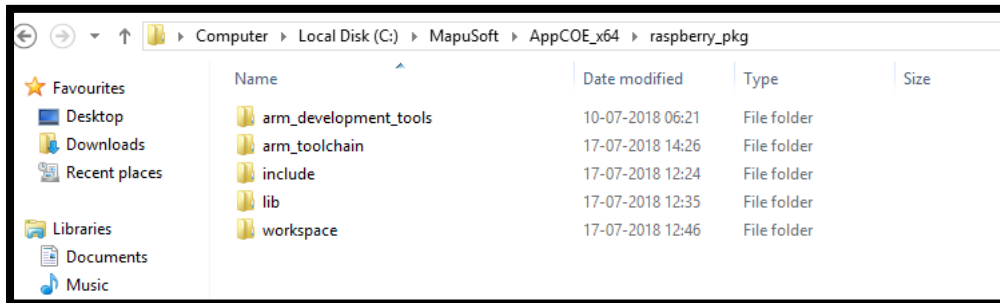


Figure-7: Raspberry package sub folders

- **arm_development_tools:** The GNU ARM Plug-ins for eclipse is provided in this folder. The plug-in inside this folder as shown in Figure-8.

Note: Already this plug-in is installed in AppCOE. No need to install this plug-in again.

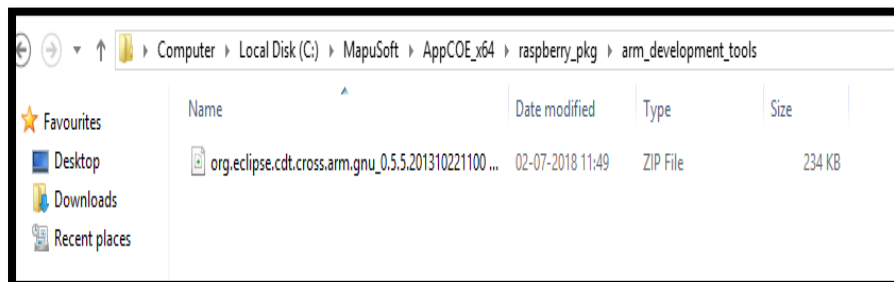


Figure-8: Plugins inside the arm_development_tools folder

- **arm_toolchain:** The arm_toolchain folder contain the “raspberry-gcc4.9.2-r4.exe”.Install this arm toolchain in the default directory as (C:\>).The toolchain inside this folder as shown in Figure-9.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

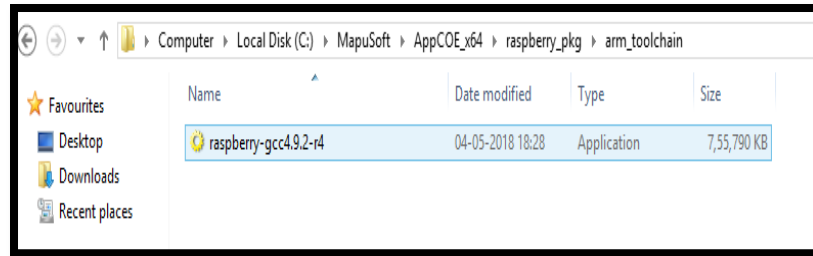


Figure-9: ARM toolchain “raspberry-gcc4.9.2-r4.exe”

- **include (folder):** This folder contain all the header files of all interfaces for creating the application on AppCOE using the Mapusoft api's to run on raspberry target. The different include folders are shown in Figure-10.

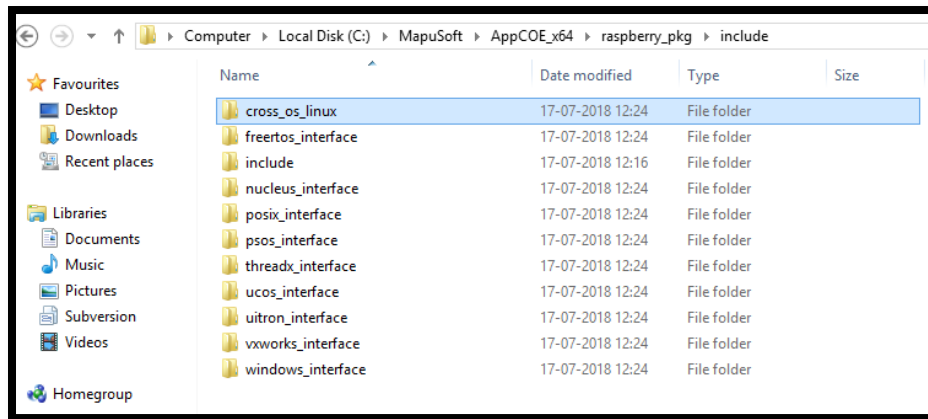


Figure-10: Include folders

- **lib (folder):** This folder contain all the library files of all interfaces for building your project. You can link the libraries depending upon application to be developed in process, non-process and non-process along with profiler enabled. The different lib folders are shown in Figure-11.

Note: profiler libraries are in non-process mode.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

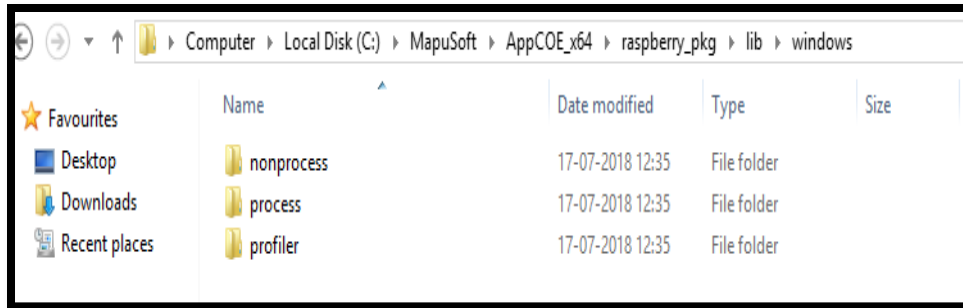


Figure-11: lib folders

- **Workspace (folder):** This folder contains raspberry workspace along with demo projects.

Note: We are providing demo applications to be build, run and debug on raspberry board.

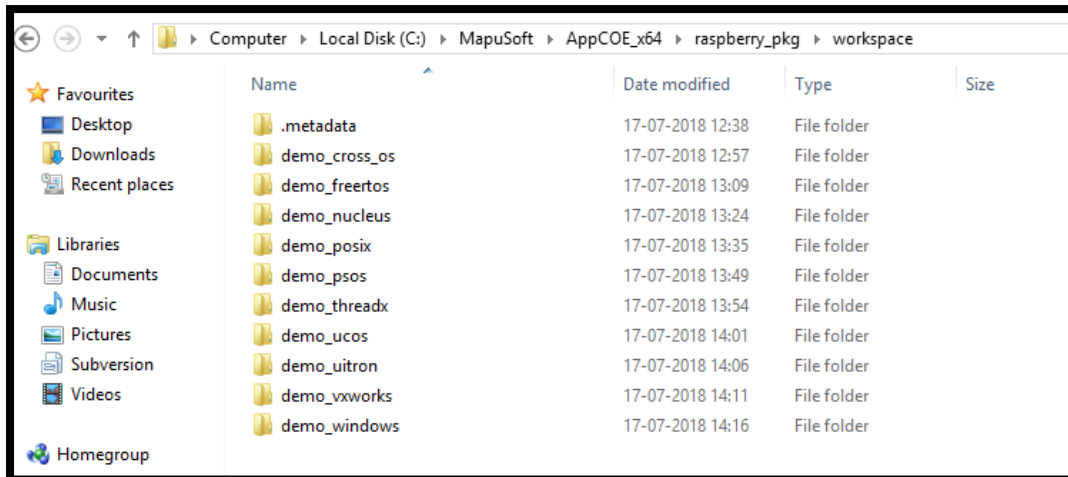


Figure-12: Workspace folder

Launching AppCOE Raspberry Workspace

Note: Do not double-click the AppCOE to open it directly.



C/C++ - Application Common Operating Environment (AppCOE)

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer

- demo_cross_os
- demo_fractals
- demo_musicles
- demo_posix
- demo_gauss
- demo_threadx
- demo_ucos
- demo_ultron
- demo_vworks
- demo_windows

An outline is not available.

Problems Tasks Console Properties

No consoles to display at this time.

0 demo selected

Note: A specific demo will run depending on your license.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Raspberry package is installed in AppCOE 1.6.1 version. The AppCOE version can be checked as shown in Figure-15.

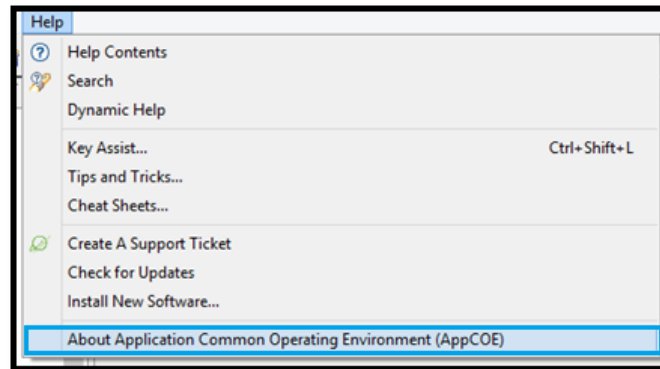


Figure-15: About AppCOE



Figure-16: AppCOE Version

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

GNU ARM C/C++ Development tool Plugins and Remote debugging tool Plugins embedded in AppCOE as shown in Figure-17.

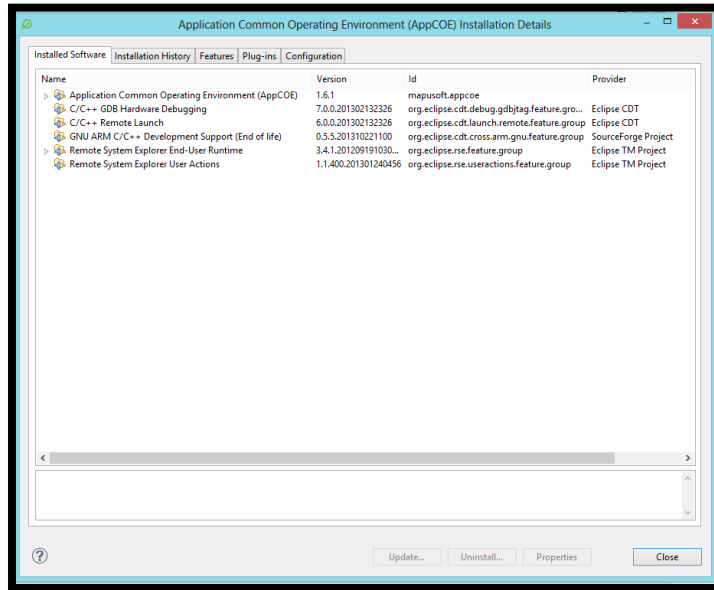


Figure-17: Installed Plugins on AppCOE

Chapter 4.Installing the ARM toolchain on windows

This chapter contains the following topics:

About AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Installing ARM Tool Chain

Go inside the arm_toolchain folder inside your raspberry_pkg as shown in Figure-18. Install raspberry-gcc4.6.3.exe inside the AppCOE installation directory as shown in Figure-19. While installing in AppCOE installation directory set the path as:

```
C:\MapuSoft\AppDataE_x64\raspberry_pkg\SysGCC\Raspberry
```

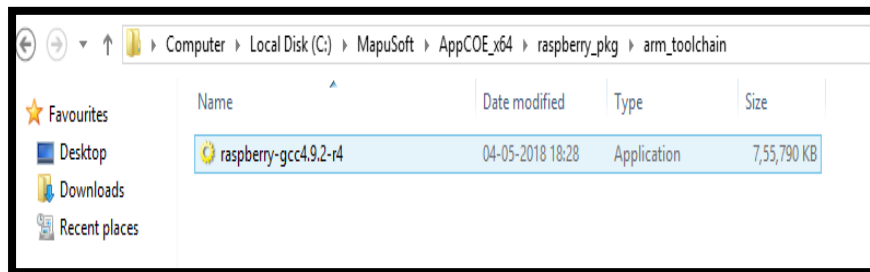


Figure-18: ARM toolchain executable file

Installing the ARM toolchain as shown below.

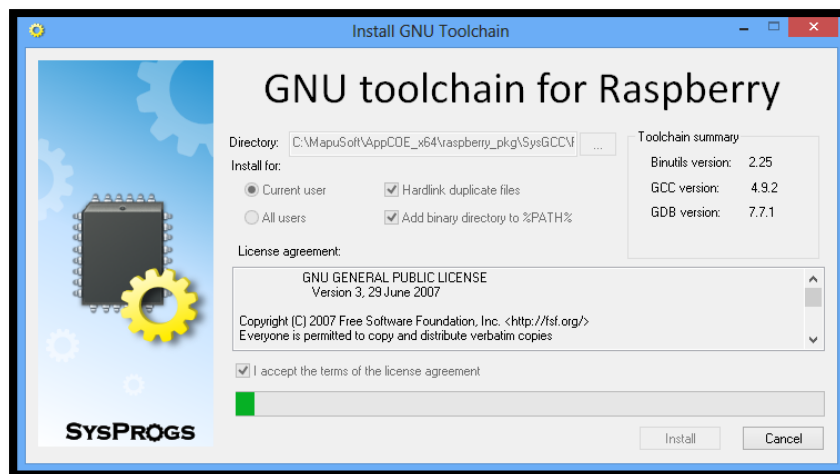


Figure-19: Installing Raspberry-pi ARM Tool Chain

Chapter 5. Configuring Raspberry-Pi

This chapter contains the following topics:

About AppCOE

Starting Raspberry-Pi

Configuring SSH Server on Raspberry-Pi

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Essential Hardware Needed for Raspberry-Pi

The following are essential hardware needed for Raspberry-Pi

- Raspberry Pi hardware board
- Prepared Operating System SD Card
- USB keyboard
- Display (with HDMI, DVI, Composite or SCART input)
- Power Supply
- Cables (Power cables, Cross over network Cable)

Highly suggested extras include:

- USB mouse
- Internet connectivity - a USB WiFi adaptor (Model A/B) or a LAN cable (Model B)
- Powered USB Hub
- Case

Starting Raspberry-Pi

1. Plug the preloaded SD Card into the Raspberry Pi target board.

Note: By Default, preloaded SD Card contains Raspbian OS (Raspbian GNU/Linux 8 (Jessie) – Linux Raspberry-pi 4.4.50-v7+) given by Mapusoft Technologies.

2. Plug the USB keyboard and mouse into the Raspberry Pi.
3. Plug the HDMI cable into the screen (TV) and other end plug into the Raspberry Pi.
4. Plug your extras into the Pi (USB WiFi, Ethernet cable, hard drive, etc) if necessary.
5. Plug the power source into the main socket.
6. With your screen on, plug the other end of the power source into the Pi.
7. The Pi should boot up and display messages on the screen.

Note: The RPI may take a long time to boot when powered-on for the first time, so be patient!

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Configuring SSH server on Raspberry-Pi

1. Once Pi got booted up, it will be shown “raspi-config” on first booting into Raspbian. To open the configuration tool after this, otherwise simply run the following from the terminal:

sudo raspi-config

2. The sudo is required because you will be changing files that you do not own as the pi user.
3. You should see a blue screen with options in a grey box in the centre as shown in Figure-20:

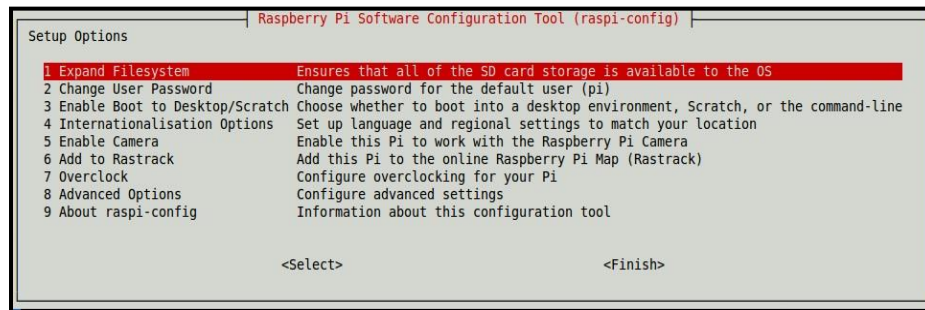


Figure-20: Raspberry Pi Software Configuration Tool

4. Go to Advanced Options> Enable the sshd server and then restart Raspberry-pi Board
5. After Pi restarted, assign the static IP address to Raspberry Pi either by editing “/etc/network/interfaces” opened with nano or gedit or Vi editor.

Example configuration:

```
auto lo
iface lo inet loopback
iface eth0 inet static
address 192.168.1.101
netmask 255.255.255.0
gateway 192.168.1.1
allow-hotplug wlan0
iface wlan0 inet manual

wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf

iface default inet dhcp
```

(OR)

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Editing */etc/dhcpd.conf* file likes below lines at end of the file

interface eth0

staticip_address=192.168.1.101/24

static routers=192.168.1.1

staticdomain_name_servers=192.168.1.1

If network is wireless, let's go and change **"wlan0"** instead of **"eth0"** from above configuration.

Note: Change 192.168.1.101 to the IP address you wish to use.

6. Add below lines in */etc/ssh/sshd_config* file to avoid Algorithm negotiation fail in SSH connectivity

KexAlgorithms diffie-hellman-group1-sha1,curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group14-sha1

Note: Add in single line without newline characters, otherwise it gives connectivity issue.

7. In windows add static IP address and hostname of Raspberry Pi into the C:\Windows\system32\drivers\etc\hosts file as shown in Figure-21 (for Linux case skip this step)

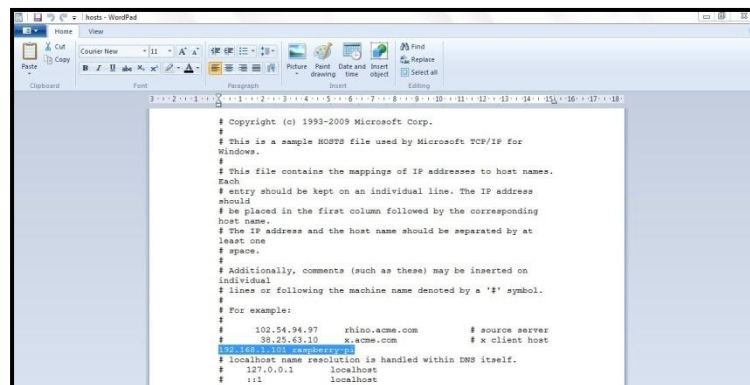
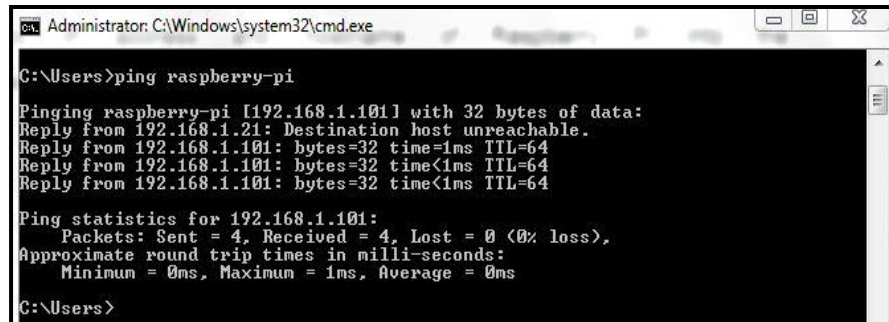


Figure-21: Set hostname of Raspberry Pi in Host machine

8. In windows press **Win+R**, Enter **"cmd"**, **"ping my-raspberry-pi"** to make sure "hosts" file was saved properly as shown in Figure-22 (for Linux machine just ping your raspberry pi hardware IP from the terminal)

INTERFACE RASPBERRY-PI BOARD WITH APPCOE



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

C:\Users>ping raspberry-pi

Pinging raspberry-pi [192.168.1.101] with 32 bytes of data:
Reply from 192.168.1.21: Destination host unreachable.
Reply from 192.168.1.101: bytes=32 time=1ms TTL=64
Reply from 192.168.1.101: bytes=32 time<1ms TTL=64
Reply from 192.168.1.101: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.1.101:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users>
```

Figure-22: ping raspberry-pi from Host Machine

9. Try logging into the Raspberry Pi using [PuTTY](#). If PuTTY doesn't work, there's no point in trying to configure Eclipse.

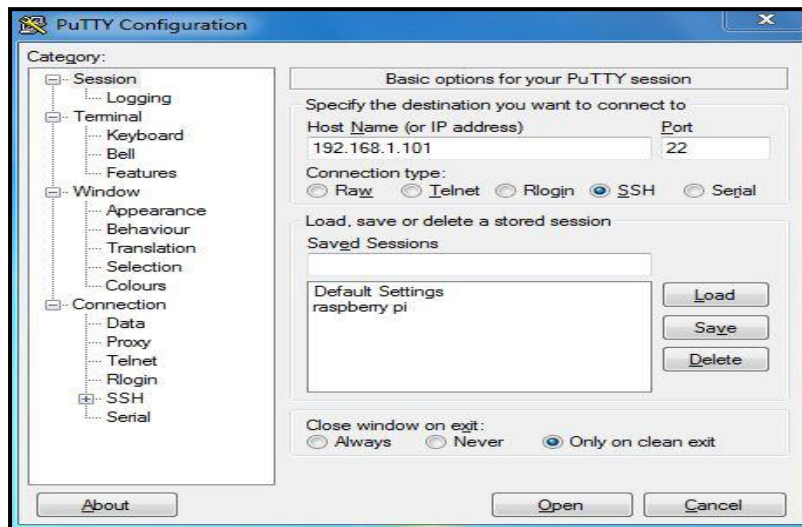


Figure-23: Check the Ethernet Connection by Used PuTTY Tool

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Accept the key (click "Yes").

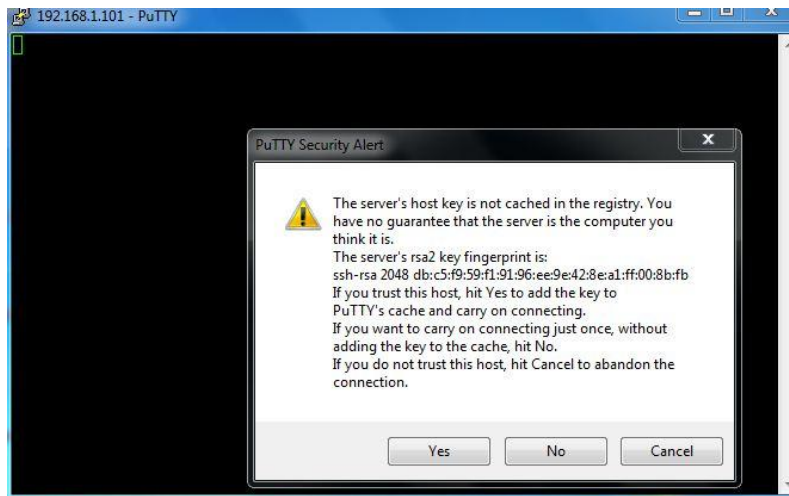


Figure-24: Accept key for PuTTY Security Alert

10. Enter login credentials. Default username is **pi** and default password is **raspberrypi**.

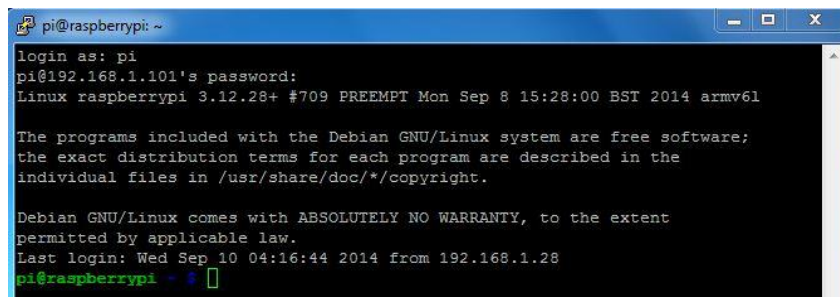


Figure-25: Raspberry Login Credential from Putty Tool

Chapter 6. Configuring AppCOE for Remote Connection to Raspberry

This chapter contains the following topics:

About AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Connecting using Remote System Explorer from AppCOE

- Once Ethernet cable is configured and made connection between the “**Raspberry Pi**” Target Board and **Host PC**
- You will see Linux booting up in the terminal window. It takes about 5 seconds for first characters to appear
- Open the AppCOE, click **Window -> Open Perspective -> Other** as shown in Figure-26.

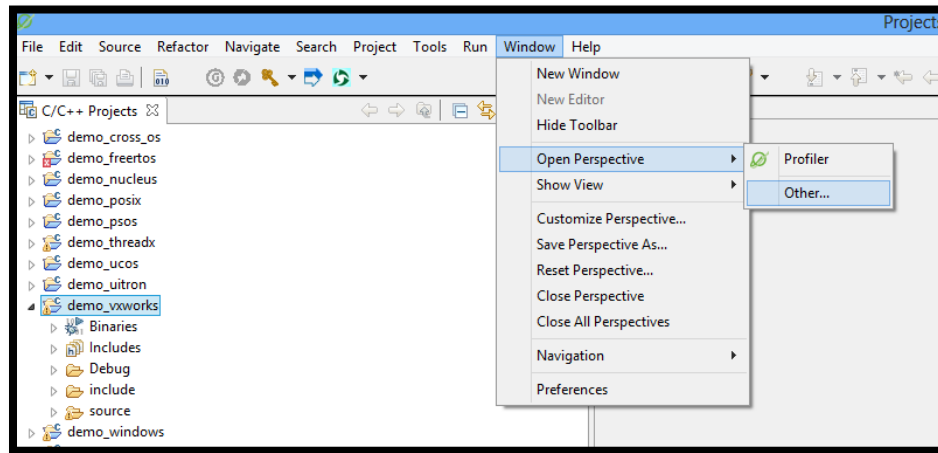


Figure-26: Navigate to other Perspective

- Select Remote System Explorer and click OK as shown in Figure-27.

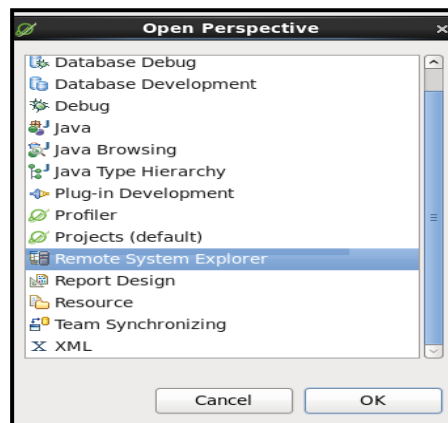


Figure-27: Select Remote System Explorer

- Click **File -> New -> Other** and then Double-click **Remote System Explorer** to expand it, select the **Connection** as shown in Figure-28.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

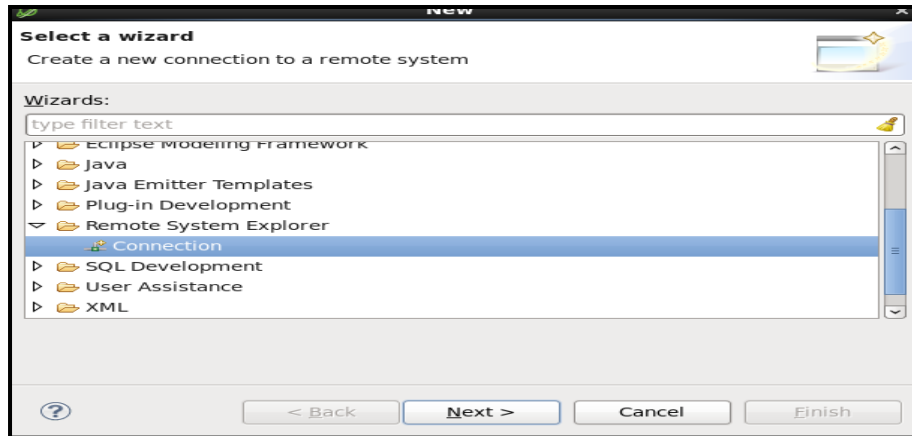


Figure-28: Selecting the Connection from Remote System Explorer

- In the New Connection window select **General -> Linux** as shown in Figure-29. Click on Next.

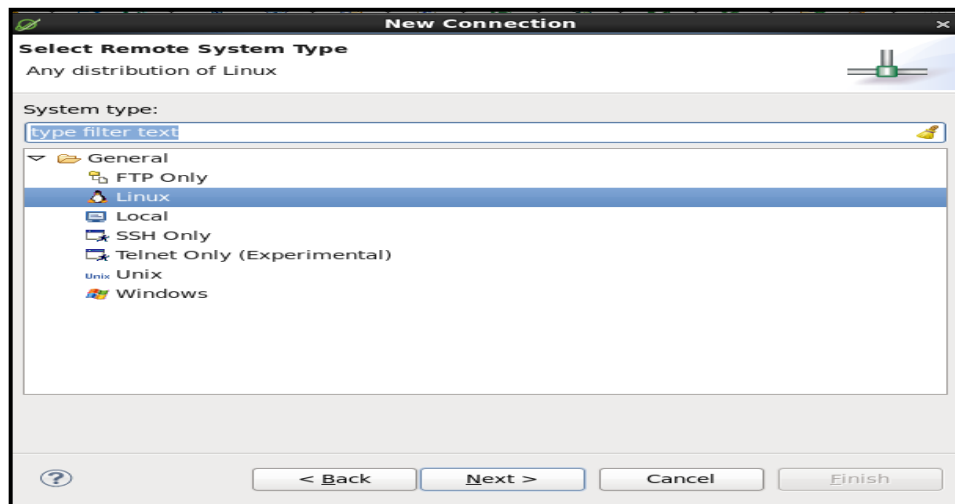


Figure-29: Selecting the Linux for Remote System type

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Select the parent profile: **localhost machine name**, fill the Host Name: box enters Raspberry Ip address: **192.168.1.101**. In Connection Name: - enter the name you want to use – this will eventually show in the Remote Systems pane. I choose “**raspberrypi**” as shown in Figure-32.

Note: In your given package if the name mentioned as rasperry will be the same.

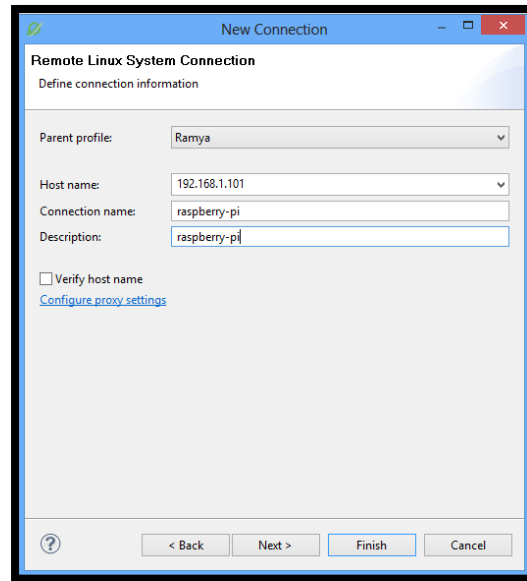


Figure-30: Define Linux System Connection information

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Click on Next and select **ssh.files** as shown in Figure-31.

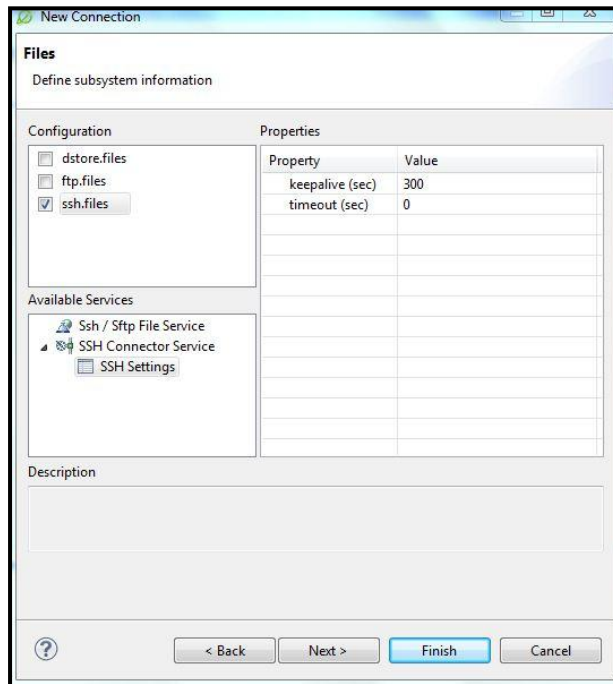


Figure-31: Selecting ssh Files Configuration

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Click on Next and select **processes.shell.linux** configuration as shown in Figure-32.

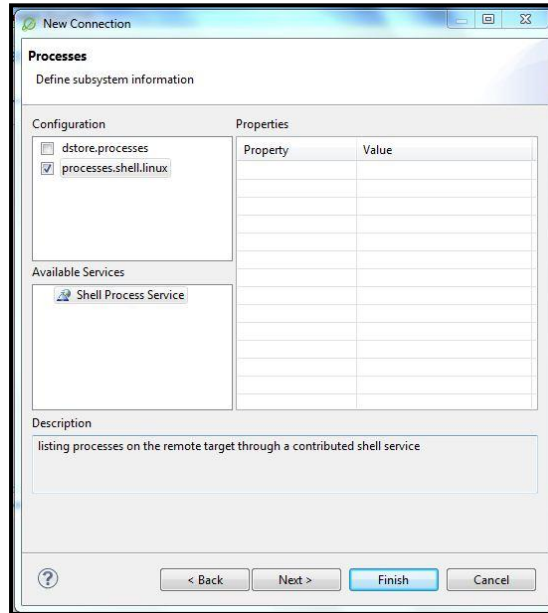


Figure-32: Selecting processes.shell.linux Configuration

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Click on Next and select **ssh.shells** configuration as shown in Figure-33.

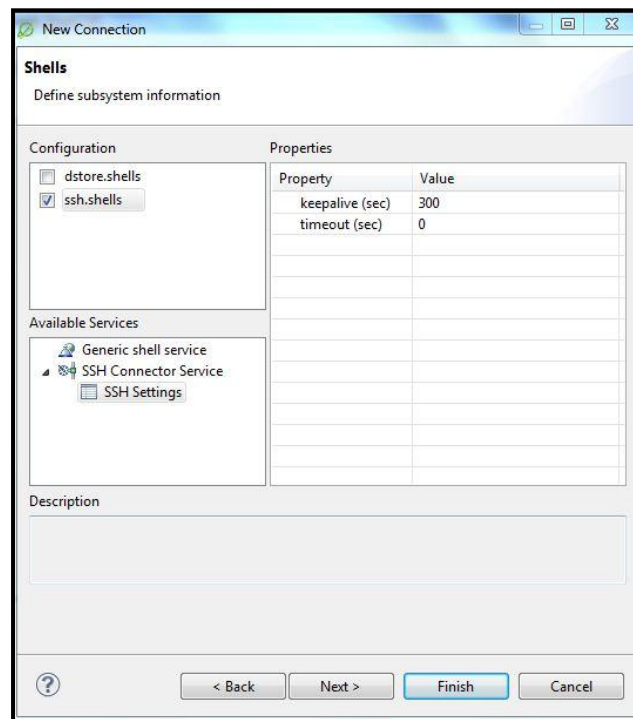


Figure-33: Selecting ssh.shells Configuration

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Click on Next and select **ssh.terminals** configuration as shown in Figure-34.

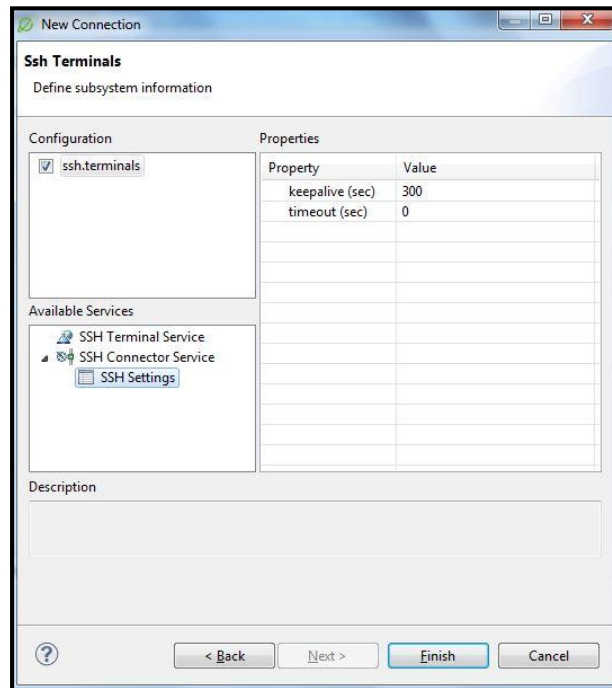


Figure-34: Selecting ssh.terminals Configuration

- Click on Next and select ssh.terminals configuration then Click on Finish and your new connection will be created.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- After this press the **"Finish"** button. Now the AppCOE window should look like Figure-35.

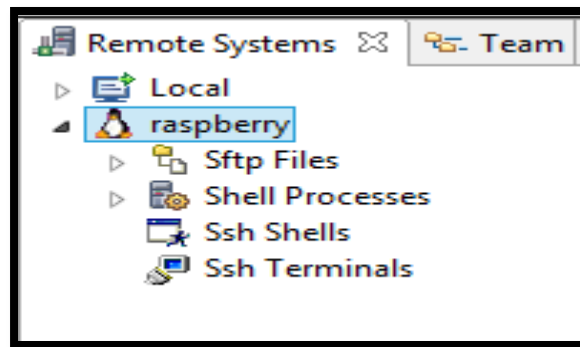


Figure-35: New Connection show on Remote System Explorer

- In the **Remote Systems** tab, right-click the **raspberry** and select **connect** options as shown in Figure-36.

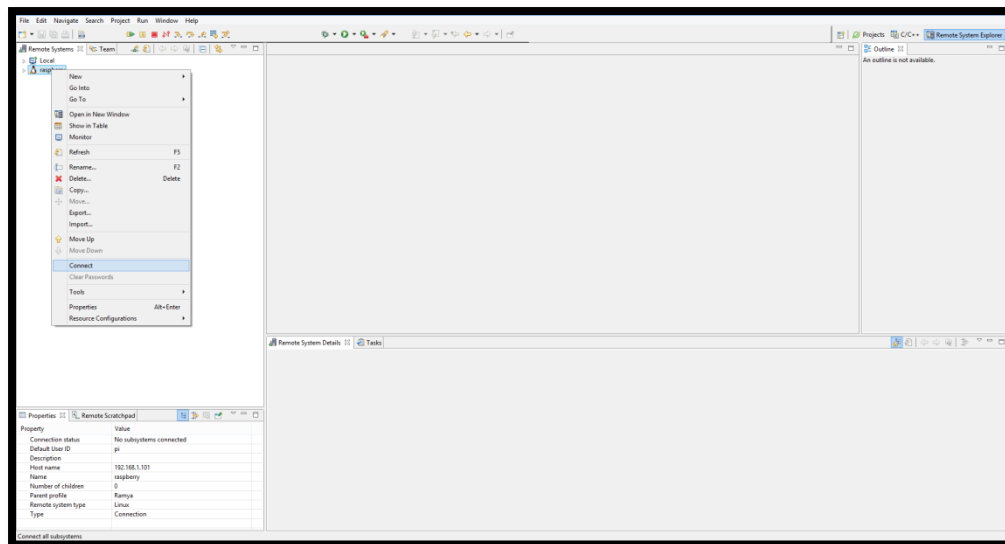


Figure-36: Selecting Connect

- If asked for a password, by Default, Raspberry **Username** is **"pi"** and **password** is **"raspberry"** as shown in Figure-37.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

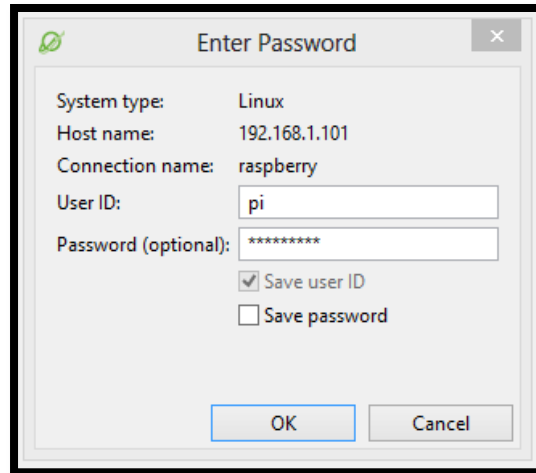


Figure-37: Setting Username & password

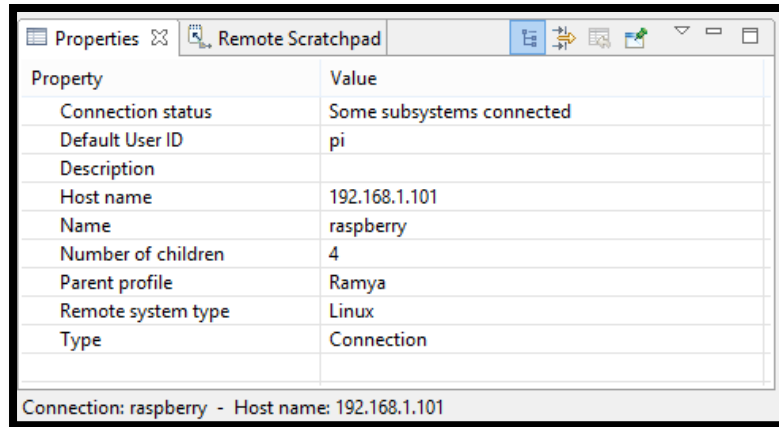
- You will see the file system and select the **Cancel** button for the secure storage password as shown in Figure-38.



Figure-38: Cancel the secure storage password

- Check the connection status from the properties tab of **Remote Systems** as shown in Figure-39.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE



Property	Value
Connection status	Some subsystems connected
Default User ID	pi
Description	
Host name	192.168.1.101
Name	raspberrypi
Number of children	4
Parent profile	Ramya
Remote system type	Linux
Type	Connection

Connection: raspberrypi - Host name: 192.168.1.101

Figure-39: Connection status

- Expand **Sftp** and then **My Home** and then **Root** as shown in Figure-40.

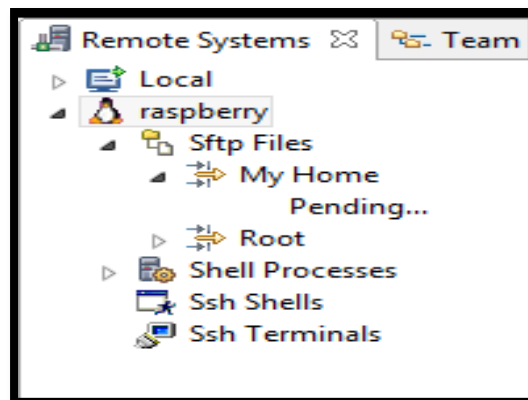


Figure-40: Expand SFTP Files

- Check the file system as shown in Figure-41.

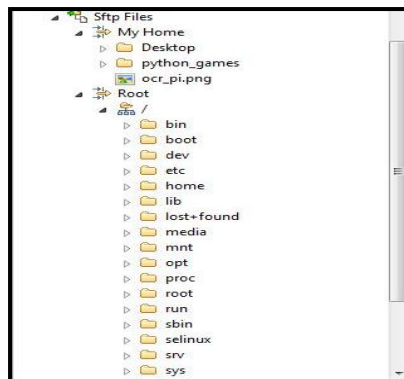


Figure-41: Check the File system

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- If you see this, everything is working correctly. If not, confirm everything is set as in the directions. Test your Ethernet connection with Ping. Make sure the target booted correctly.
- If you have trouble with the username or password, right click on your connection name (Raspberry_pi in this case) and select Clear Password and attempt a connection again
- Right click on **Ssh Terminals** and select Launch Terminal as shown in Figure-42.

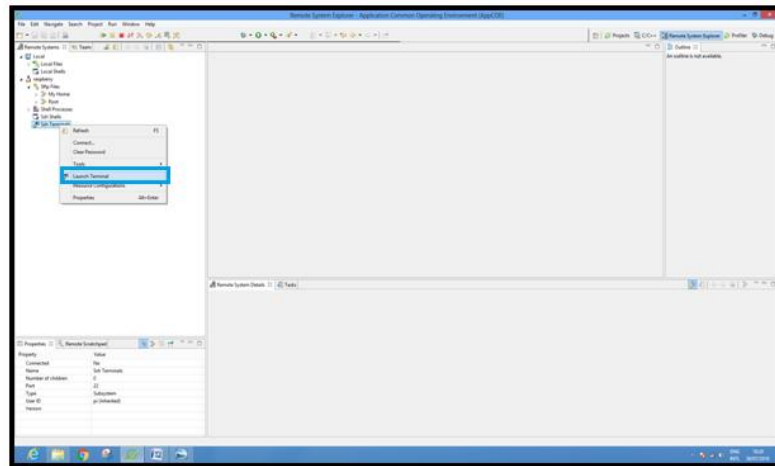


Figure-42: Launch Terminal

- A terminal will be created in the Terminal pane as shown in Figure-43.

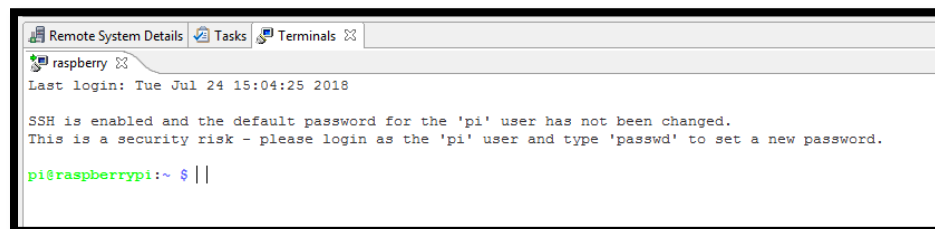


Figure-43: Terminal launched on Terminal pane

Chapter 7. Building Demo Projects on AppCOE

This chapter contains the following topics:

About AppCOE

Building Demo Projects on AppCOE Host.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Building Demo Projects on AppCOE Host

We are providing the binaries of all demos that will be deployed and run on Raspbian target of ARM 1176 jzf-s board as shown in Figure-44.

Note: As per your license, you can able to run only those demos on ARM target.

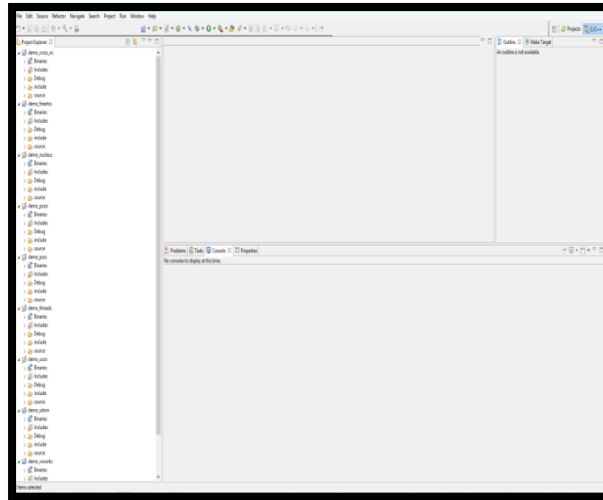


Figure-44: Raspberry demos with binaries

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

If you want to build the project on AppCOE host then do the step shown in Figure-45.

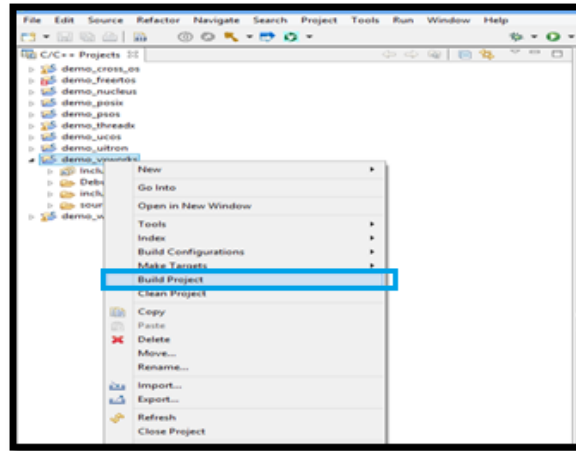


Figure-45: Build the demo

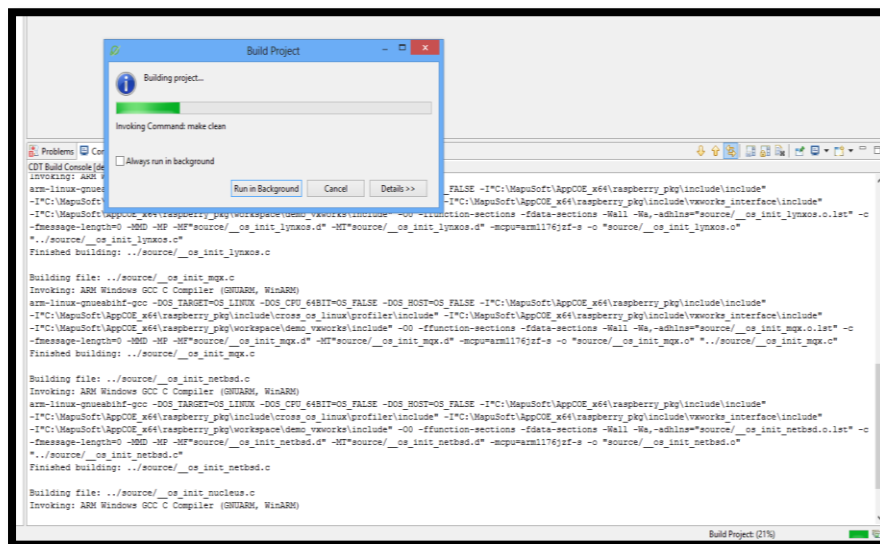


Figure-46: Building project

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

If you want to clean the project then do the step shown in Figure-47.

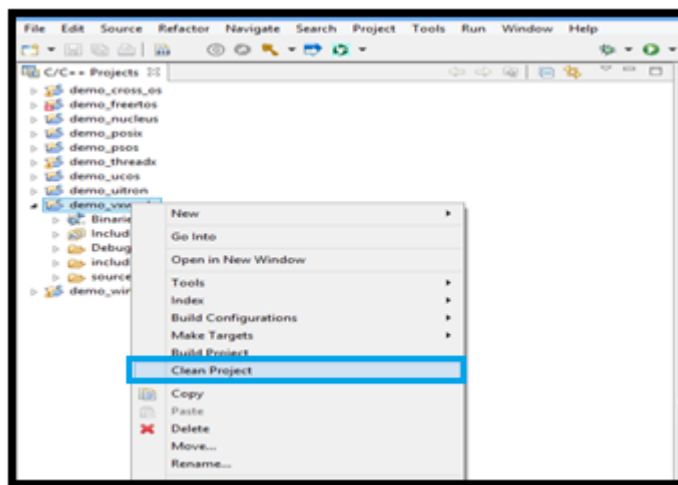


Figure-47: Clean the demo project

Chapter 8. Running the Demo Projects on Raspberry-Pi

This chapter contains the following topics:

About AppCOE

Setting up the Run Configuration in AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Setting up the Run Configuration

- Select any demo project, for example, select **demo_vxworks** project and go to **Run as->Run Configurations** as shown in Figure-48.

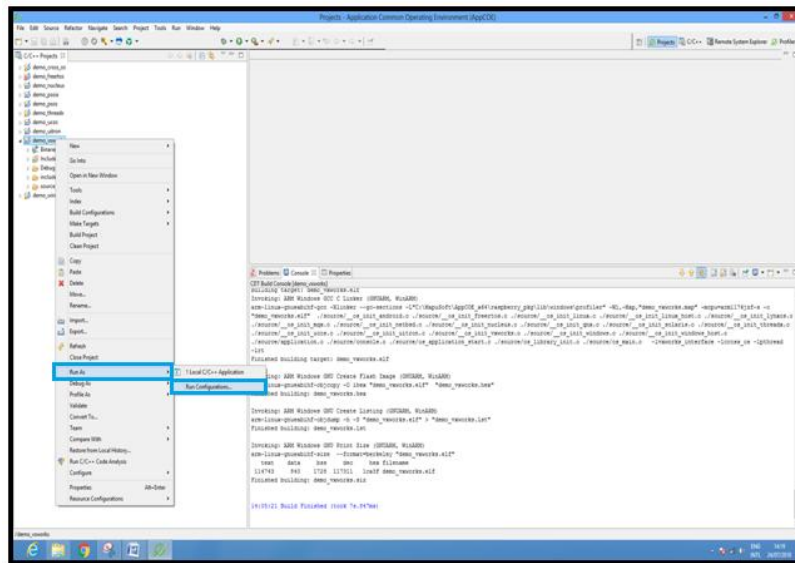


Figure-48: Navigate to Run Configuration

- Select the **C/C++ Remote Application** then create a new launch configuration by right click and select **New** as shown in Figure-49.

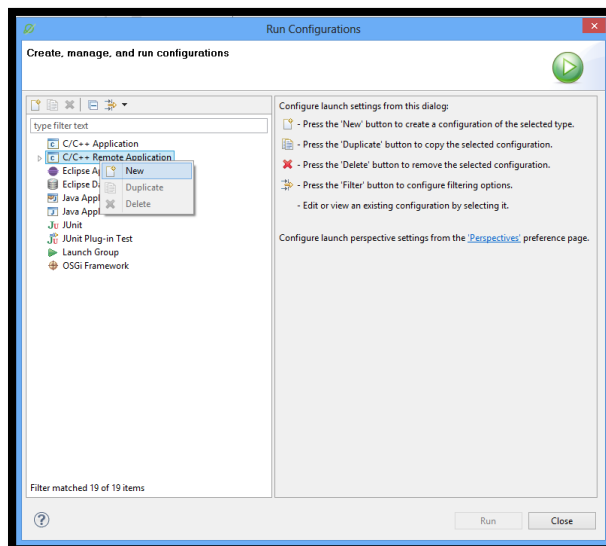


Figure-49: Remote application in Run configuration

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

After select New, the appropriate demo_vxworks Debug project will be shown as in Figure-5, when you expand the C/C++ Remote Application.

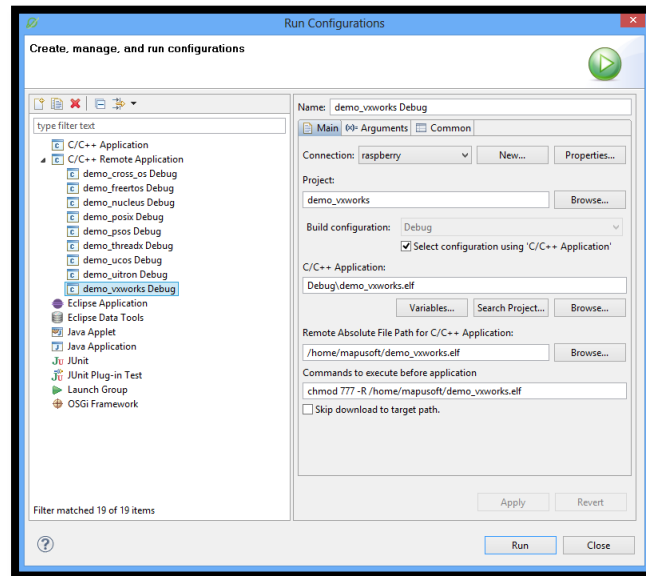


Figure-50: Create a new launch configuration

- Select the appropriate project **demo_vxworks Debug** and then make given configurations below.
- In the **Main** tab, at Connection, select the connection **raspberry_pi** you create above
- At **Project**, browse the project you created earlier
- At **Build Configuration**, select configuration using C/C++ Application
- At **C/C++ Application**, browse to the **demo_vxworks** executable file
- At **Remote Absolute File Path for C/C++ Application**, browse to or enter the directory and the filename where you want to upload the executable file on the Raspberry Pi target Board, for example: I Created a directory:"mapusoft" from path /home , give read/write permission to "mapusoft" directory

/home/mapusoft/arm_project.elf

- At Commands to execute before application, enter chmod 777 followed by the location of the uploaded executable file on the Raspberry Pi Target Board:

chmod 777 /home/mapusoft/arm_project.elf

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Click **common tab** and select the **Run** option from **Display in favourites menu** and click **Apply** option marked as green arrow shown in Figure-51.
- Finally click the **Run** option marked as blue arrow shown in Figure-51.

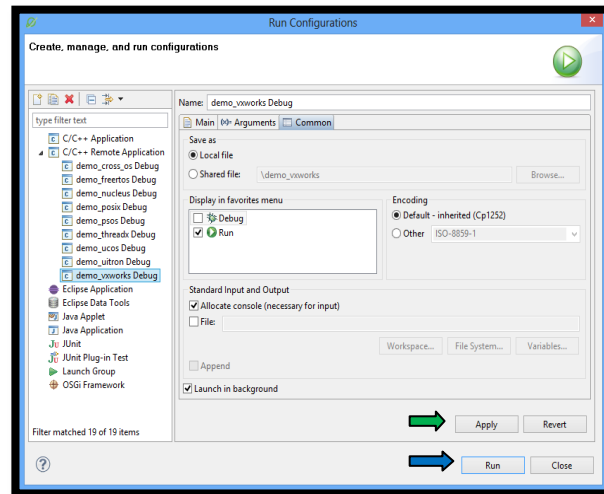


Figure-51: Select Run option from Display in favourites Menu

- The printf output should appear in the Console tab in AppCOE as shown in Figure-52.



Figure-52: Console output

Chapter 9. Debugging the Demo Projects on Raspberry-Pi

This chapter contains the following topics:

About AppCOE

Setting up the Debug Configuration in AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Set up the Debug Configuration in AppCOE

- Select the **demo_vxworks** project and go to **Debug As->Debug Configurations** as shown in Figure-53.

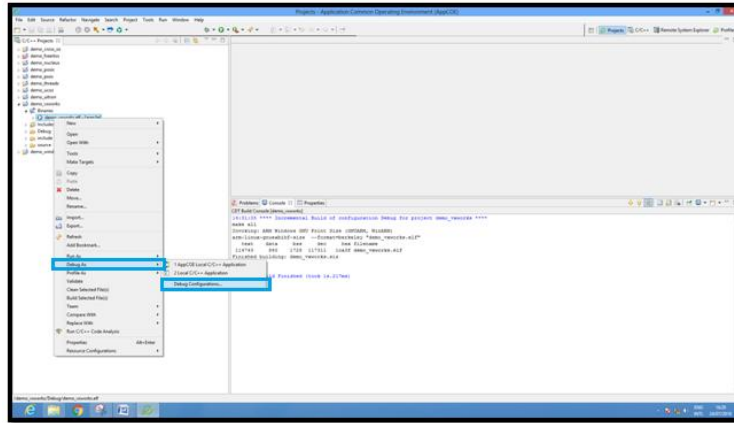


Figure-53: Navigate to Debug Configuration

- In **C/C++ Remote Application**, select **demo_vxworks Debug** and then make appropriate configurations as shown in Figure-54.

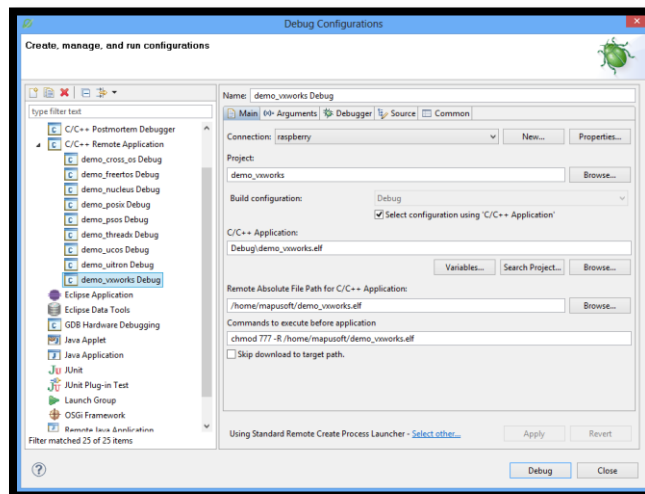


Figure-54: Debug launch configuration

- In the **Main** tab, at Connection, select the connection **raspberry** you create above
- At **Project**, browse the project you created earlier

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- At **Build Configuration**, select configuration using C/C++ Application
- At **C/C++ Application**, browse to the **demo_vxworks** executable file
- At **Remote Absolute File Path for C/C++ Application**, browse to or enter the directory and the filename where you want to upload the executable file on the Raspberry Pi target Board, for example: I Created a directory.”mapusoft” from path /home , give read/write permission to “mapusoft” directory

/home/mapusoft/arm_project.elf

- At **Commands to execute before application**, enter `chmod 777` followed by the location of the uploaded executable file on the Raspberry Pi Target Board:

chmod 777 -R /home/mapusoft/arm_project.elf

- Click **common tab** and select the **Debug** option from **Display in favourites menu** shown in Figure-55.

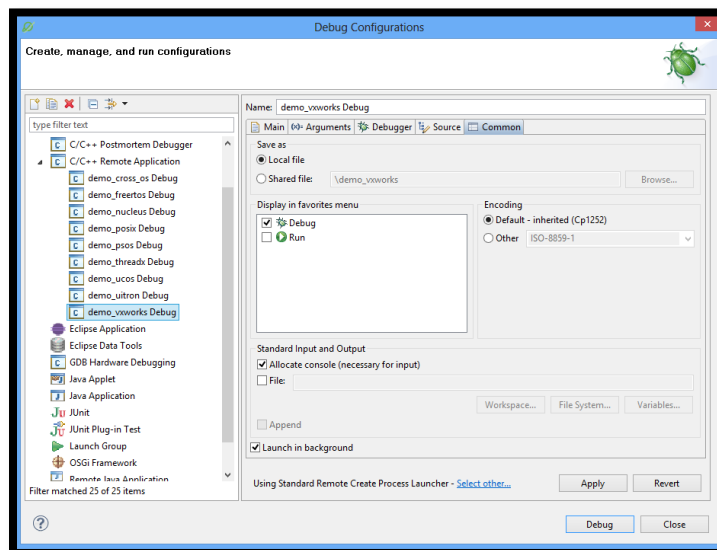


Figure-55: Remote debug launcher

- Before debug the application , you need to check the following two things:
 1. Need to Create a **.gdbinit text file**.
 2. Need to Check that **gdbserver** installed in the Target OS.

1. Need to Create a .gdbinit text file.

- On the Development Host PC, Go to project's Directory,
For ex:<AppCOE_workspace>**demo_vxworks\Debug**"

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Create a text File,<AppCOE_workspace>**demo_vxworks\Debug\.gdbinit**” with the following contents:

```
"set sysroot ${ProjDirPath}/../../SysGCC/Raspberry/arm-linux-gnueabihf"
```

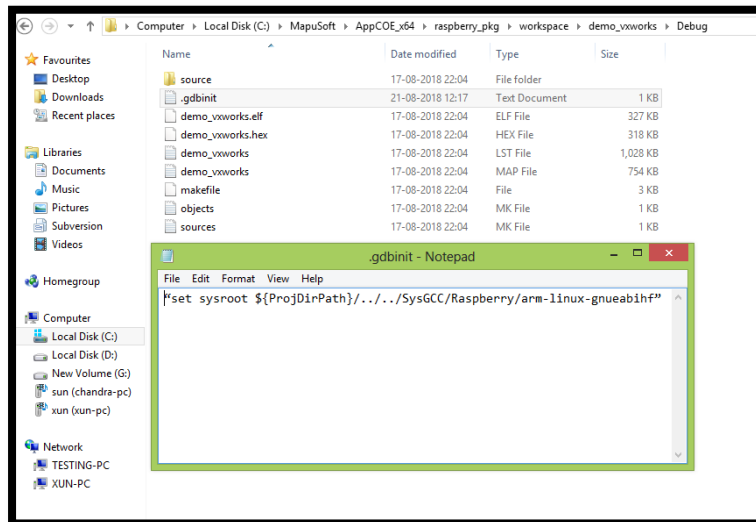


Figure-56: Create a .gdbinit text file

- Select the remote debug launcher by **Select other**.
- Select Preferred launcher Windows is appeared. Under Launcher, select the **Standard Remote Create Process Launcher** and click OK as shown in Figure-57.

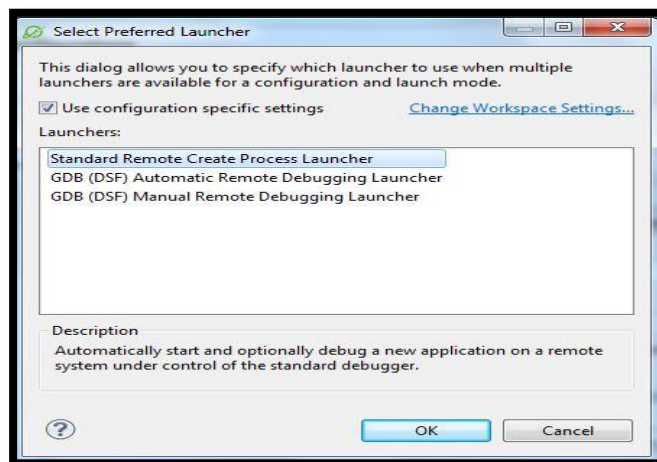


Figure-57: Standard Remote Create Process Launcher

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- In **Debugger** Tab, select the **Remote gdb/mi** Debugger.
- At **GDB debugger**, enter the location of arm-linux-gnueabi-hf-gdb on your Host PC
Location : " C:\MapuSoft\AppData\Local\Programs\MapuSoft\arm-linux-gnueabi-hf-gdb\bin\arm-linux-gnueabi-hf-gdb"

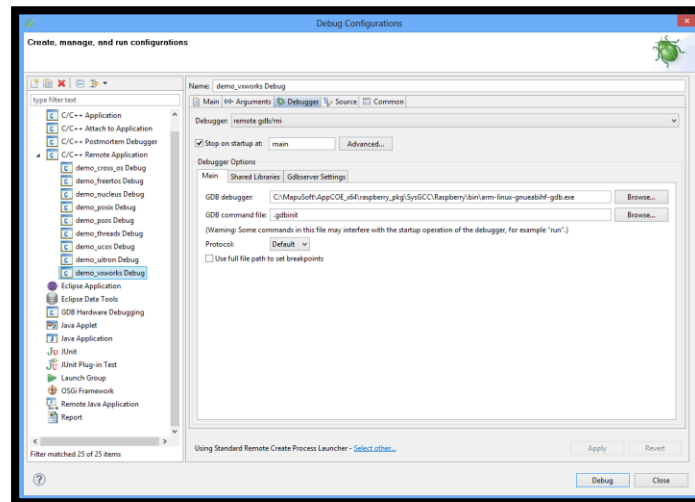


Figure-58: Select the Remote gdb/mi Debugger

- Caution: when I inadvertently added a space after the debugger name and path, on attempting to debug, I got this error:
Error creating session
- Note that the default settings select Stop on startup at: main, which means that when you run the debugger, it will stop at the first line in main as shown in Figure-59.

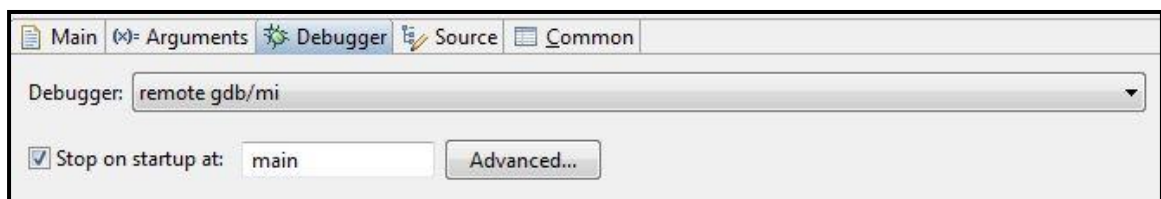


Figure-59: Default settings select Stop on startup at: main

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Under Debugger Options, select the **Gdbserver** Settings tab, At Port number; enter a free TCP port (2345 in the example) as shown in Figure-60.

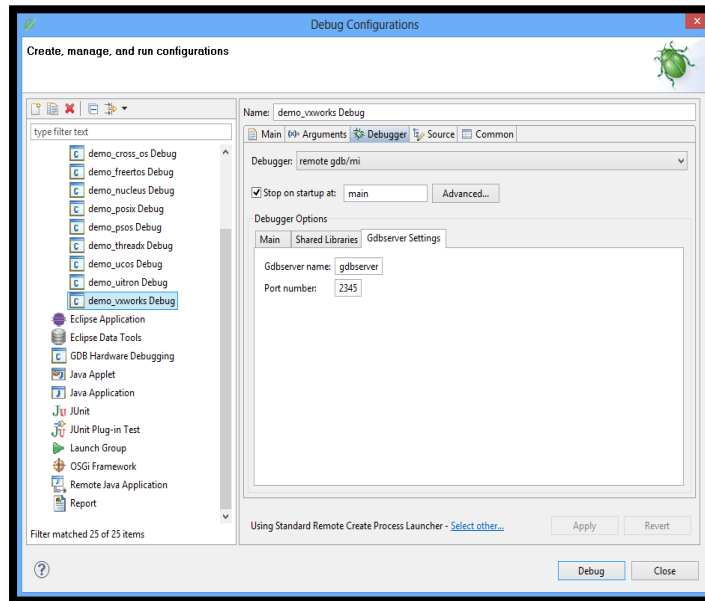


Figure-60: Default settings for Gdbserver settings tab

- Click Apply.

2. Need to Check that gdbserver installed in the Target OS

- Make sure that gdbserver installed on the target OS: "Raspberry-pi", if not already installed, install with the following commands

sudo apt-get install gdbserver

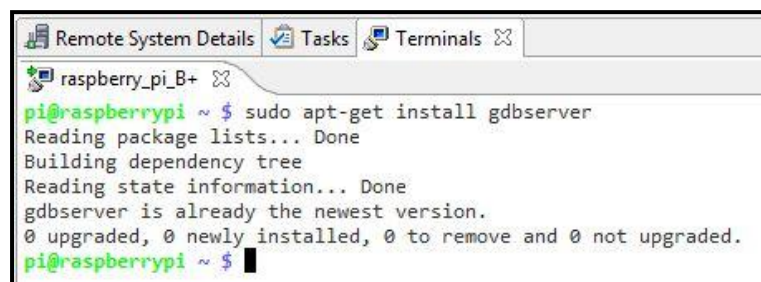


Figure-61: Checking gdbserver installed on TargetOS

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Start Remote Debugging

- Click Debug
- If asked to Confirm Perspective Switch, click yes as shown in Figure-62.

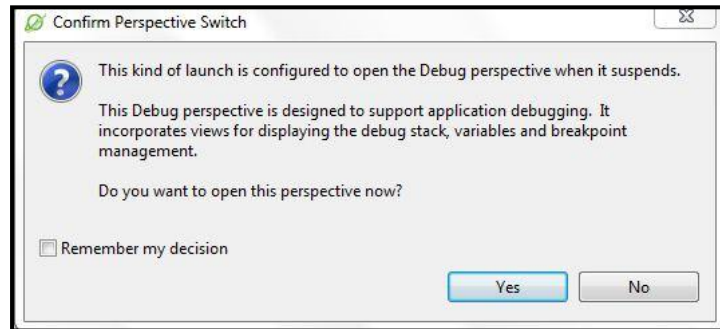


Figure-62: Confirm perspective switch

- AppCOE should now upload vxworks_project to the Raspberry_pi and begin remote debugging. With the default setting to stop on **start-up** at main as shown in Figure-63, you should see the program stopped at the first program line.

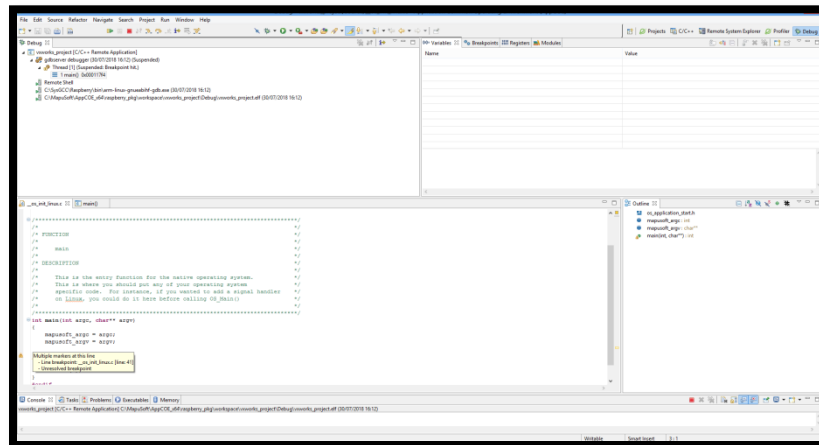


Figure-63: Remote Debugging Stop on main

- Resume the remote debugging by click resume button and then put breakpoints on required place of code.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- The printf output should appear in the Console tab in AppCOE as shown in Figure-64.

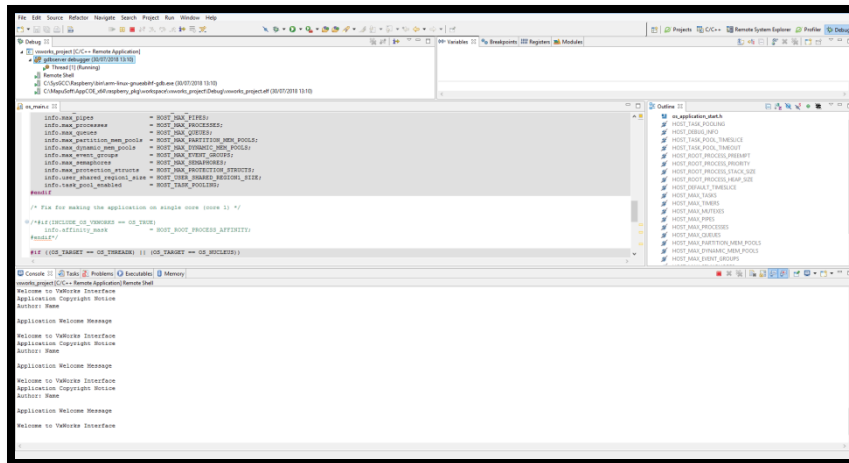


Figure-64: Console output

- Once remote Debugging is complete, you should terminate the Remote Debugging by right click the vxworks_project_debug then select Terminate, then output screen look like below:Figure-65

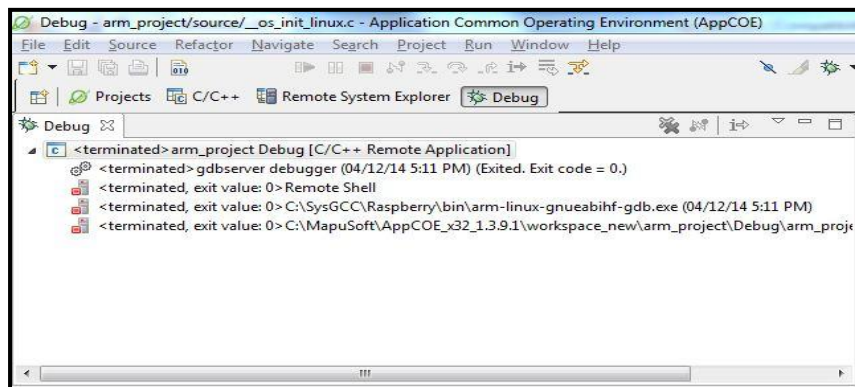


Figure-65: Terminate Remote Debugging

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Once remote Debugging is complete, you should disconnect otherwise to debug the other different demos on Raspberry –Pi B+ Target Board from windows Host PC as shown in Figure-66.

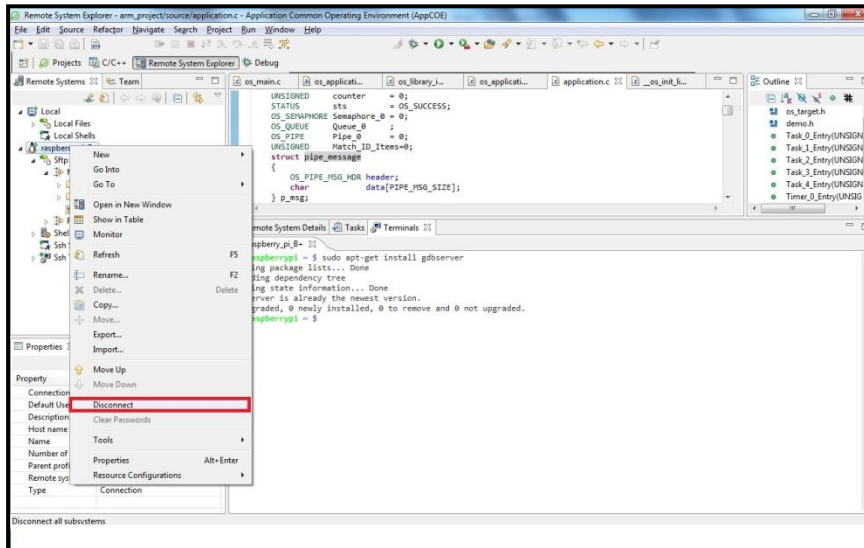


Figure-66: Disconnect the Connection

10. Creating ARM Cross OS Target Application on AppCOE

This chapter contains the following topics:

About AppCOE

Steps to create your own Cross OS Target Application on AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Steps to create your own Cross OS Target Application on AppCOE

1. Create an ARM Cross Target Application Project by select **File > New > Project**. Under Project Types, select the **C project** and click Next as shown in Figure-67.

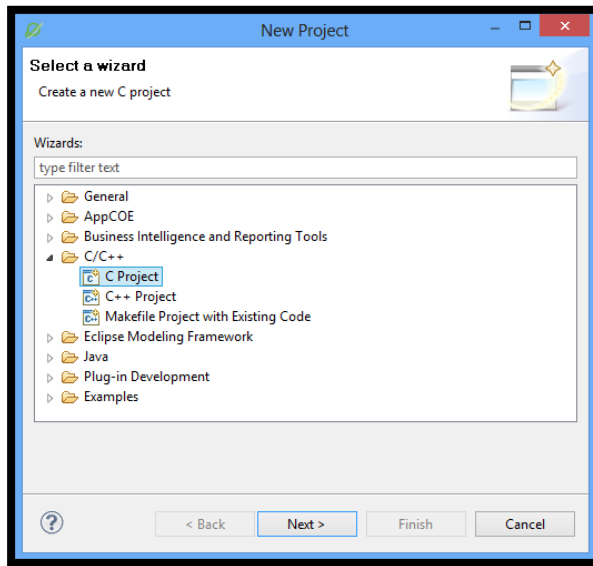


Figure-67: Select a wizard

2. On C Project Wizard window, type a project name is “**vxworks_project**” and gives a location next to **Project Name** text box, by default workspace path was selected.
3. Under Project Types, expand the **ARM Cross Target Application** menu. Select **Empty Project** and then select the Tool chain: ARM Windows GCC for Windows Host (GNUARM, WinARM), click **Next** as shown in Figure-68.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

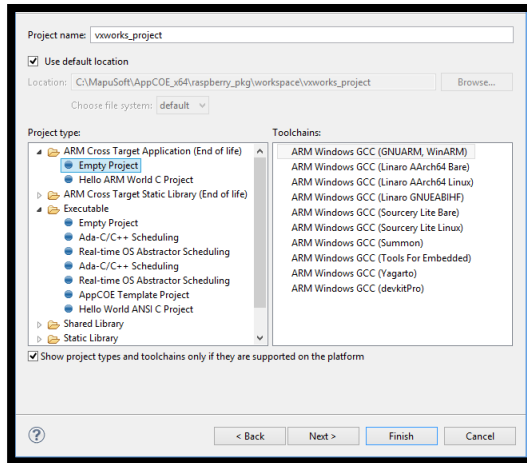


Figure-68: Creating the ARM Cross Target Application Project

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

4. Press the "**Finish**" button, now ARM Cross Target Application got successfully created.
5. After project has been created, project explorer window looks like below as shown in Figure-69.

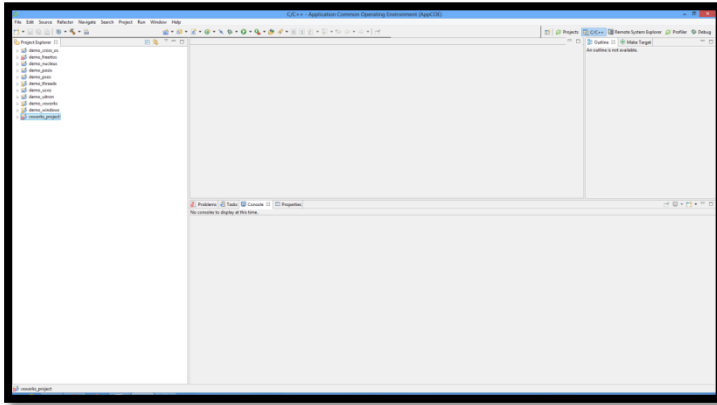


Figure-69: Project Explorer view

6. Add required include and source files based on application.
7. As of now for testing purpose can import the "include & source file" of vxworks_project. Right click the vxworks_project and select import option.
8. Now select **File system** in **General** and click next. Now browse the directory to import the "include & source file" and then select the files by give **Select All** and click finish.

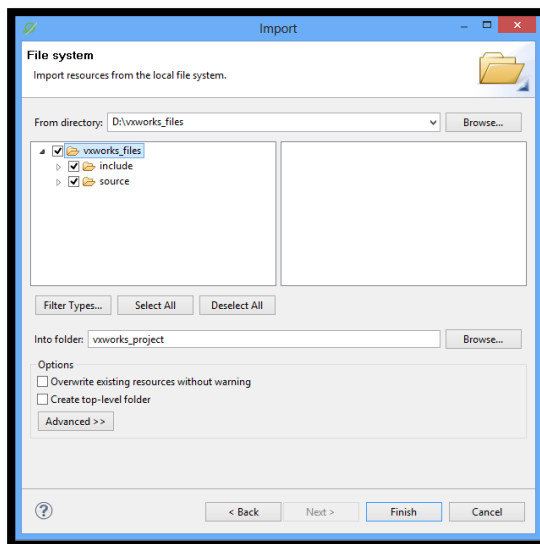


Figure-70: Import file

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

9. By refreshing your “vxworks_project” project in the workspace to update include & source files in Project Explore as shown in Figure-71.

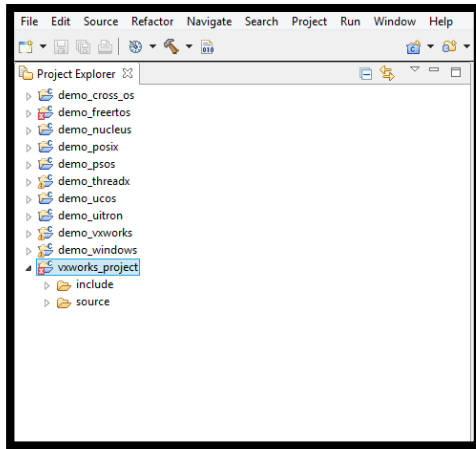


Figure-71: Refresh project to view the imported files

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

10. Select the **"vxworks_project"** project, then use the "Project" menu again & select Properties
11. Select the "Discovery Options" under "C/C++ Build" page from **"vxworks_project"** properties window
12. In windows host, modified the "Compiler invocation command" with **"arm-linux-gnueabi-hf-gcc"** for ARM Windows GCC Assembler and Compiler.

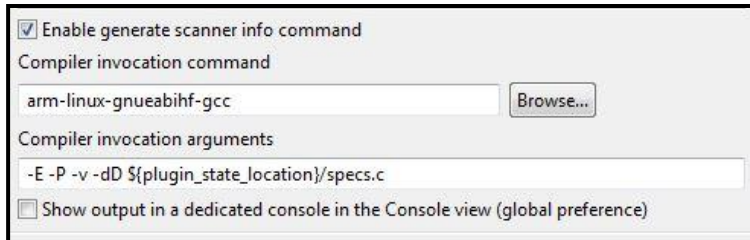


Figure-72: Changing command invocation command for project

13. Select **"vxworks_project"** project, then use the "Project" menu again and select "Properties" then select Environment option on left hand side and click on Add option.
14. Now include your Raspberry tool- chain directories,

- fill in **Name:** "PATH"

In windows host, fill in **Value:** `${ProjDirPath}/../../SysGCC/Raspberry/bin`

- Finally select **apply** and OK.

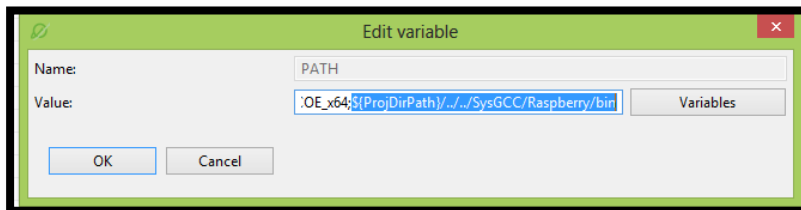


Figure-73: Path variable setting

15. Select the **"vxworks_project"** project and go to Project Properties->C/C++ Build>Tool Settings>Target Processor, then set the following options for Target Processor:
 - **Processor:** ARM1176 jzf-s
 - **Architecture:** Toolchain Defaults
 - **Thumb:** OFF
 - **Endianness:** Toolchain Default
 - **Float ABI:** Toolchain Default

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- **FPU Type:** Toolchain Default

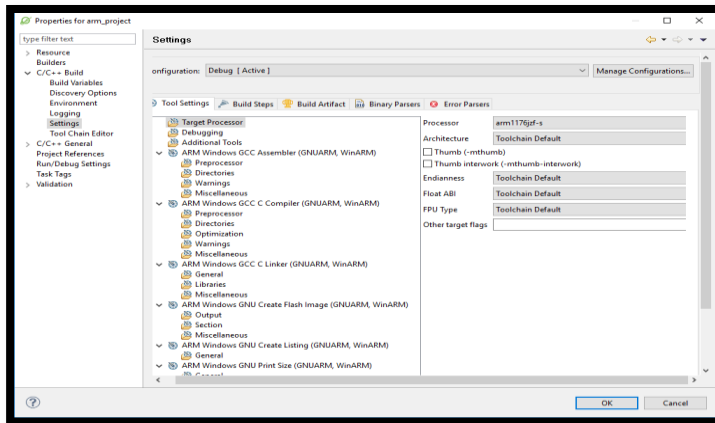


Figure-74: Target Processor Settings

16. Select the **“vxworks_project”** project and go to Project Properties>C/C++ Build>Tool Settings>change Command as **‘arm-linux-gnueabi-hf-gcc’** in command box for ARM Windows GCC Assembler, ARM Windows GCC C Compiler and ARM Windows GCC Linker as shown in Figure-75.

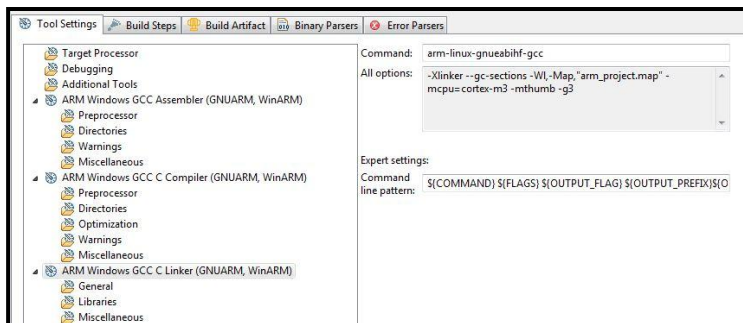


Figure-75: Changing Command for ARM Windows GCC Linker

17. Select the **“vxworks_project”** project and go to Project Properties>C/C++ Build>Tool Settings>ARM Windows GCC Compiler>Preprocessor as shown in Figure-76.

Add these below Preprocessor symbols

- **OS_TARGET=OS_LINUX**
- **OS_HOST=OS_FALSE**
- **OS_CPU_64BIT=OS_FALSE**

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

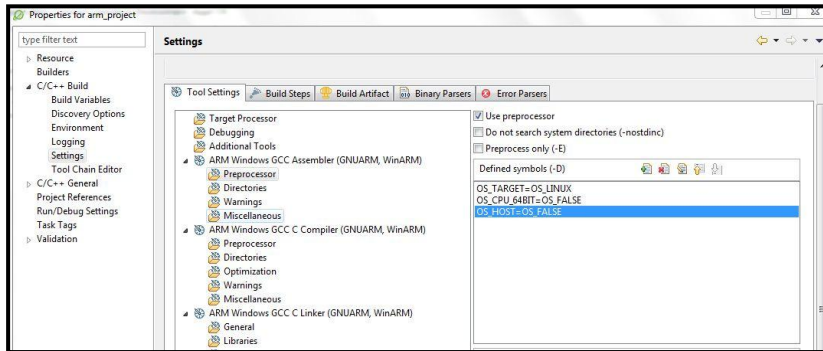


Figure-76: Setting Preprocessor Symbols

18. Select the **"vxworks_project"** project and go to Project Properties>C/C++ Build>Tool Settings >ARM Windows GCC Assembler & ARM Windows GCC C Compiler >Directories

Add those include Files from the following Path:

```
"${ProjDirPath}/../include/include"
```

```
"${ProjDirPath}/../include/cross_os_linux/nonprocess/include"
```

```
"${ProjDirPath}/../include/vxworks_interface/include"
```

```
"${workspace_loc}/${ProjName}/include"
```

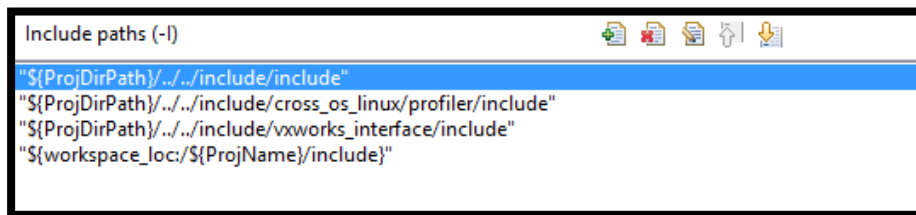



Figure-77: Added Include paths

19. Click add  and browse include file from workspace in add directory path as shown in Figure-78.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

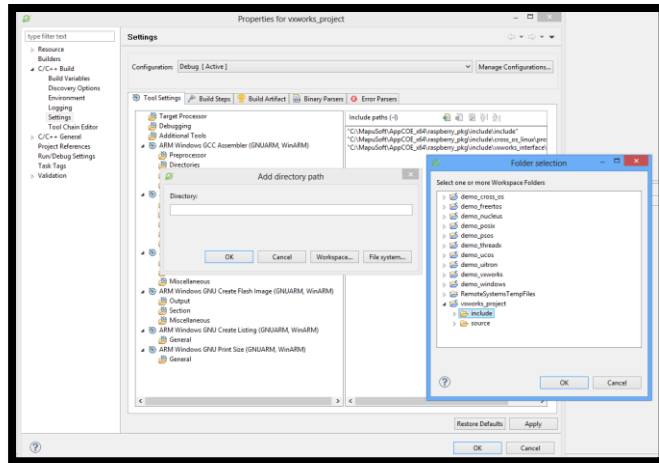


Figure-78: Added directory paths

20. Default Miscellaneous settings for ARM Windows GCC Compiler > Miscellaneous as shown in Figure-79.

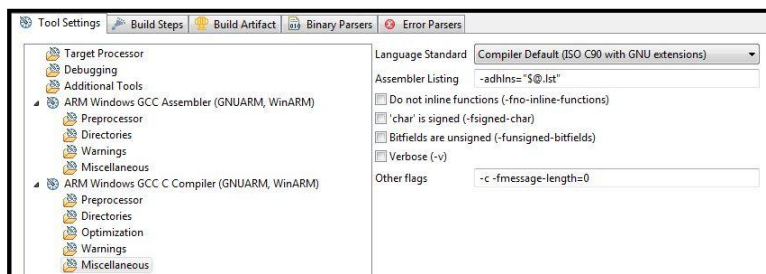


Figure-79: Default Miscellaneous settings for ARM Windows GCC Compiler

21. Add the Libraries value from **Settings>ARM Windows GCC Linker>Libraries** and Library values as shown in Figure-80.

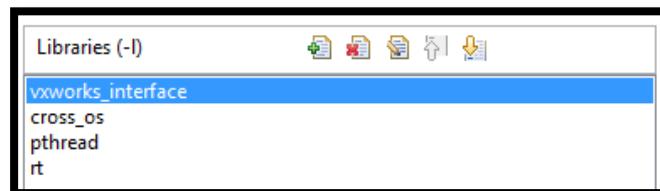


Figure-80: Libraries Values

22. Add the Generated cross_os Library search path as like below as shown in Figure-81.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

```
"${ProjDirPath}/../../lib/nonprocess"
```

```
"${ProjDirPath}/../../lib/process"
```

```
"${ProjDirPath}/../../lib/profiler"
```

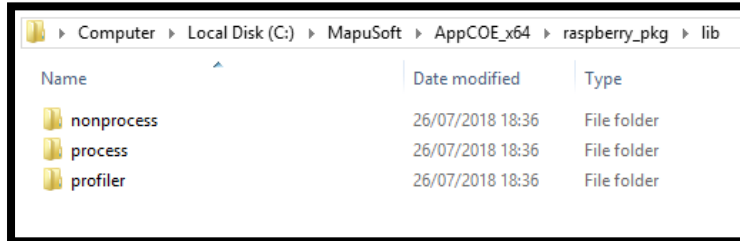


Figure-81: Libraries path

23. Find the profiler, process mode & non-process mode libraries under Lib directory as shown in Figure-82.

Note: Based on your application, select either process mode or non process mode library

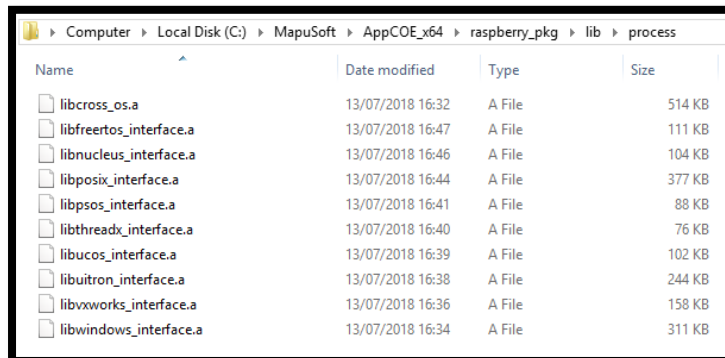


Figure-82: Process mode library

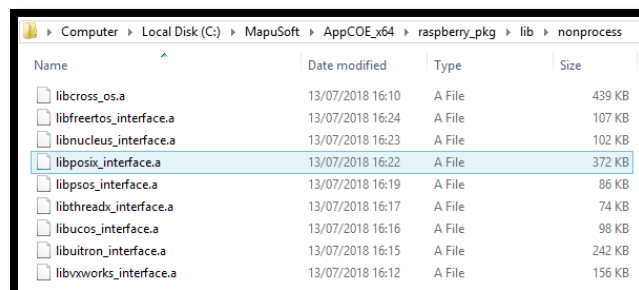


Figure-83: Non Process mode library

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

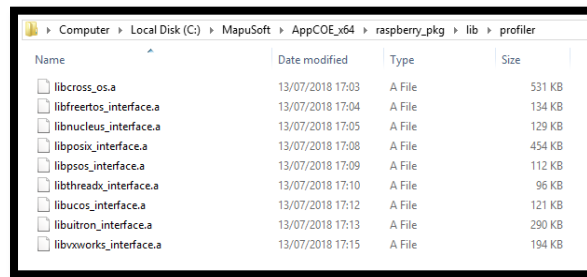


Figure-84: Profiler mode library

24. Click the Add button in library search path and select the path shown in Figure-85.

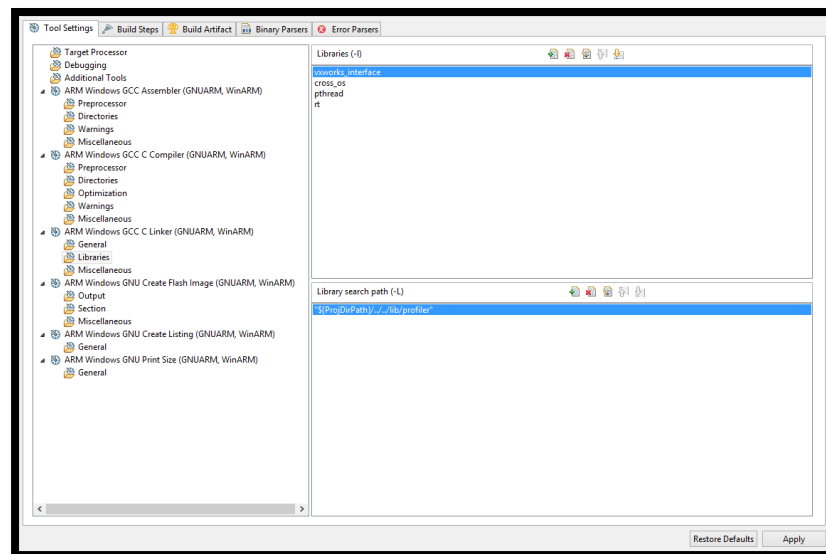


Figure-85: Libraries values & Search path

25. Enable the Create Flash image, Create Extended Listing, print size tools from Additional Tools option as shown in Figure-86.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

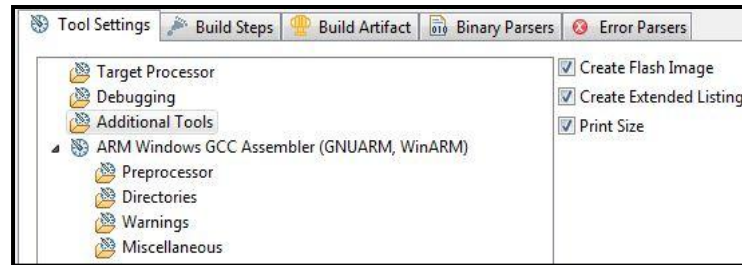


Figure-86: Additional Tools

26. Select the “**vxworks_project**” project and go to Project Properties>C/C++ Build>Tool Settings>for windows ARM Windows GNU Create Flash Image; change Command as ‘**arm-linux-gnueabi-hf-objcopy**’ in command box as shown in Figure-87.

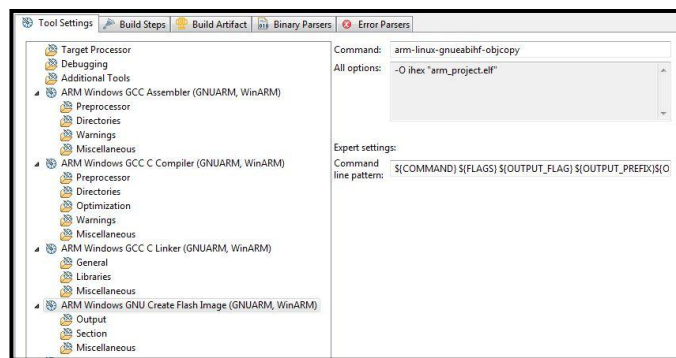


Figure-87: Changing Command in ARM Windows GNU Create Flash Image

27. Select the “**vxworks_project**” project and go to Project Properties>C/C++ Build>Tool Settings>ARM Windows GNU Create Listing, change Command as ‘**arm-linux-gnueabi-hf-objdump**’ in command box as shown in Figure-88.

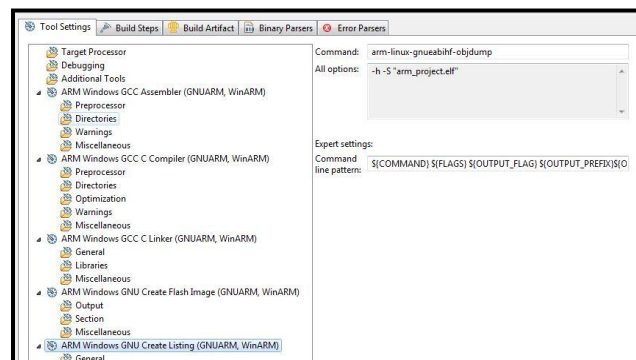


Figure-88: Changing Command in ARM Windows GNU Create Listing

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

28. Select the “**vxworks_project**” project and go to **Project Properties>C/C++ Build>Tool Settings>ARM Windows GNU Print Size**, change Command as ‘**arm-linux-gnueabi-hf-size**’ in command box as shown in Figure-89.

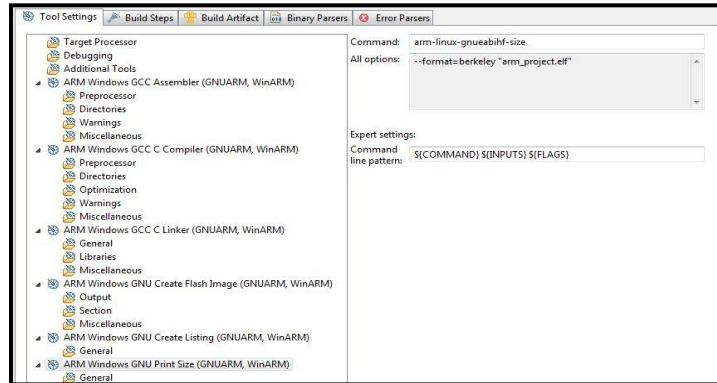


Figure-89: Changing Command in ARM Windows GNU Print Size

29. Select the “**vxworks_project**” project and go to **Project Properties>C/C++ Build>Settings>Binary Parsers**, Enable the GNU Elf parser option for “Elf” Executable Format as shown in Figure-90.

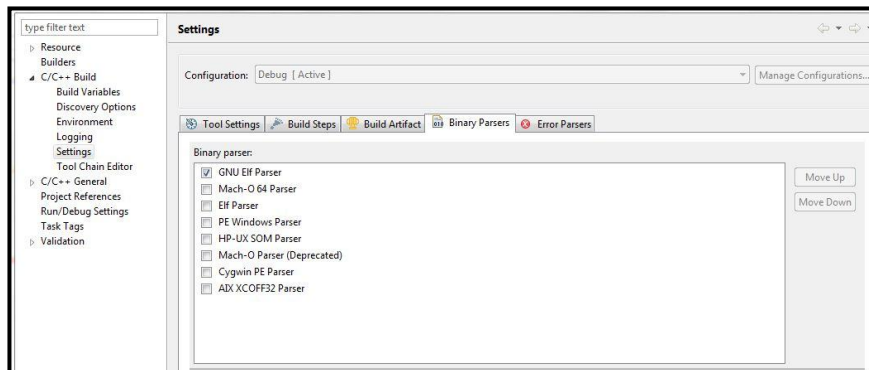


Figure-90: Enable the GNU Elf parser option

30. Select the “**vxworks_project**” project and go to **Project Properties>C/C++ Build>Settings>Binary Parsers**, Check all Error Parser for Find Out the **C/C++ Compiler, Linker, Assembler Error & Warnings** as shown in Figure-91.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

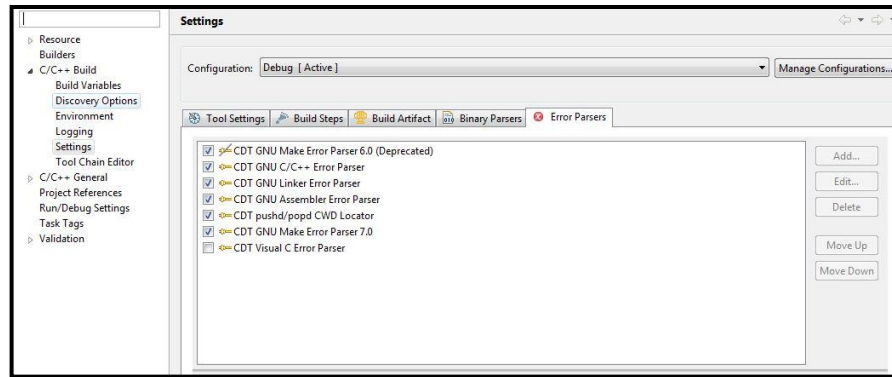


Figure-91: Enable the GNU Compiler, Linker, Assembler Error parser

31. Before building the “**vxworks_project**” project, you should check the included directories are included by click the “**vxworks_project**” dropdown option as shown in Figure-92.

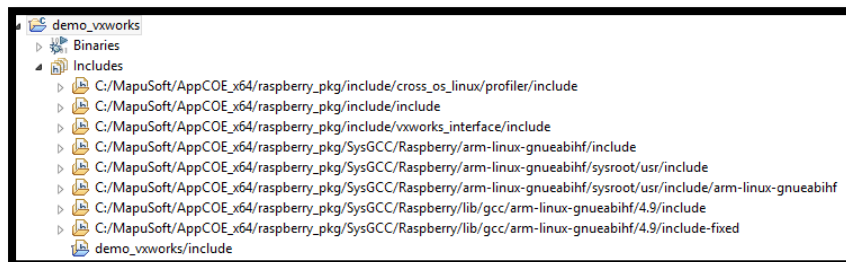


Figure-92: Check included directories for vxworks_project for Windows Host

32. Select a “**vxworks_project**” project under C/C++ Projects pane, right click and select **Build Project** as Build an Application project as shown in Figure-93.

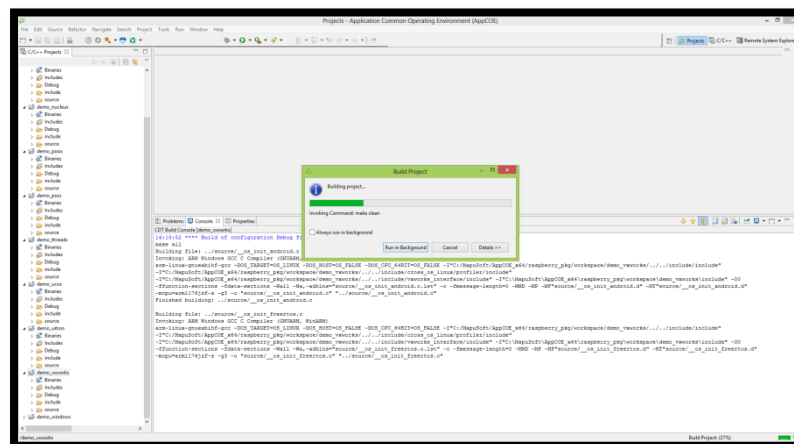


Figure-93: Building vxworks_project Application Project

33. After Building a vxworks_project, **vxworks_project.elf** was generated as shown in Figure-94.



11. Steps to Install Plugins on AppCOE

This chapter contains the following topics:

About AppCOE

Installing Remote Debugging Capabilities

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Installing GNU ARM C/C++ Development Tools

- Download the **0.5.5- GNU ARM Eclipse Plug-ins** from this weblink <http://sourceforge.net/projects/gnuarmeclipse/files/Current%20Releases/0.5.5/>
- In AppCOE, Open “**Help**” ->“**Install New Software**” in the main menu as shown in Figure-97.

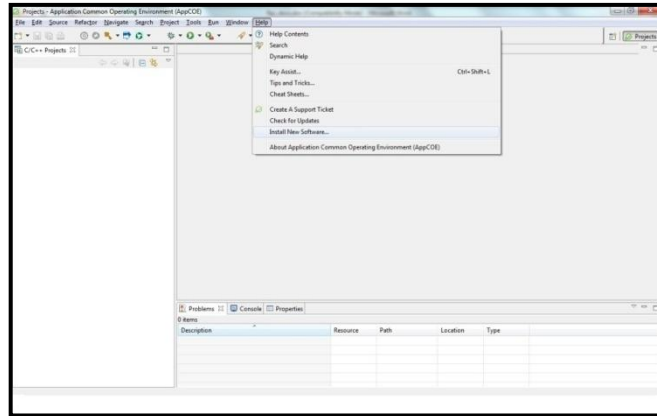


Figure-95: Install New Software

In the **Work with** window, select **Add** option.

- fill in Name: with **GNU ARM Plug-ins**
- fill in Location
- **<Downloads Folder>/org.eclipse.cdt.cross.arm.gnu_0.5.5.201310221100**
- Click the **OK** button as shown in Figure-96.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

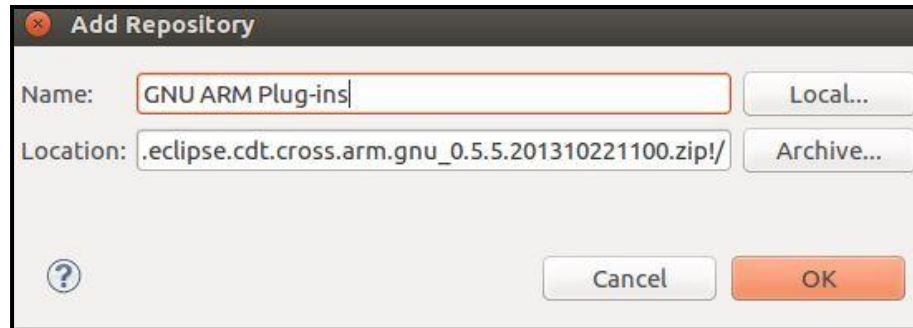


Figure-96: Add Repository from LocalSites

- Normally the main window should list a group of named GNU Cross Development Tools and expand it
- In case the main window display only message as “There are no categorized items”, you are probably using a very old version; disable the Group items by category option.
- Select the **GNU ARM C/C++ Development Support (End of Life)** as shown in Figure-97.

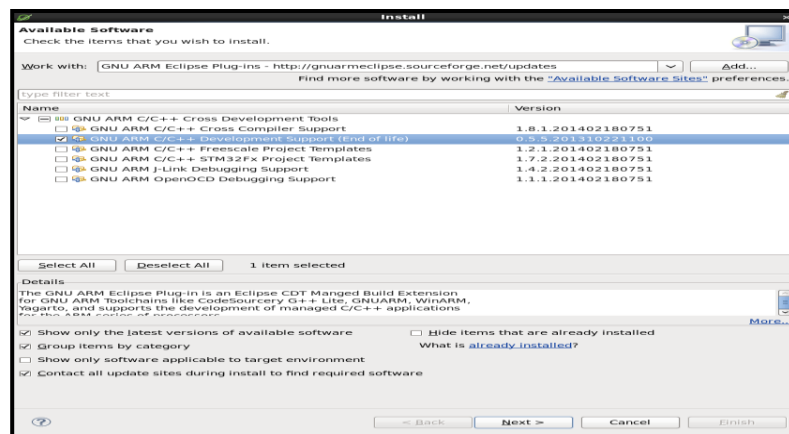


Figure-97: Selection GNU ARM C/C++ Development Support Plugins

- Click **Next**, and view the **Install Details** as shown in Figure-98, which tells you if any of the selected items are already present.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

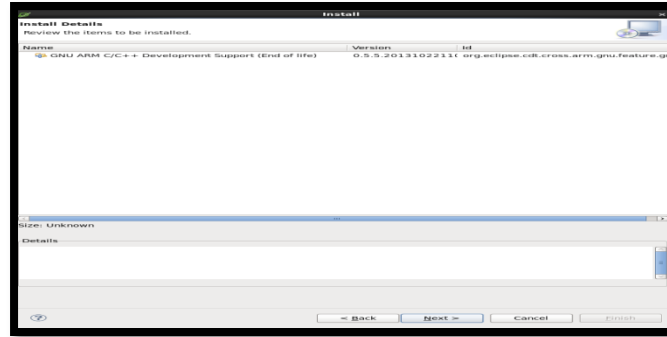


Figure-98: Install Details

- Click **Next**, then Accept the terms of the license agreement & click Finish as shown in Figure-99.

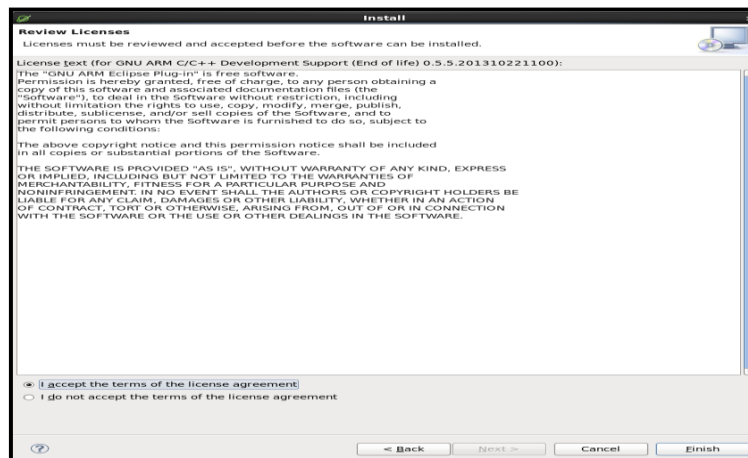


Figure-99: Accept the License

- Finally, Restart the AppCOE by click **Restart Now** from popup message box.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Installing Remote Debugging Capabilities

- Run **AppCOE** and from menu bar, select **Help -> Install New Software**.

In Install window, select **Add** button.

- fill in Name: Eclipse Juno Project
- Fill in Location: with <http://download.eclipse.org/releases/juno> as shown in Figure-100.

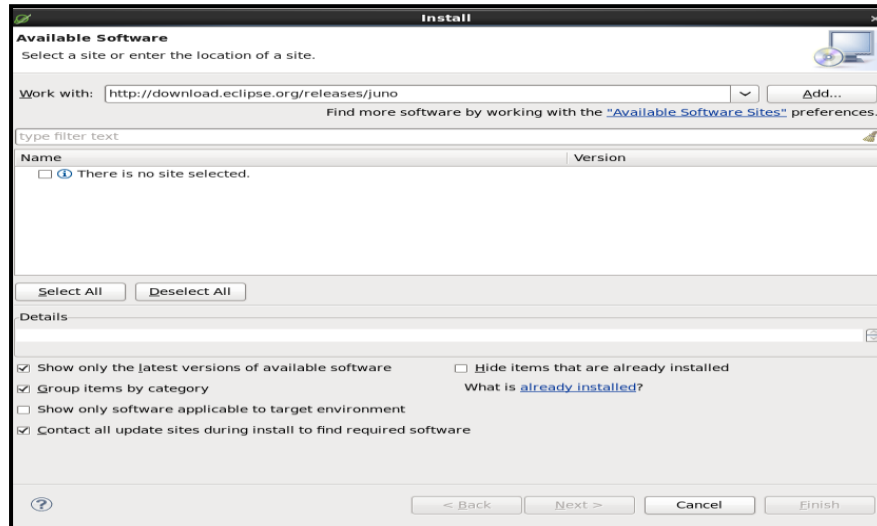


Figure-100: Add Repository into Available Software Window

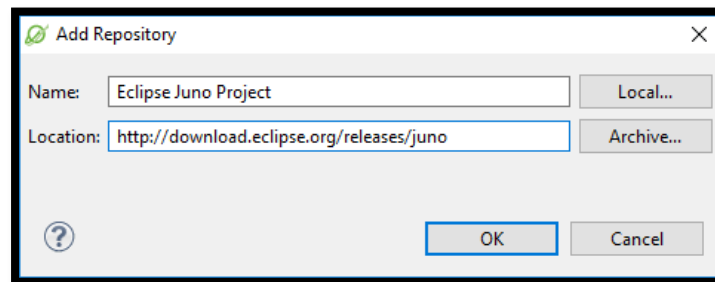


Figure-101: Path of remote login Plugins.

- Click the **OK** button and wait for a while to fetch all software from the eclipse repository.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

- Under **Name**, expand **Mobile and Device Development** and select these items as shown in Figure-102.
 - C/C++ GDB Hardware Debugger**
 - C/C++ Remote Launch**
 - Remote System Explorer End-User Runtime**
 - Remote System Explorer User Actions**

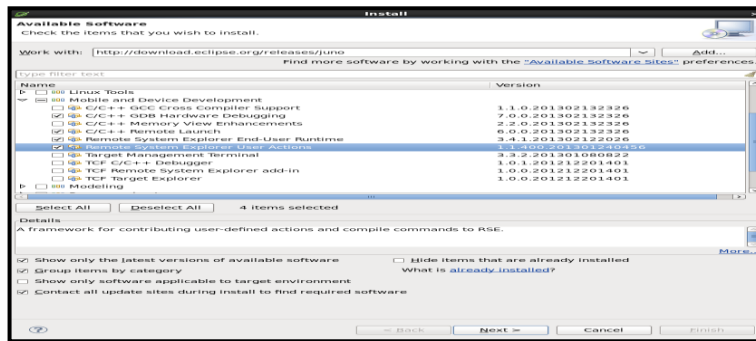


Figure-102: Selecting Mobile and Device Development Software Tool

- Click **Next**, and view the **Install Details**, which tells you if any of the selected items are already present as shown in Figure-103.

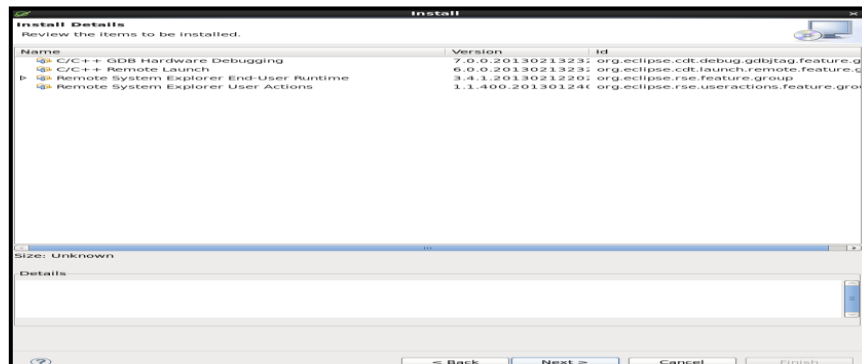


Figure-103: Install Details

- Click Next **Accept** the terms of the license agreement and click **Finish** as shown in Figure-104.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE



Figure-104: Accept a Licenses

- Now it can takes some time while the new software will be installed as shown in Figure-105.

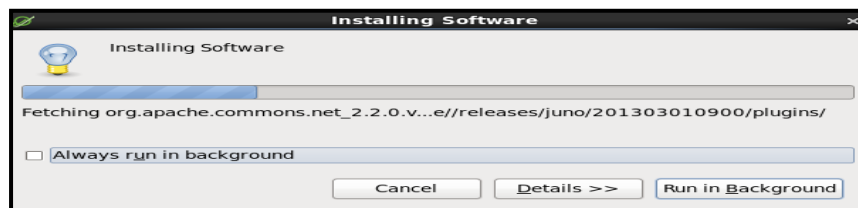


Figure-105: Installing Software

- After the software update you must restart AppCOE now by press the **Yes** button as shown in Figure-106.

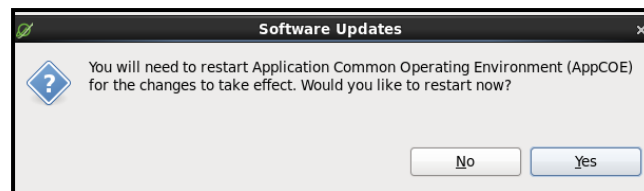


Figure-106: Restart Now

Press the "Yes" button and wait while AppCOE will be restarted, after the restart you will see the start screen again.

12. Importing the ARM Cross OS Target Template Application on AppCOE

This chapter contains the following topics:

About AppCOE

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Steps to import the ARM Cross OS Template Project on AppCOE

1. Copy the “raspberry-pi.zip” file from the Raspberry DVD Package given by MapuSoft Technologies and paste into AppCOE <InstallDir> directory.
2. Extract into same directory by using zip password (password sent by Mapusoft Technologies in mail), find the template_projects directory.
3. Find the template project for rpi_cos_template ,rpi_vxw_template & rpi_thx_template under template_projects directory
4. Open the AppCOE, Import a project under C/C++ Projects pane by right clicks & select **Import** as shown in Figure-107.

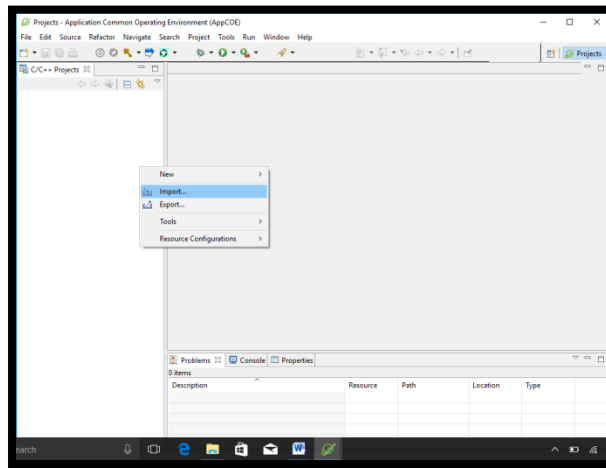


Figure-107: Import into C/C++ Project Pane

5. Select Existing Projects into Workspace as shown in Figure-108.

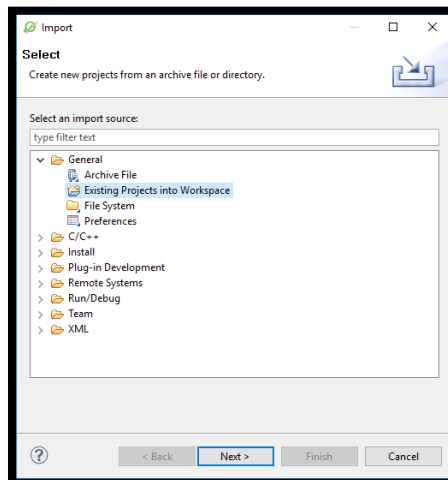


Figure-108: Import into C/C++ Project Pane

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

6. Import template project from extracted source as shown in Figure-109.

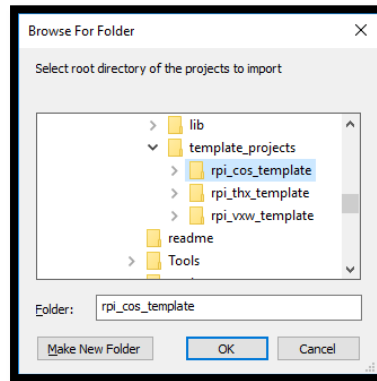


Figure-109: Import template project from Extracted source.

7. Select check box to enable “Copy projects into workspace”, so that the importing template projects will get copied into user workspace.

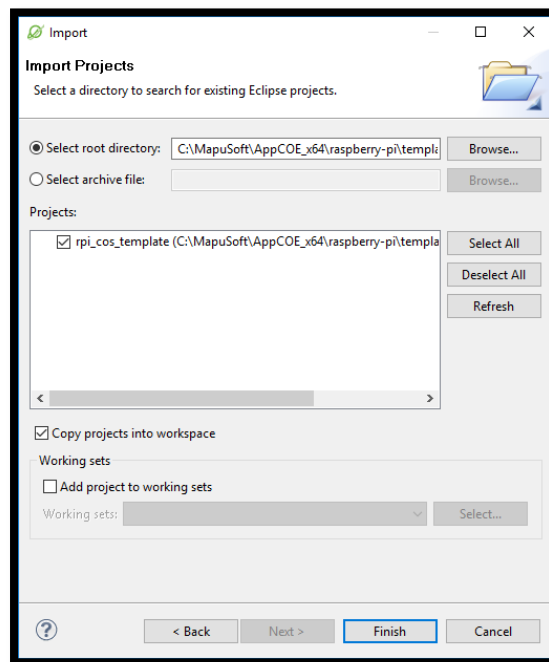


Figure-110: Copy template project into workspace

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

8. Select a template project under C/C++ Projects pane, right click and select **Build Project** as Build an Application project as shown in Figure-111.

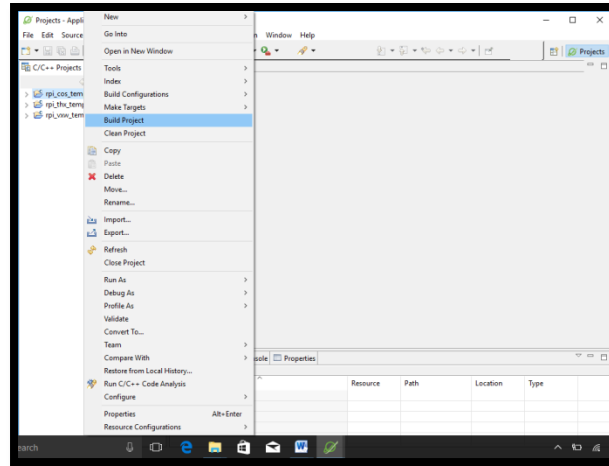


Figure-111: Building the ARM Cross Target Application Template Project

9. Let's follow the same steps for run & remote debug the ARM Cross Target Application Template Project like did for Application Project.

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Reference

- Introduction about Raspberry Pi can be found at <https://www.raspberrypi.org/>
- Installing Operating System Image into the Raspberry Pi hardware, do refer this link <https://www.raspberrypi.org/documentation/installation/installing-images/README.md>
<https://www.raspberrypi.org/documentation/installation/installing-images/windows.md>
- BCM2835 library (can be found at <https://github.com/jperkin/node-rpio/blob/master/src/bcm2835.c>)
- Wiring pi library (can be found at <https://git.drogon.net/?p=wiringPi;a=summary>

INTERFACE RASPBERRY-PI BOARD WITH APPCOE

Revision History

Document's Title: Interface Raspberry-pi Board with AppCOE

Release Number: 1.6.1

Release	Revision	Orig. of Change	Description of Change
1.6.1	1.0	VV	<ul style="list-style-type: none">Added support for the Raspberry-Pi B+ target Board with AppCOE libraries