K J Somaiya Institute of Management

Vidyavihar, Mumbai – 400 077

Master of Computer Applications

Trimester 4 (2021 – 23)

This is to certify that Mr. Ishan Sheth ***Roll No.*** 44 ***of MCA, has completed his*** *Internet of Things* ***Journal as per the syllabus for the academic year 2021–2023. The Journal has also been evaluated by the concerned faculty of KJ Somaiya Institute of Management.***

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Signature of the Faculty-In charge

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Signature of the Course Coordinator

Date: _ _ / _ _ / _ _ _ _

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Signature of the External Examiner

Index

Index **2**

Practical 1: CCS Installation, User Interface tour, Project creation, Compiling and debugging with MSP432 **4**

CCS Installation 4

MSP432 SDK Installation 7

Energia Installation with Board Manager details 9

Explain the front end main components of CCS IDE 14

Practical 2: Hello world of Embedded Systems **19**

Importing existing CCS project from Energia 19

Practical 3: Installing Node.js ,Node-Red and Node Red interface **24**

Installing Node.js 24

Installing Node-red 27

Launching node red 28

NodeRed Components 30

Practical 4: Introducing the inject, function, debug and switch nodes **32**

Inject Node 32

The Inject node can be used to manually trigger a flow by clicking the node's button within the editor. It can also be used to automatically trigger flows at regular intervals. 32

Function Node 32

Debug Node 33

Switch Node 33

Hello IOT Course 34

Multi Payload Function 34

Practical 5: Random number generator with selection of color. Introduce the HTTP node. 35

Random Number Function 35

Selection of Colors 35

HTTP-In and HTTP-Response Node 36

Two Wheelers and Four Wheelers WebPages 37

Practical 6: CPU utilization flow **40**

Installing node-red-contrib-cpu and node-red-dashboard 40

Flow of CPU usage 40

Practical 7: WiFi Setup **42**

Importing wifi.io file 42

Code Explanation 44

Checking the port on which the microcontroller MSP432 is connected and Configuring PuTTY 45

Detecting available devices 46

Practical 8: Analog To Digital Converter 47

Analog to Digital Converter 47
Code Explanation 48
API functions used configuring ADC 48
Readings from the Potentiometers 49

Practical 9: Sending Raw data to the cloud 50

Creating Instance 50
MQTT input node : SensorPOT 51
MQTT out node : MSP432LaunchPad 52
Flow between subscriber and publisher 52

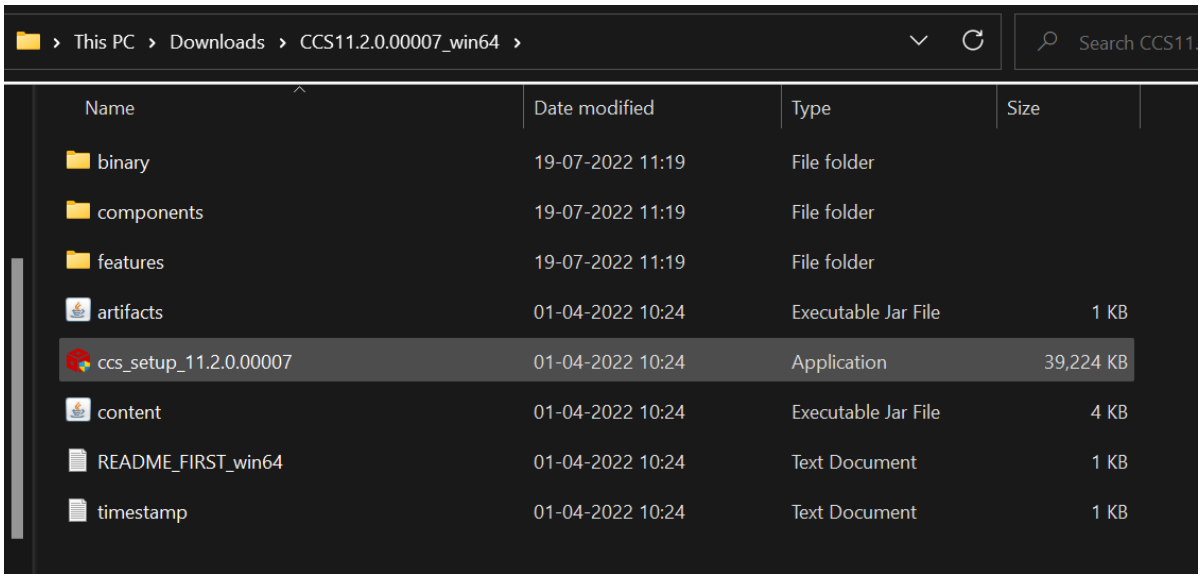
Practical 10: Connecting MSP432 to the cloud 53

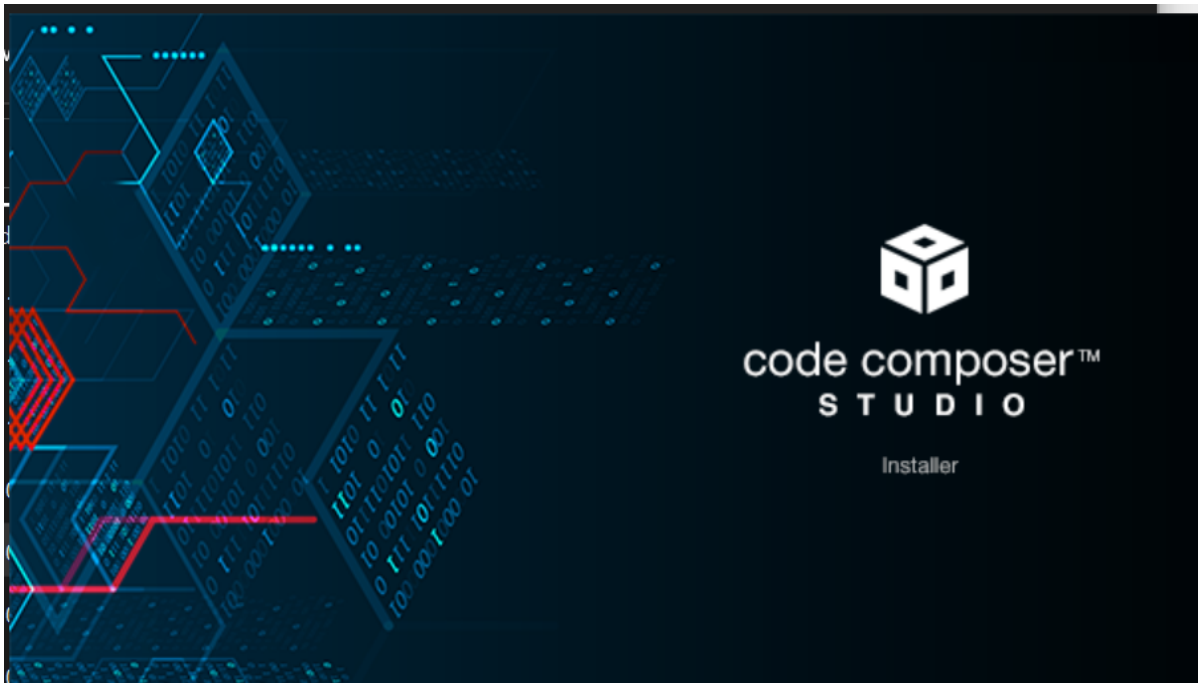
Code 53
Dashboard for Remotely Controlling LED 58
Using PuTTY emulator to record the LED changes 58

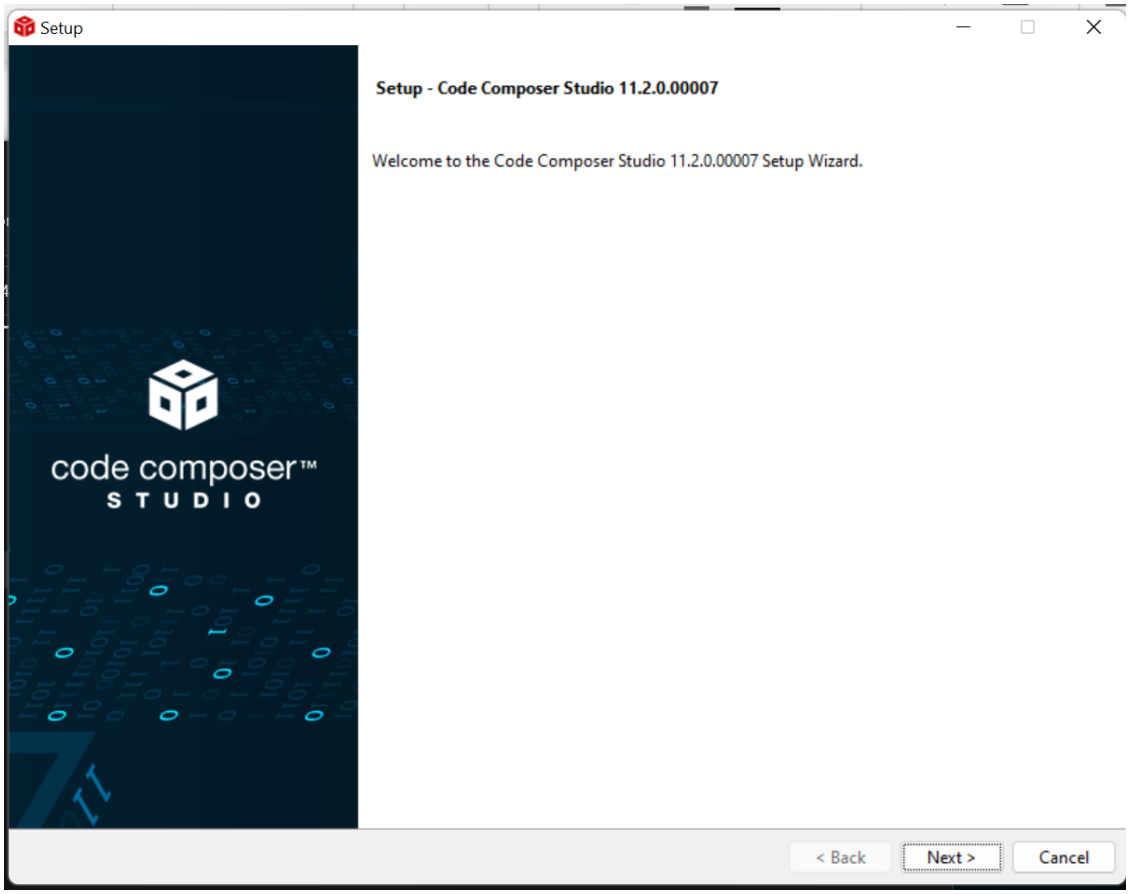
Practical 1: CCS Installation, User Interface tour, Project creation, Compiling and debugging with MSP432

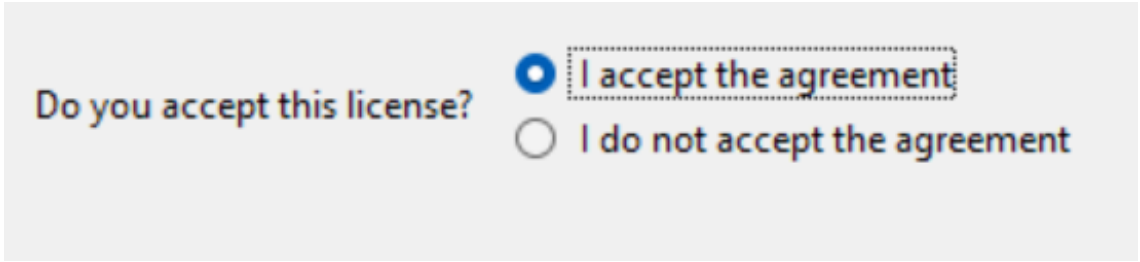
The following practical provides basic installation instructions for the Code Composer Studio IDE, MSP432 SDK, and Energia with board manager. The practical also focuses on describing CCS IDE's front end components.

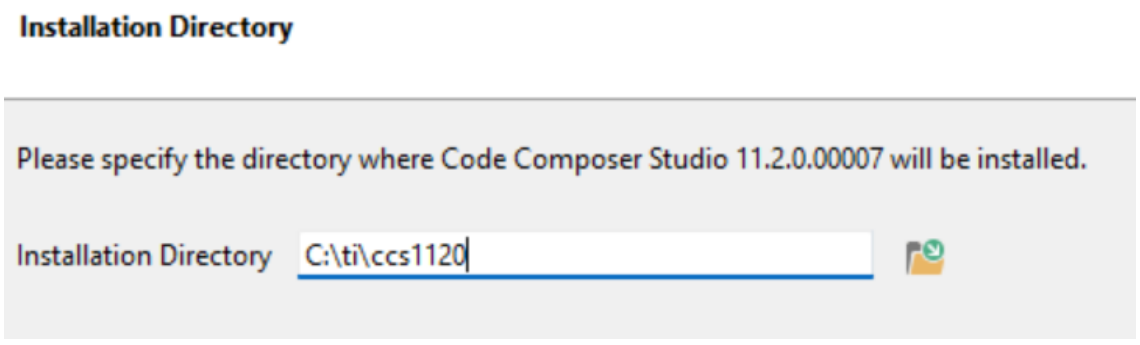
1. CCS Installation

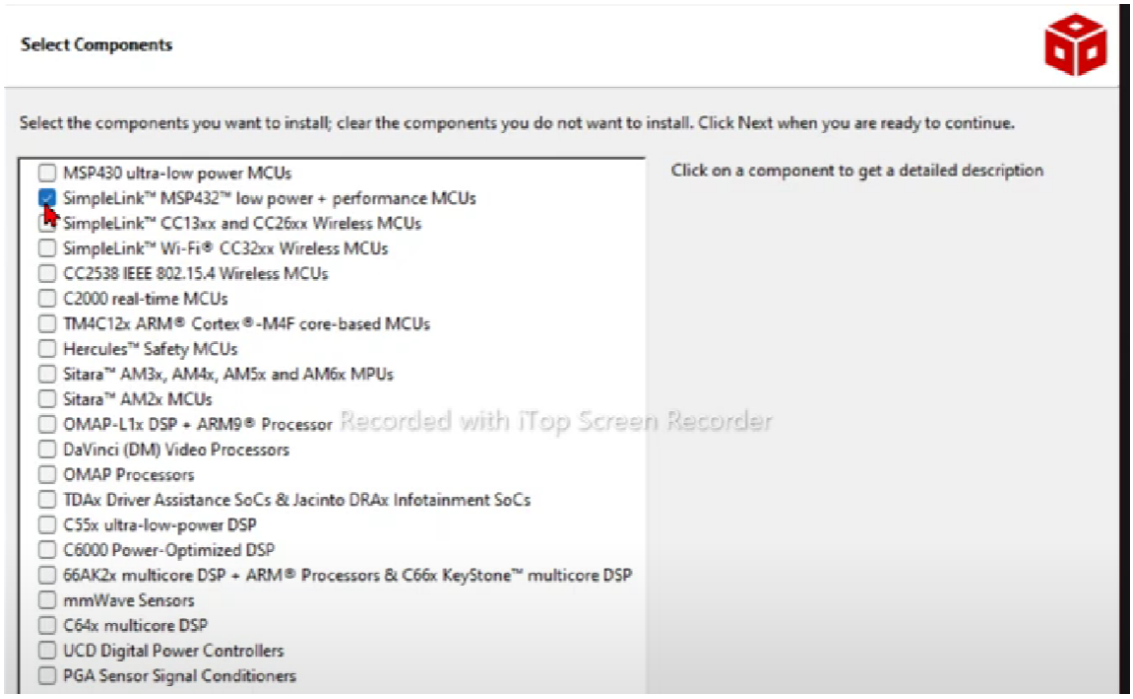


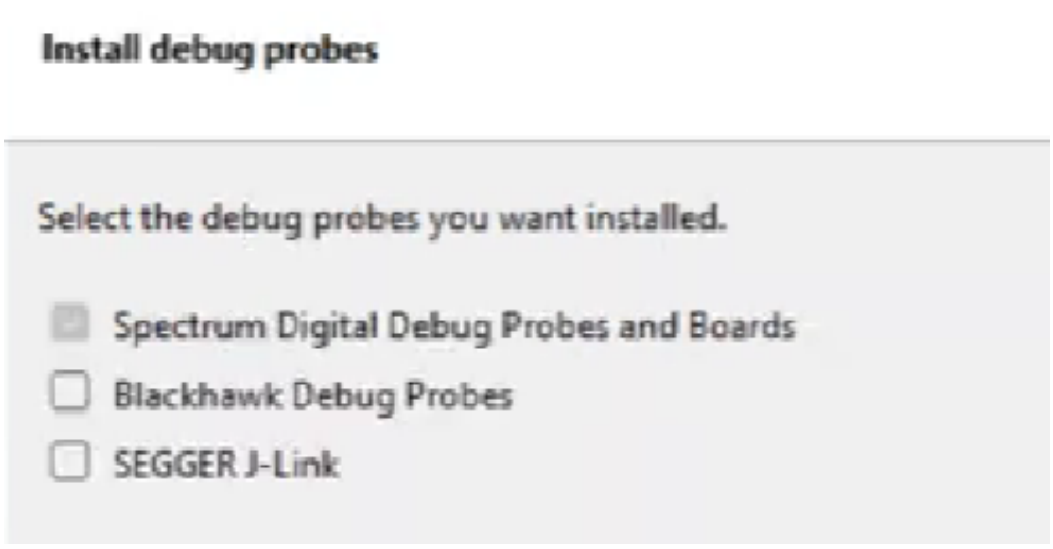


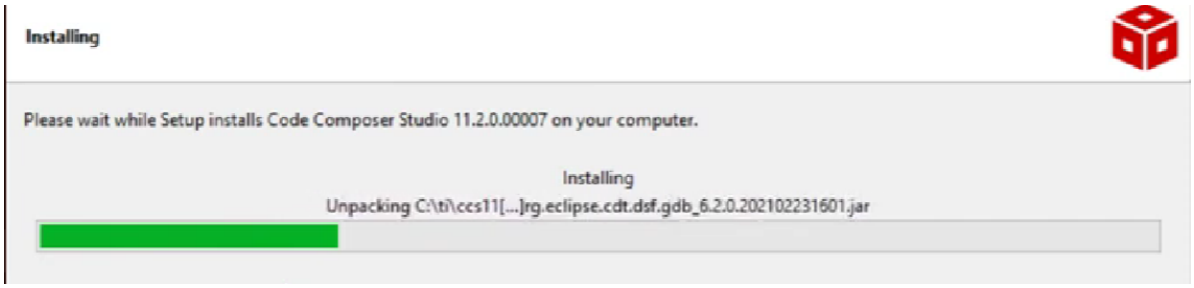

Setup
Setup - Code Composer Studio 11.2.0.00007
Welcome to the Code Composer Studio 11.2.0.00007 Setup Wizard.
code composer™
STUDIO
< Back
Next >
Cancel

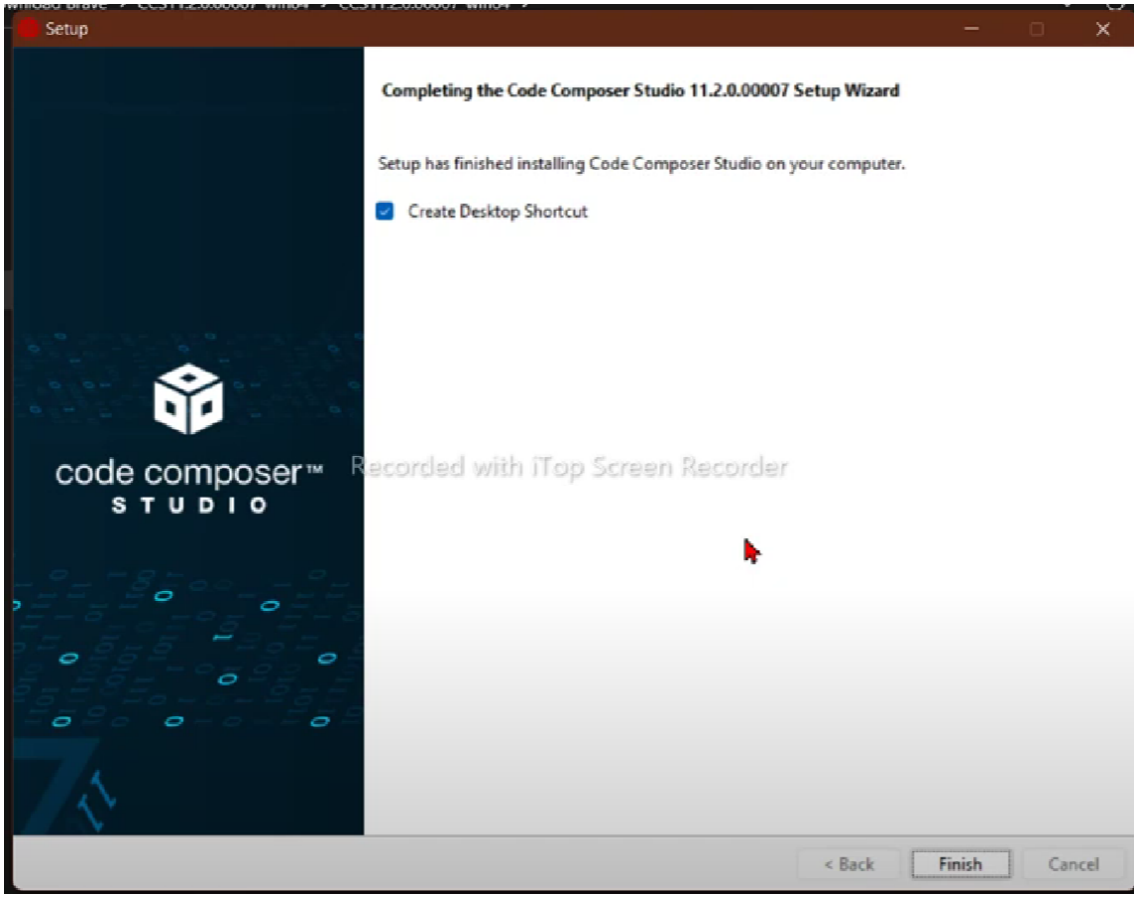

Do you accept this license?
I accept the agreement
I do not accept the agreement


Installation Directory
Please specify the directory where Code Composer Studio 11.2.0.00007 will be installed.
Installation Directory
C:\ti\ccs1120

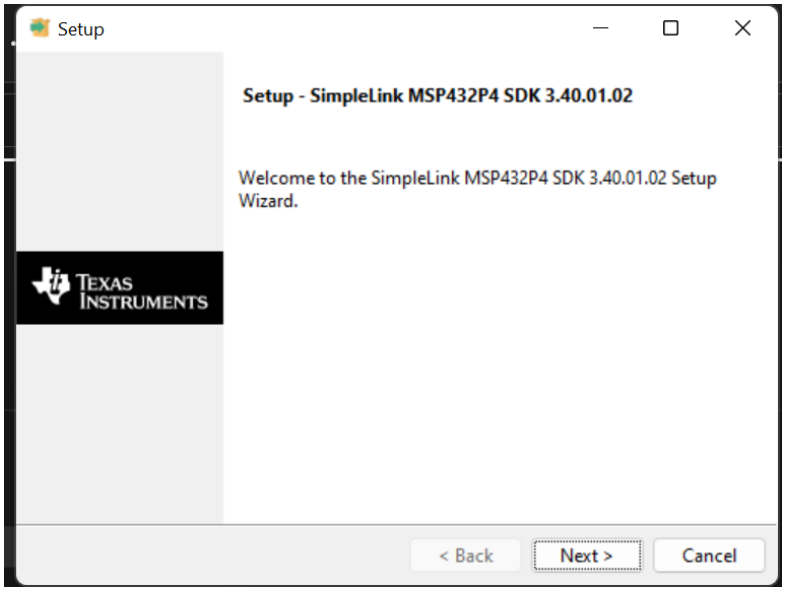

Select Components
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
MSP430 ultra-low power MCUs
SimpleLink™ MSP432™ low power + performance MCUs
SimpleLink™ CC13xx and CC26xx Wireless MCUs
SimpleLink™ Wi-Fi® CC32xx Wireless MCUs
CC2538 IEEE 802.15.4 Wireless MCUs
C2000 real-time MCUs
TM4C12x ARM® Cortex®-M4F core-based MCUs
Hercules™ Safety MCUs
Sitara™ AM3x, AM4x, AM5x and AM6x MPUs
Sitara™ AM2x MCUs
OMAP-L1x DSP + ARM9® Processor
DaVinci (DM) Video Processors
OMAP Processors
TDAx Driver Assistance SoCs & Jacinto DRAx Infotainment SoCs
C55x ultra-low-power DSP
C6000 Power-Optimized DSP
66AK2x multicore DSP + ARM® Processors & C66x KeyStone™ multicore DSP
mmWave Sensors
C64x multicore DSP
UCD Digital Power Controllers
PGA Sensor Signal Conditioners
Click on a component to get a detailed description
Recorded with iTop Screen Recorder

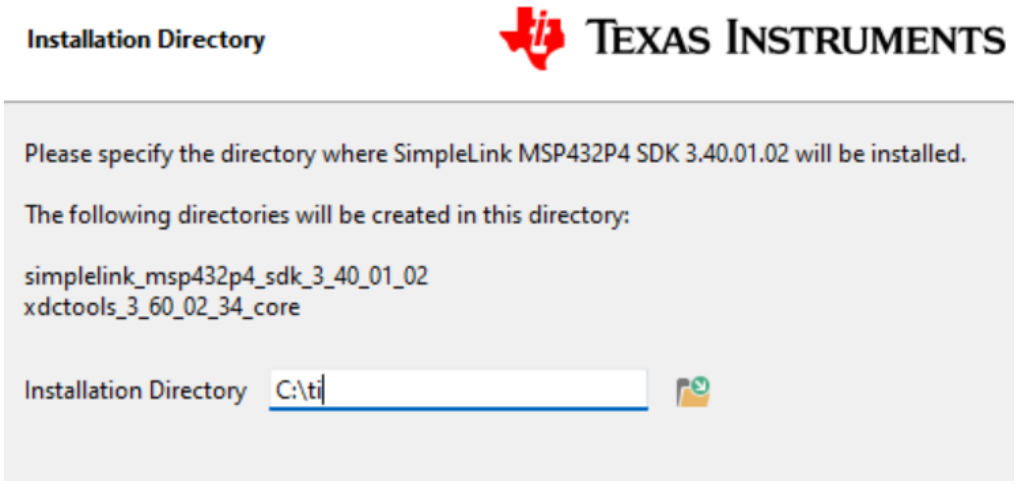

Install debug probes
Select the debug probes you want installed.
Spectrum Digital Debug Probes and Boards
Blackhawk Debug Probes
SEGGER J-Link


Installing
Please wait while Setup installs Code Composer Studio 11.2.0.00007 on your computer.
Installing
Unpacking C:\ti\ccs11[...]rg.eclipse.cdt.dsf.gdb_6.2.0.202102231601.jar

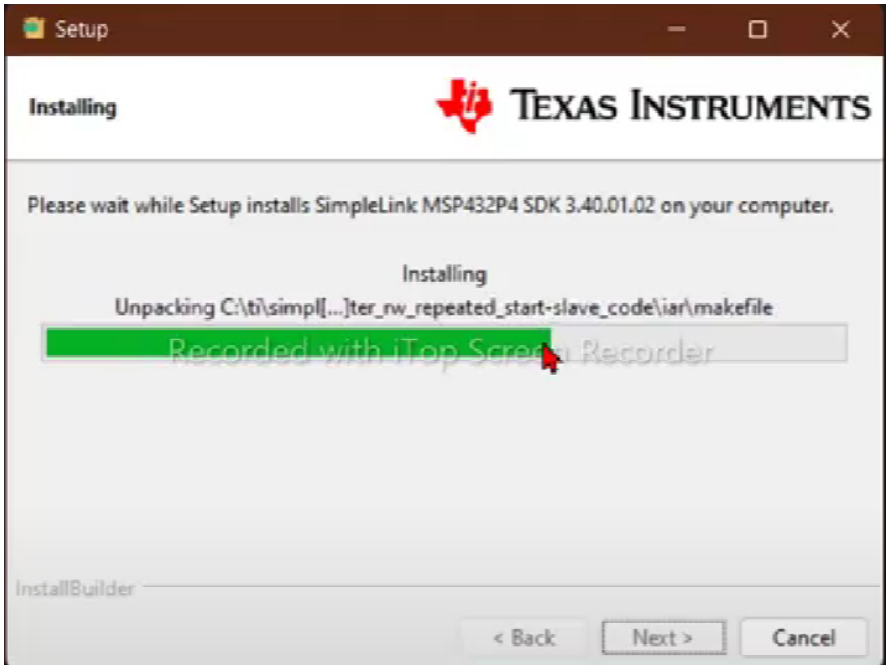


2. MSP432 SDK Installation

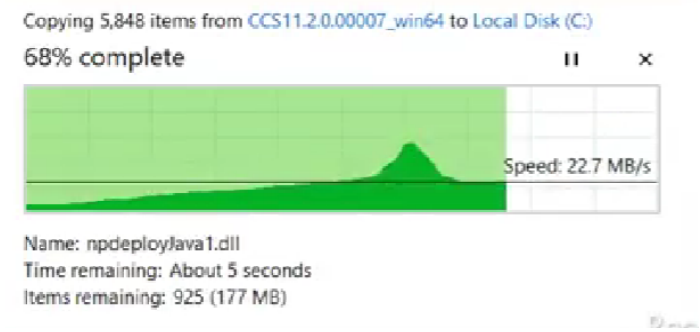


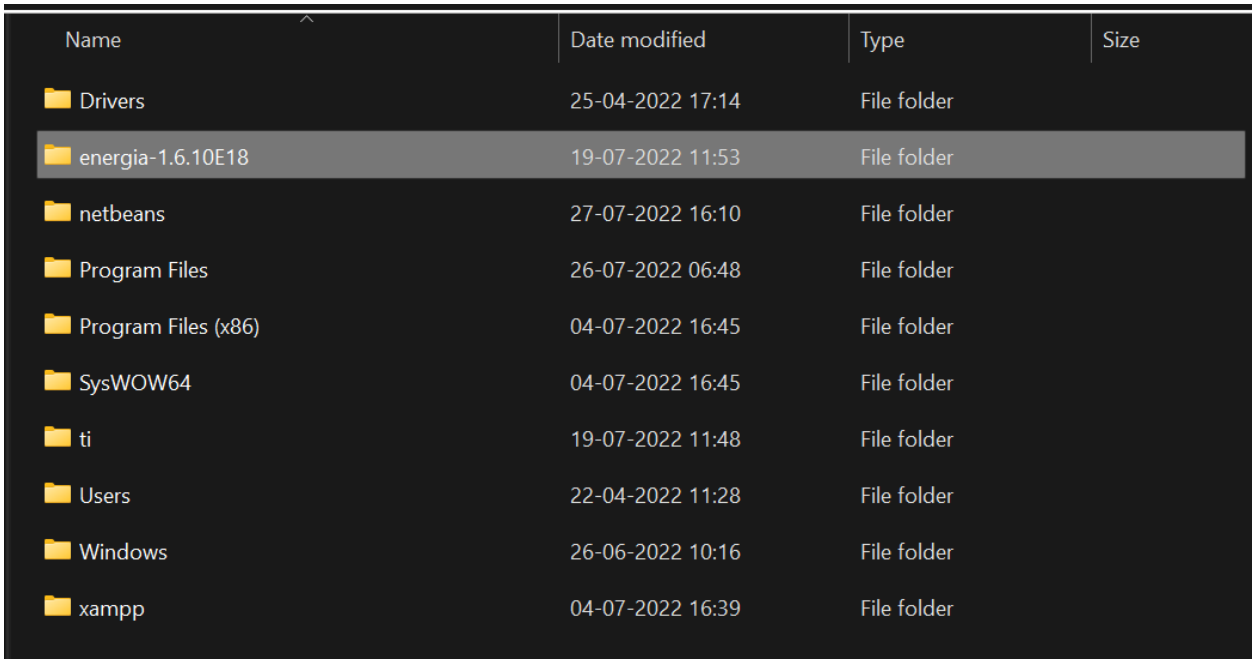

Installation Directory
TEXAS INSTRUMENTS
Please specify the directory where SimpleLink MSP432P4 SDK 3.40.01.02 will be installed.
The following directories will be created in this directory:
simplelink_msp432p4_sdk_3_40_01_02
xdctools_3_60_02_34_core
Installation Directory
C:\ti

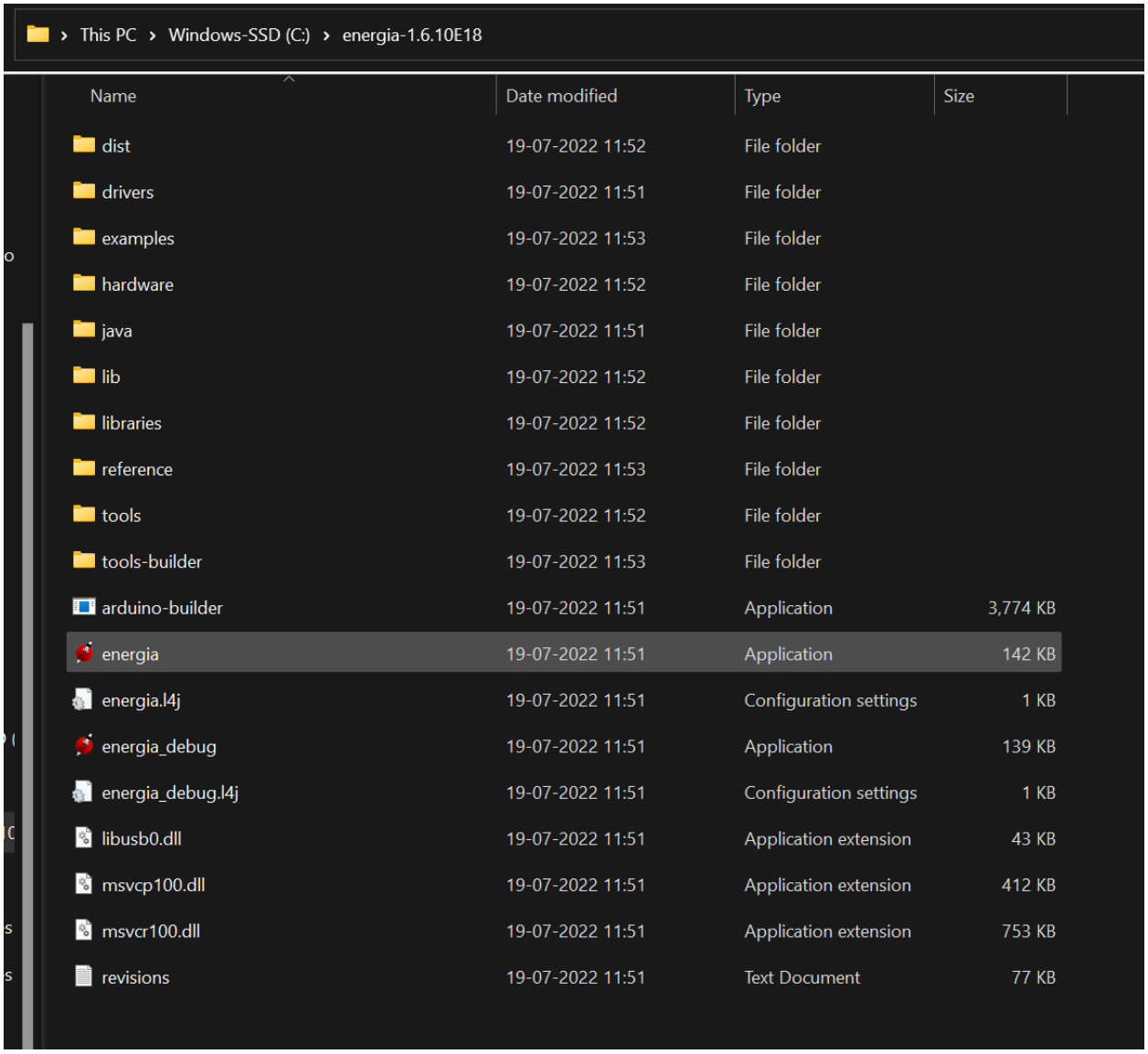

Ready to Install
TEXAS INSTRUMENTS
Setup is now ready to begin installing SimpleLink MSP432P4 SDK 3.40.01.02 on your computer.

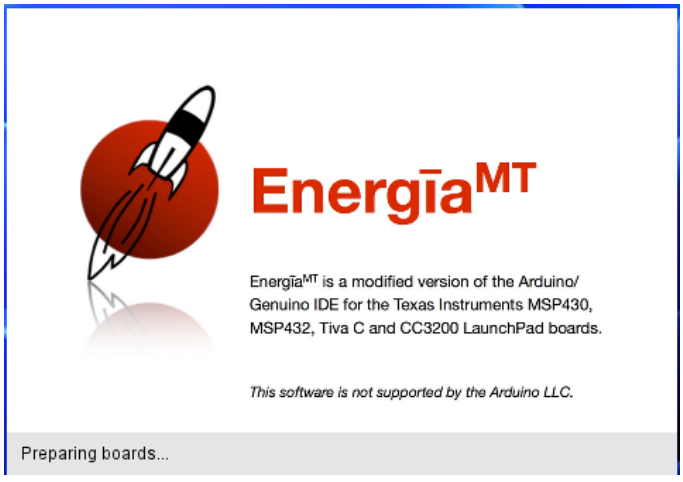

Setup
Installing
TEXAS INSTRUMENTS
Please wait while Setup installs SimpleLink MSP432P4 SDK 3.40.01.02 on your computer.
Installing
Unpacking C:\ti\simpl[...]ter_rw_repeated_start-slave_code\iar\makefile
Recorded with iTop Screen Recorder
InstallBuilder
< Back
Next >
Cancel

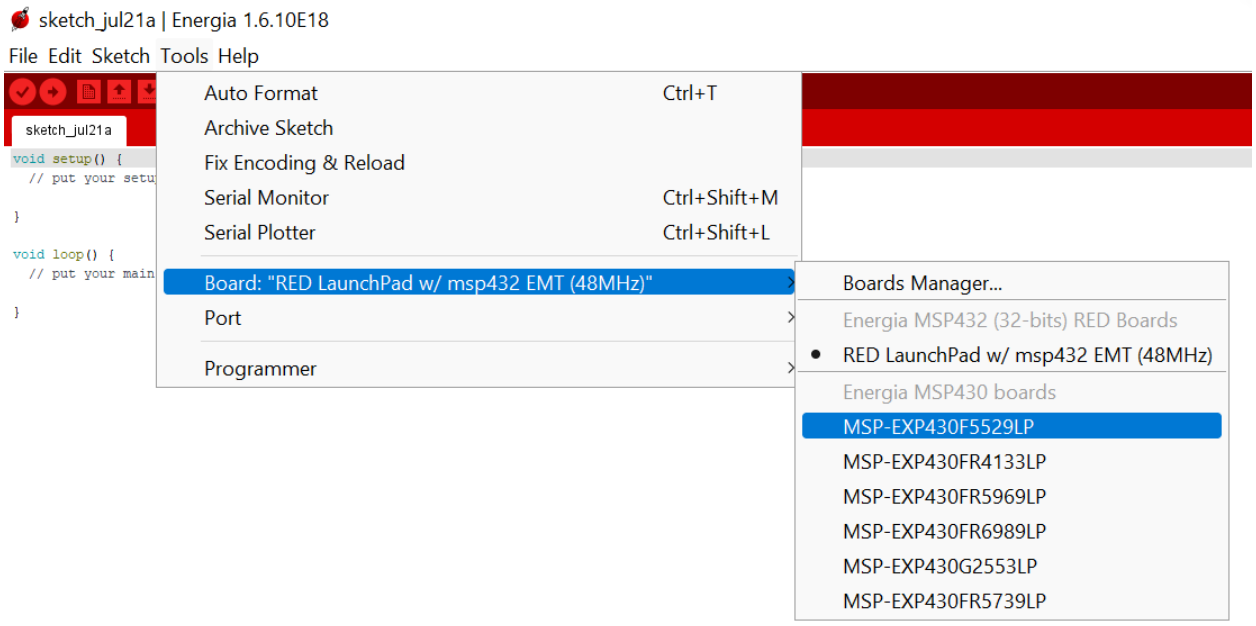
3. Energia Installation with Board Manager details

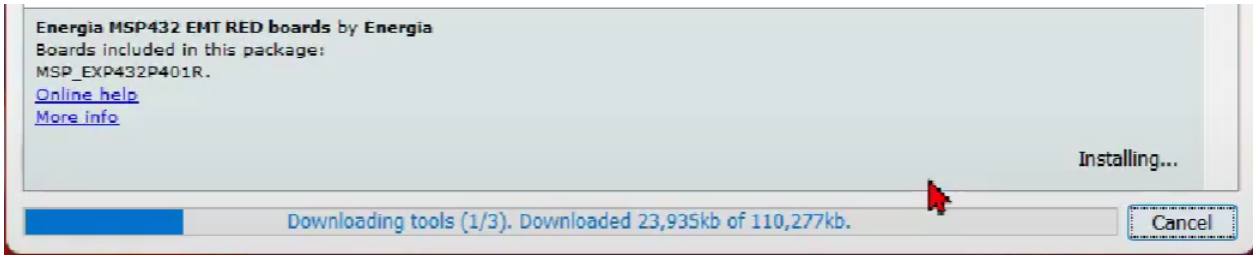


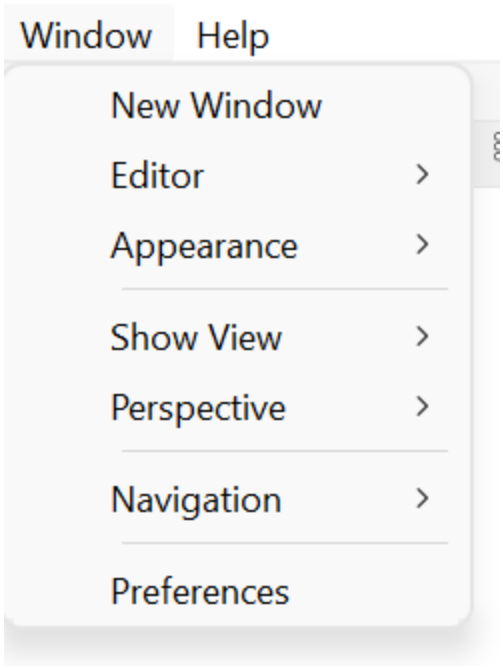


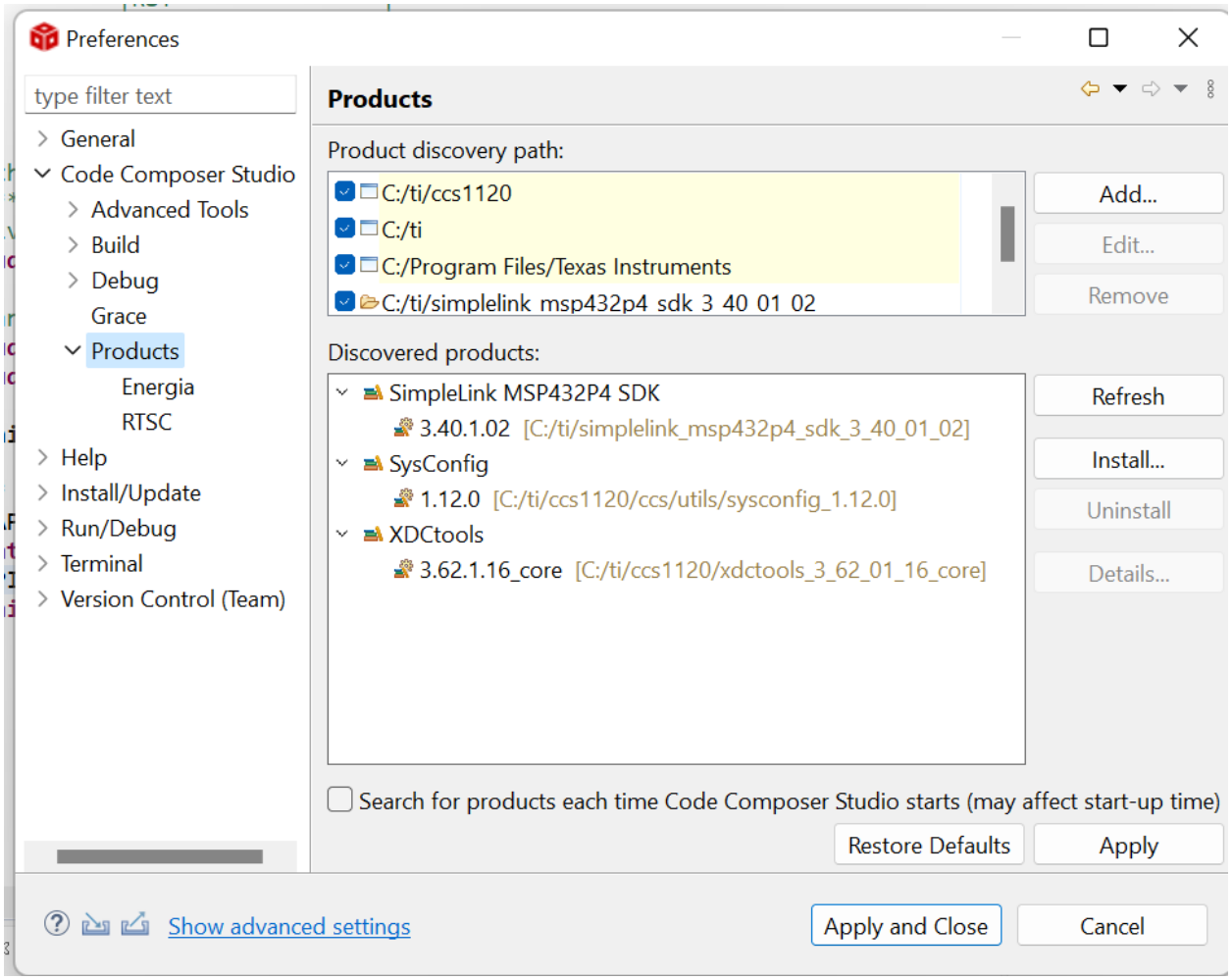

This PC › Windows-SSD (C:) › energia-1.6.10E18
Name
Date modified
Type
Size
dist 19-07-2022 11:52 File folder
drivers 19-07-2022 11:51 File folder
examples 19-07-2022 11:53 File folder
hardware 19-07-2022 11:52 File folder
java 19-07-2022 11:51 File folder
lib 19-07-2022 11:52 File folder
libraries 19-07-2022 11:52 File folder
reference 19-07-2022 11:53 File folder
tools 19-07-2022 11:52 File folder
tools-builder 19-07-2022 11:53 File folder
arduino-builder 19-07-2022 11:51 Application 3,774 KB
energia 19-07-2022 11:51 Application 142 KB
energia.l4j 19-07-2022 11:51 Configuration settings 1 KB
energia_debug 19-07-2022 11:51 Application 139 KB
energia_debug.l4j 19-07-2022 11:51 Configuration settings 1 KB
libusb0.dll 19-07-2022 11:51 Application extension 43 KB
msvcp100.dll 19-07-2022 11:51 Application extension 412 KB
msvcr100.dll 19-07-2022 11:51 Application extension 753 KB
revisions 19-07-2022 11:51 Text Document 77 KB

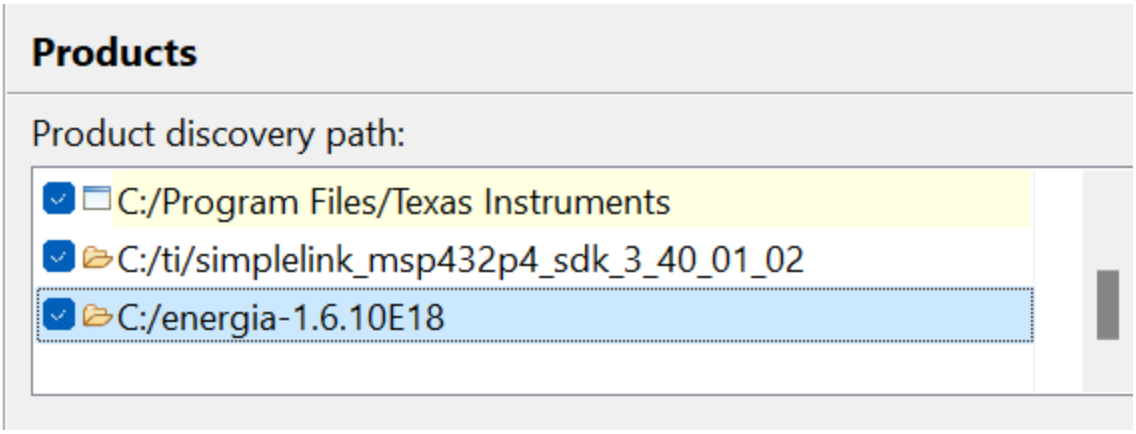

EnergīaMT
EnergīaMT is a modified version of the Arduino/ Genuino IDE for the Texas Instruments MSP430, MSP432, Tiva C and CC3200 LaunchPad boards.
This software is not supported by the Arduino LLC.
Preparing boards...

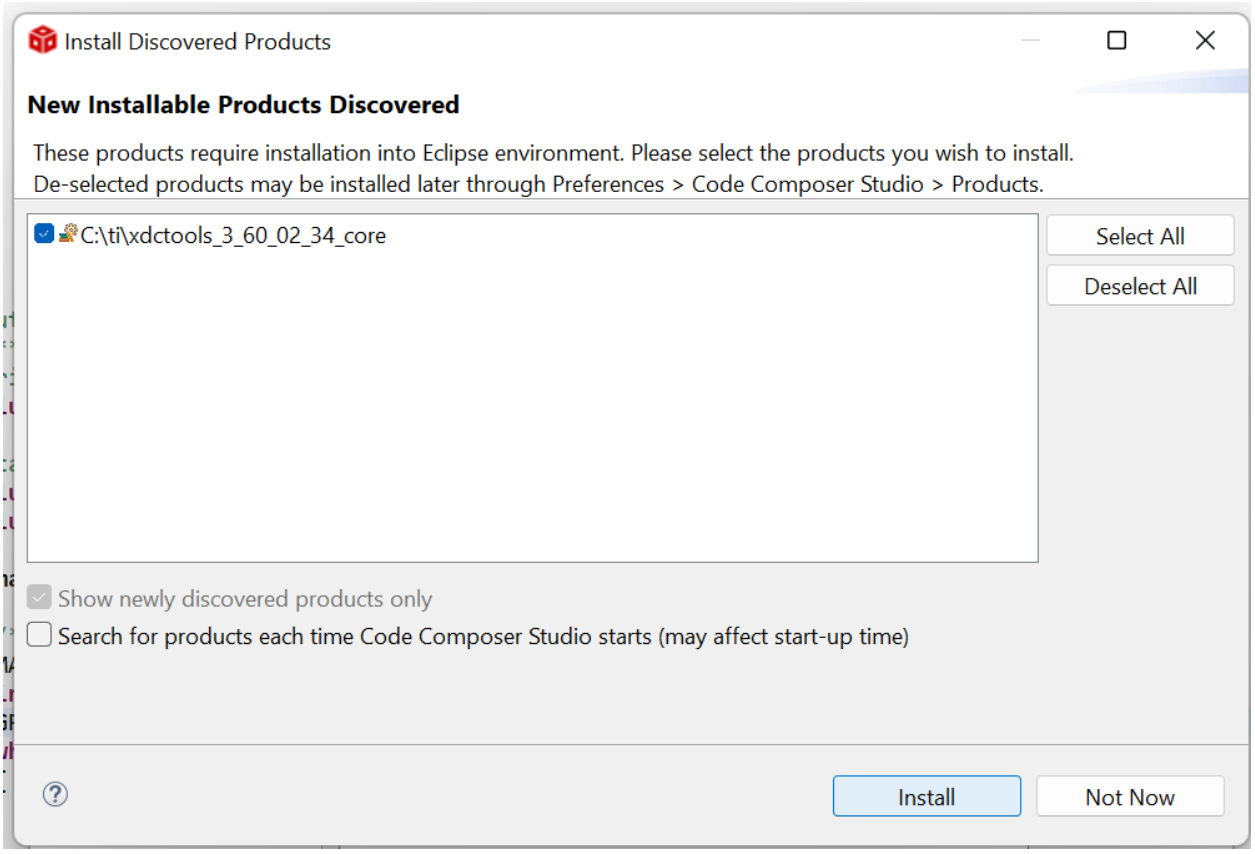

sketch_jul21a | Energia 1.6.10E18
File Edit Sketch Tools Help
sketch_jul21a
void setup() {
// put your setu
}
void loop() {
// put your main
}
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
Board: "RED LaunchPad w/ msp432 EMT (48MHz)"
Port
Programmer
Boards Manager...
Energia MSP432 (32-bits) RED Boards
RED LaunchPad w/ msp432 EMT (48MHz)
Energia MSP430 boards
MSP-EXP430F5529LP
MSP-EXP430FR4133LP
MSP-EXP430FR5969LP
MSP-EXP430FR6989LP
MSP-EXP430G2553LP
MSP-EXP430FR5739LP

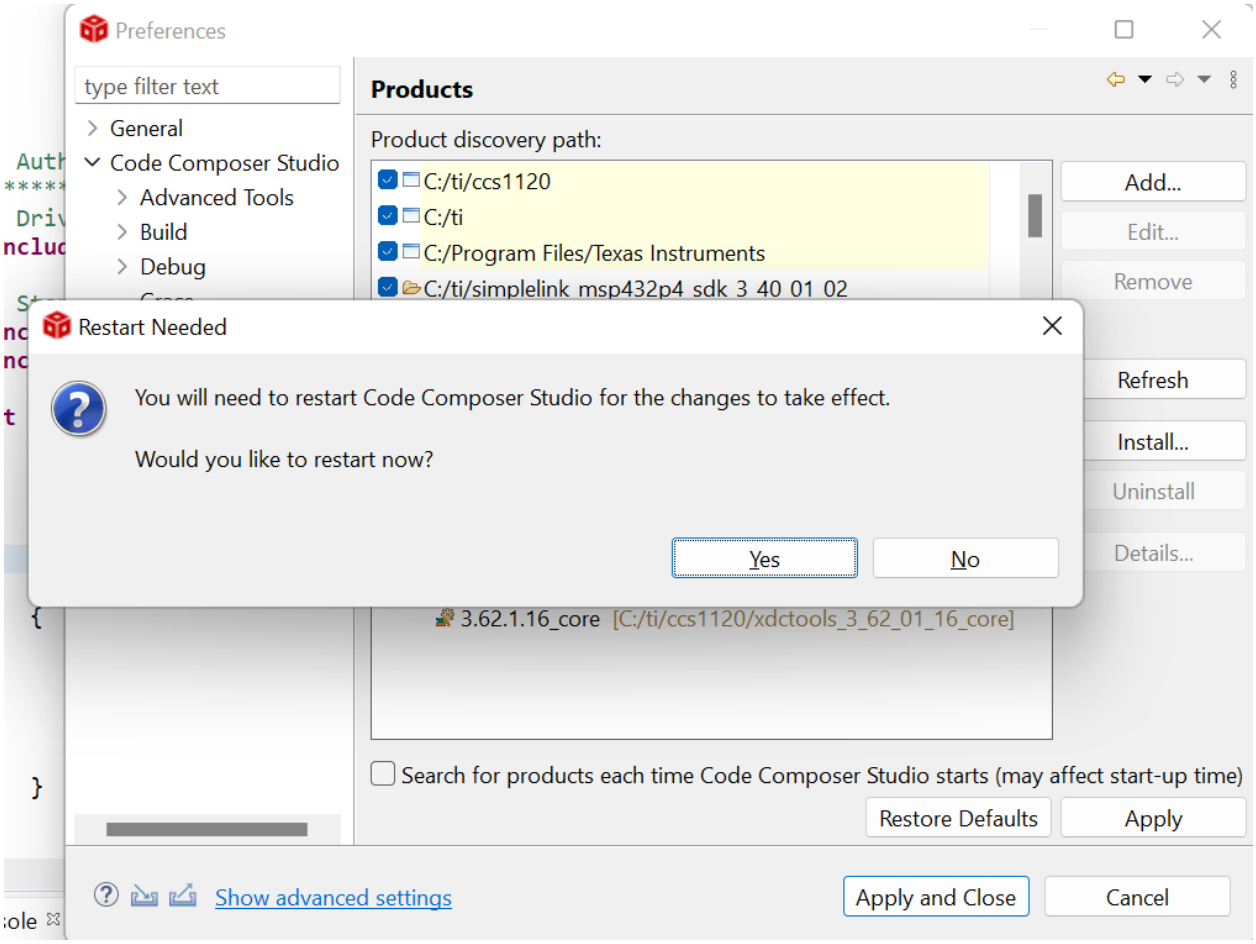

Energia MSP432 EMT RED boards by Energia
Boards included in this package:
MSP_EXP432P401R.
Online help
More info
Installing...
Downloading tools (1/3). Downloaded 23,935kb of 110,277kb.
Cancel


Window
Help
New Window
Editor
Appearance
Show View
Perspective
Navigation
Preferences


Preferences
type filter text
General
Code Composer Studio
Advanced Tools
Build
Debug
Grace
Products
Energia
RTSC
Help
Install/Update
Run/Debug
Terminal
Version Control (Team)
Products
Product discovery path:
C:/ti/ccs1120
C:/ti
C:/Program Files/Texas Instruments
C:/ti/simplelink_msp432p4_sdk_3_40_01_02
Add...
Edit...
Remove
Discovered products:
SimpleLink MSP432P4 SDK
3.40.1.02 [C:/ti/simplelink_msp432p4_sdk_3_40_01_02]
SysConfig
1.12.0 [C:/ti/ccs1120/ccs/utils/sysconfig_1.12.0]
XDCtools
3.62.1.16_core [C:/ti/ccs1120/xdctools_3_62_01_16_core]
Refresh
Install...
Uninstall
Details...
Search for products each time Code Composer Studio starts (may affect start-up time)
Restore Defaults
Apply
Show advanced settings
Apply and Close
Cancel

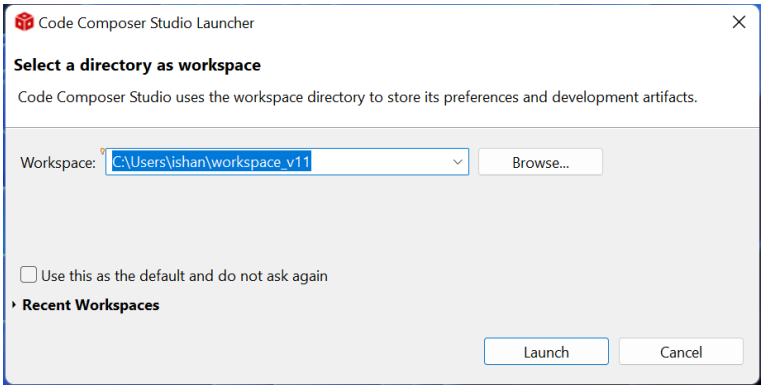

Products
Product discovery path:
C:/Program Files/Texas Instruments
C:/ti/simplelink_msp432p4_sdk_3_40_01_02
C:/energia-1.6.10E18

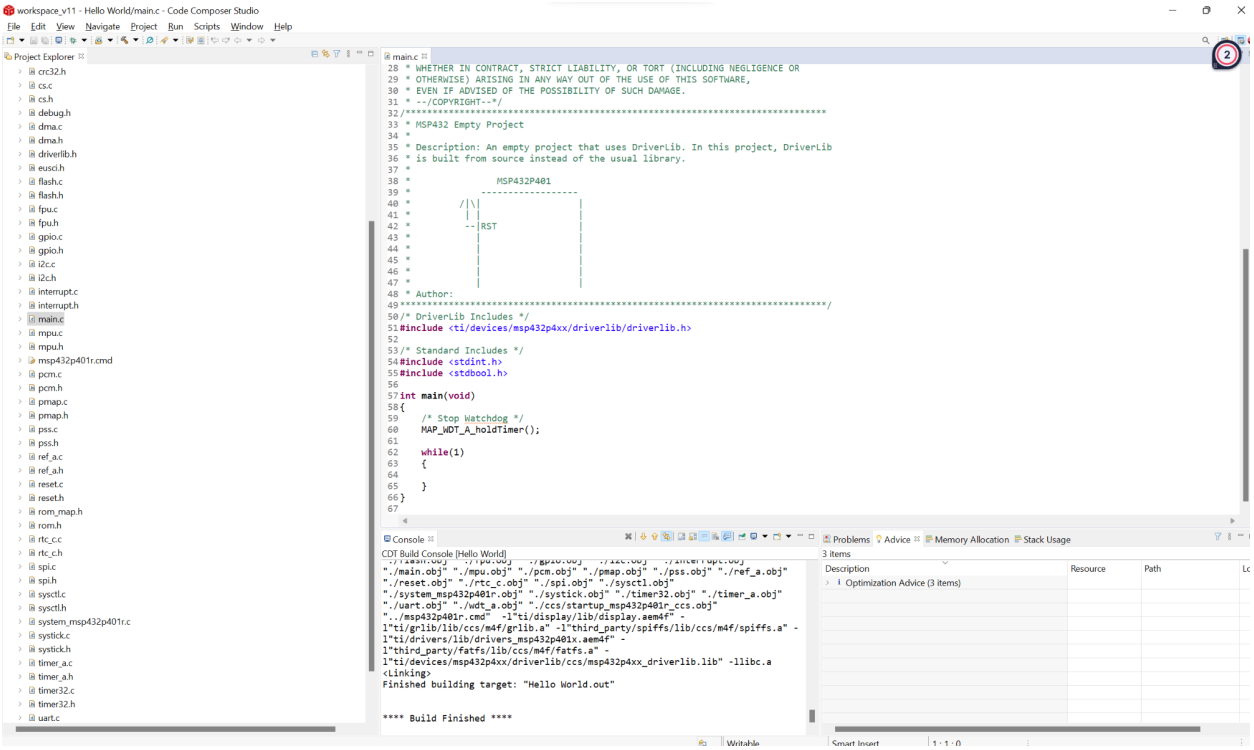

Install Discovered Products
New Installable Products Discovered
These products require installation into Eclipse environment. Please select the products you wish to install.
De-selected products may be installed later through Preferences > Code Composer Studio > Products.
C:\ti\xdctools_3_60_02_34_core
Select All
Deselect All
Show newly discovered products only
Search for products each time Code Composer Studio starts (may affect start-up time)
Install
Not Now



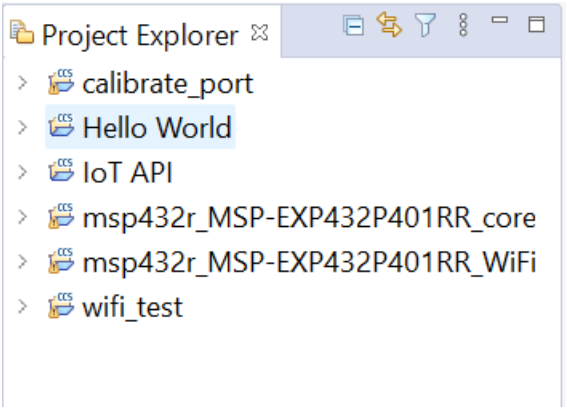
4. Explain the front end main components of CCS IDE

Code Composer Studio (CCS) is an IDE to develop software for microcontroller like MSP430,MSP432,etc.CCS is based on Eclipse platform and supports a plug and play architecture in which multiple compilers/debuggers can be used for developing and debugging software.



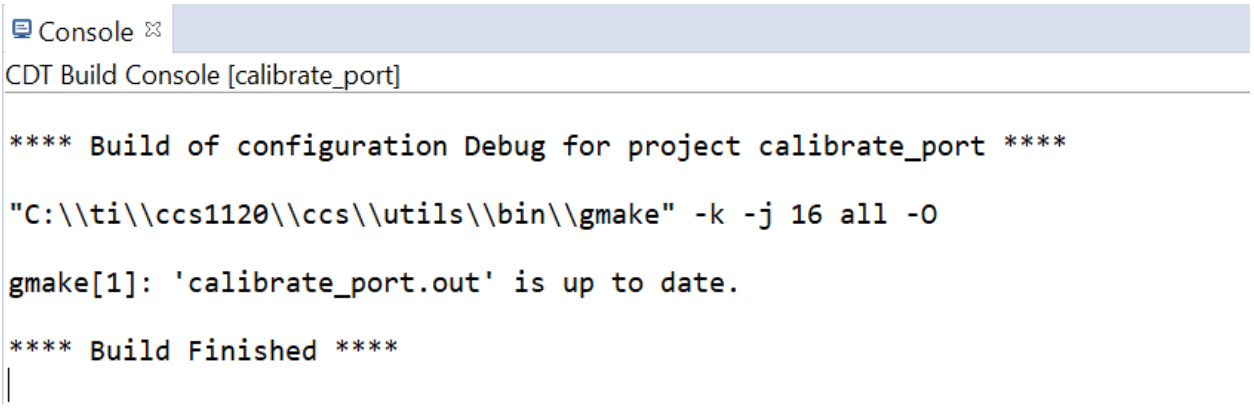


1. **Project Explorer window**

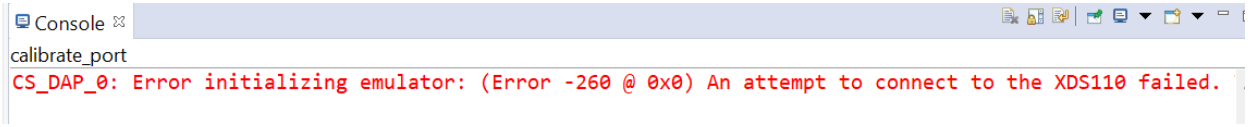


The **Project Explorer view** displays all projects that are part of the active workspace.The workspace will make a reference to the project after it is created, and the project will then be viewable and usable from the **Project Explorer view.** The name of each project in the workspace must be distinct and hence it needs to be renamed after its creation.

2. **Console View**



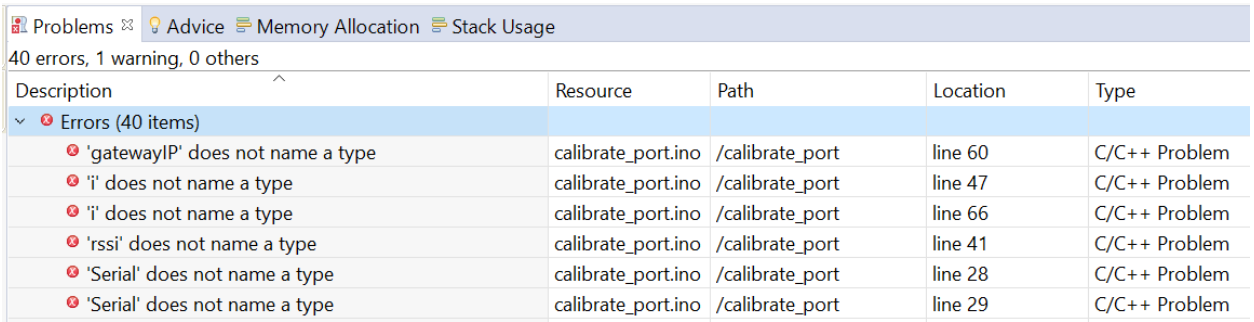
The **Console view** shows the standard and error outputs of the build tools during build time. The output for the active project chosen in the Project Explorer view will be displayed in the Console view while building multiple projects.



3. Problems View

The Problems view displays a summary of all errors and warnings encountered during the project build. There will be links in certain messages that can be clicked to access additional diagnostic information. The view contains the following information in the columns:

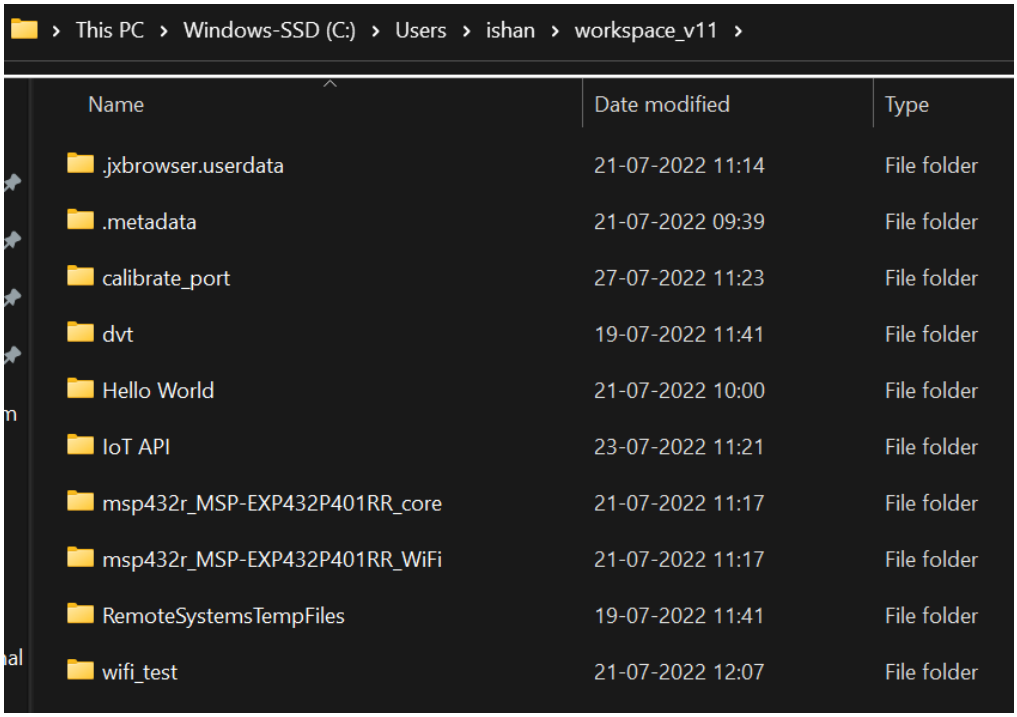
- **Description**: Description of the error.
- **Resource**: Resource/file where the error occurred.
- **Path**: Path to the resource that generated the error.
- **Location**: Location/Line in the resource that generated the error.
- **Type**: Type of error.



Description	Resource	Path	Location	Type
Errors (40 items)				
'gatewayIP' does not name a type	calibrate_port.ino	/calibrate_port	line 60	C/C++ Problem
'i' does not name a type	calibrate_port.ino	/calibrate_port	line 47	C/C++ Problem
'i' does not name a type	calibrate_port.ino	/calibrate_port	line 66	C/C++ Problem
'rssi' does not name a type	calibrate_port.ino	/calibrate_port	line 41	C/C++ Problem
'Serial' does not name a type	calibrate_port.ino	/calibrate_port	line 28	C/C++ Problem
'Serial' does not name a type	calibrate_port.ino	/calibrate_port	line 29	C/C++ Problem

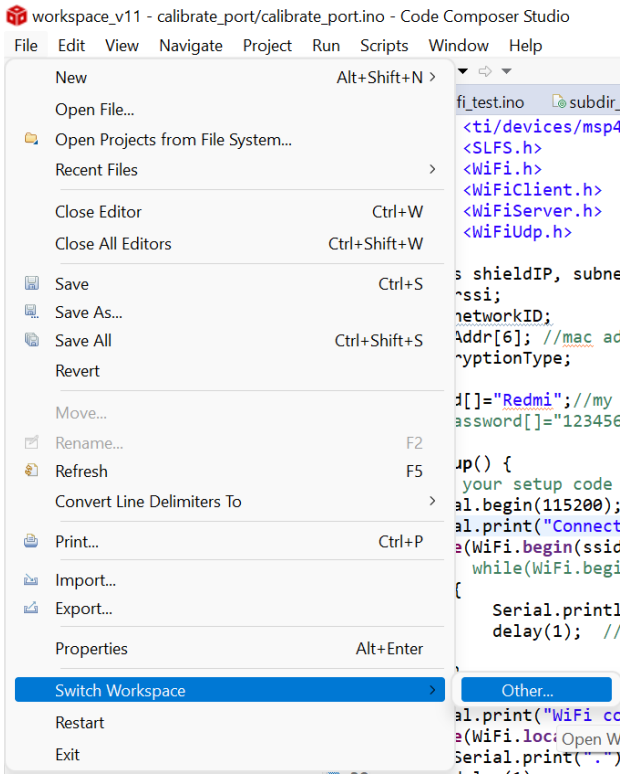
4. Workspaces

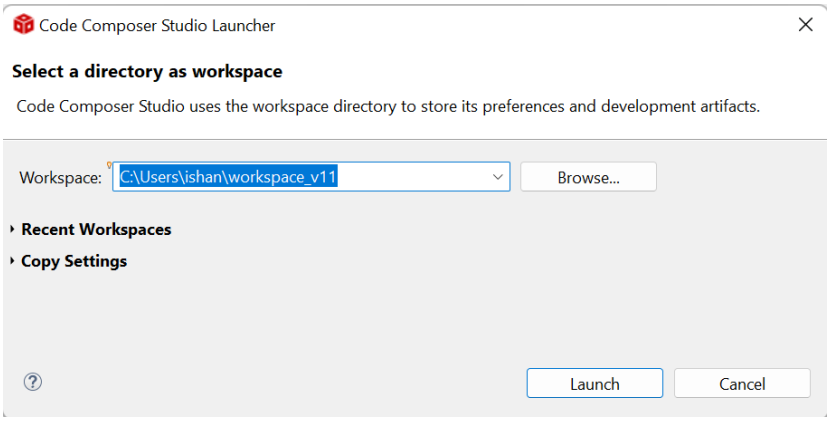
The main working folder for CCS is called a workspace, and it has data for managing all the projects assigned to it. CCS will ask for the location of the workspace folder when it is run. You can choose to use the chosen folder as the default folder to avoid being prompted in the future. The CCS Edit viewpoint will be automatically available after it has launched. The Project Explorer, Editor, and Problems views, which are frequently used during code development, are all included in this perspective.



Switching between workspaces

It is possible to have multiple workspaces. Only one workspace is active at a time in Code Composer Studio but you can switch workspaces using the menu File → Switch Workspace

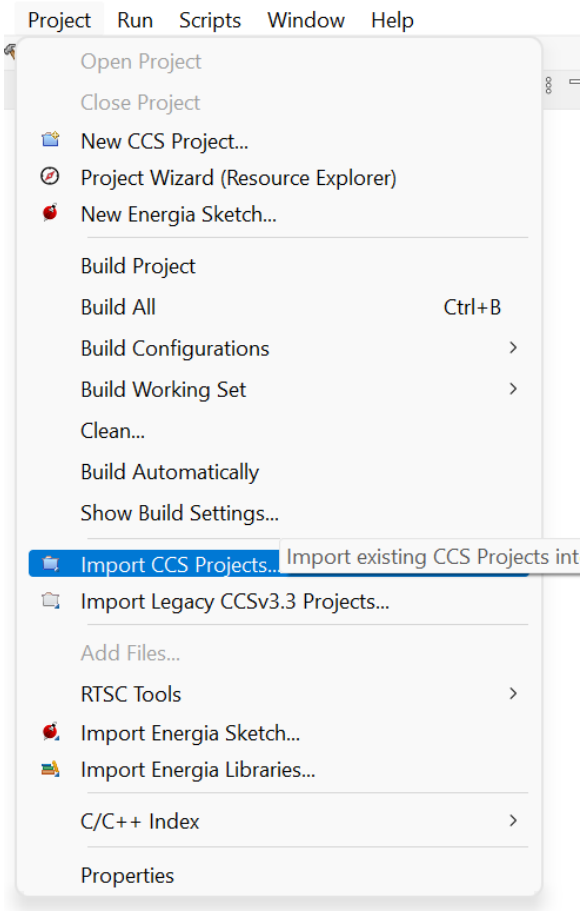

workspace_v11 - calibrate_port/calibrate_port.ino - Code Composer Studio
File Edit View Navigate Project Run Scripts Window Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Close Editor Ctrl+W
Close All Editors Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Import...
Export...
Properties Alt+Enter
Switch Workspace
Restart
Exit
Other...
Open W

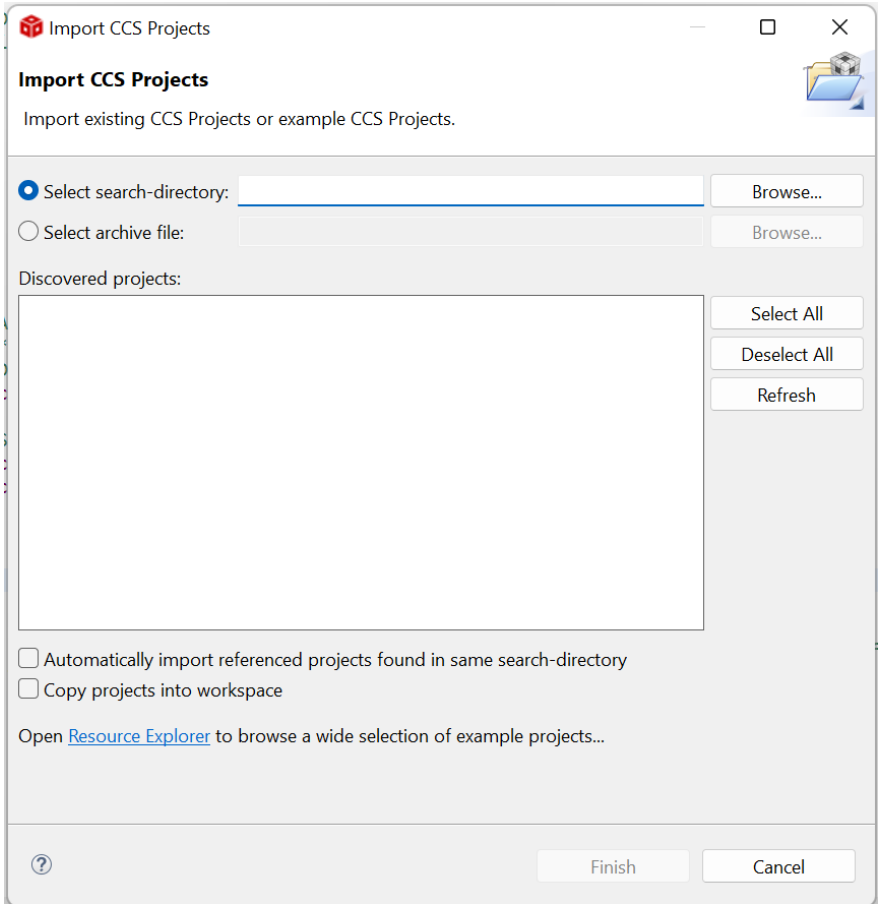

Code Composer Studio Launcher
Select a directory as workspace
Code Composer Studio uses the workspace directory to store its preferences and development artifacts.
Workspace:
C:\Users\ishan\workspace_v11
Browse...
Recent Workspaces
Copy Settings
Launch
Cancel

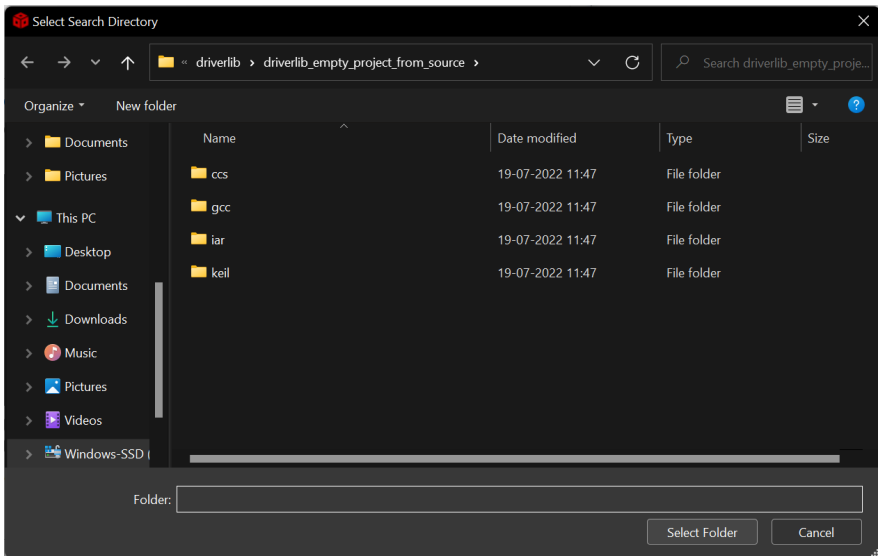
Practical 2: Hello world of Embedded Systems

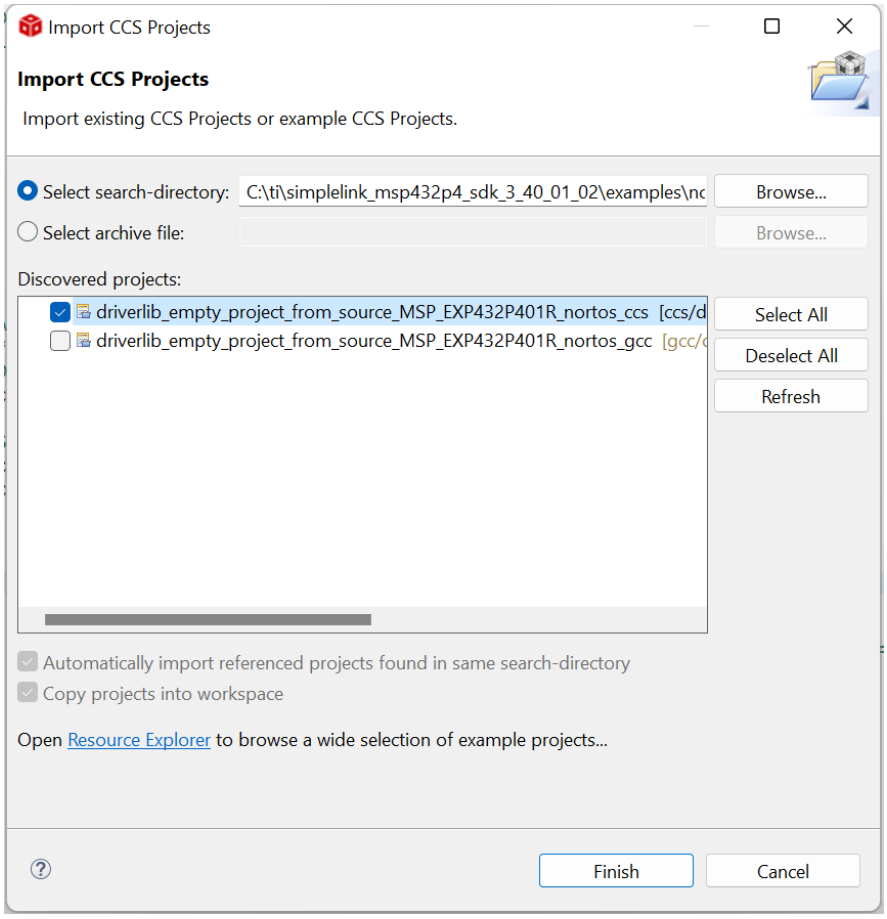
The blinking LED program is regarded as the "Hello World" program for embedded systems. Getting an LED to blink shows that the toolchain is properly configured. This practical walk through the whole project creation process. Also covered are the API functions used in the program, the Delay loop, and how the timing can be varied by tweaking the delay values.

Importing existing CCS project from Energia




Import CCS Projects
Import CCS Projects
Import existing CCS Projects or example CCS Projects.
Select search-directory:
Browse...
Select archive file:
Browse...
Discovered projects:
Select All
Deselect All
Refresh
Automatically import referenced projects found in same search-directory
Copy projects into workspace
Open Resource Explorer to browse a wide selection of example projects...
Finish
Cancel

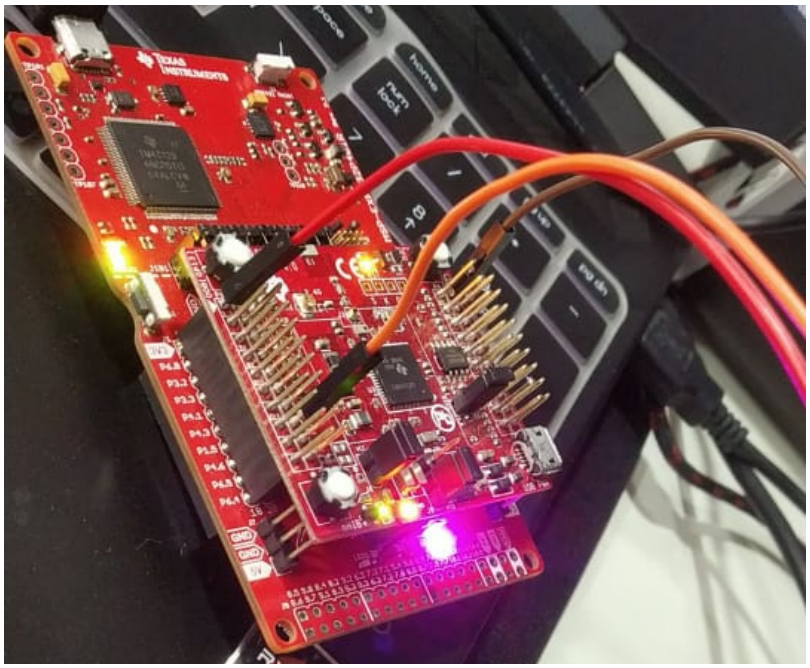

Select Search Directory
driverlib › driverlib_empty_project_from_source ›
Search driverlib_empty_proje...
Organize
New folder
Documents
Pictures
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows-SSD
Name
Date modified
Type
Size
ccs
19-07-2022 11:47
File folder
gcc
19-07-2022 11:47
File folder
iar
19-07-2022 11:47
File folder
keil
19-07-2022 11:47
File folder
Folder:
Select Folder
Cancel




```
#include <ti/devices/msp432p4xx/driverlib/driverlib.h>

/* Standard Includes */
#include <stdint.h>
#include <stdbool.h>

int main(void)
{
    /* Stop Watchdog */
    MAP_WDT_A_holdTimer(); //stops the timer the moment power is turned off
    int i;
    GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0); //setting output pin as pin 0 of port 2
    while(1)
    {
        GPIO_setOutputHighOnPin(GPIO_PORT_P2, GPIO_PIN0); // 1
        for(i=0;i<1000;i++); //random delay
        GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN0); // 0
        for(i=0;i<1000;i++); //random delay
```

```
  }
}
```

```
Console
CDT Build Console [Hello World]

**** Build of configuration Debug for project Hello World ****

"C:\\ti\\ccs1120\\ccs\\utils\\bin\\gmake" -k -j 16 all -O

gmake[1]: 'Hello World.out' is up to date.

**** Build Finished ****
```



Code Explanation

An LED blinking program has to accomplish the following:

- Set a GPIO (general purpose input/output) pin to be an output.
- Alternately drive this output pin (connected to an LED) high (1) and low (0).
- Delay each change in an output pin for a visible period of time.

MAP_WDT_A_holdTimer(); This function halts the timer as soon as the power is turned off.
GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0); This function configures the selected Pin as output pin. Setting Port 2's Pin 0 as the output pin in this case.
GPIO_setOutputHighOnPin(); This function sets output HIGH (1) on the selected pin.

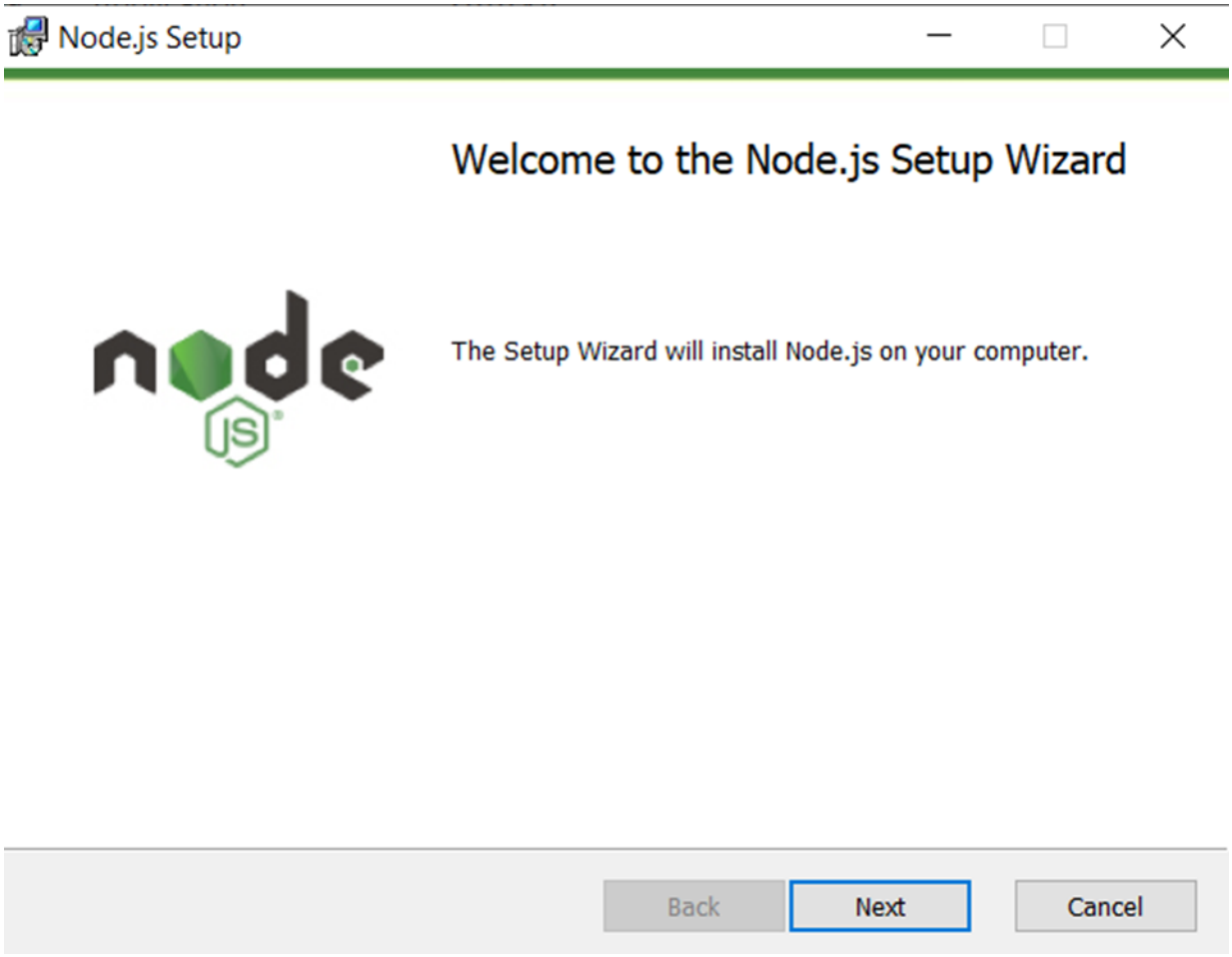
GPIO_setOutputLowOnPin(); This function sets output LOW (0) on the selected pin

The LED will get bright once the **GPIO_setOutputHighOnPin();** function has been called. Now, in the MSP sketch, we add a random delay to enjoy the bright glow for 1000 milliseconds (1 second). Because of this, the LED will glow for exactly 1 second after applying the high voltage. **GPIO_setOutputLowOnPin();** will provide a low voltage, causing no current to flow and, as a result, the LED to turn off. The subsequent for loop will once more provide a delay of one second to maintain the low volume for 1 second. In this manner, the LED shows a blinking effect until the power is delivered, i.e., while(1).

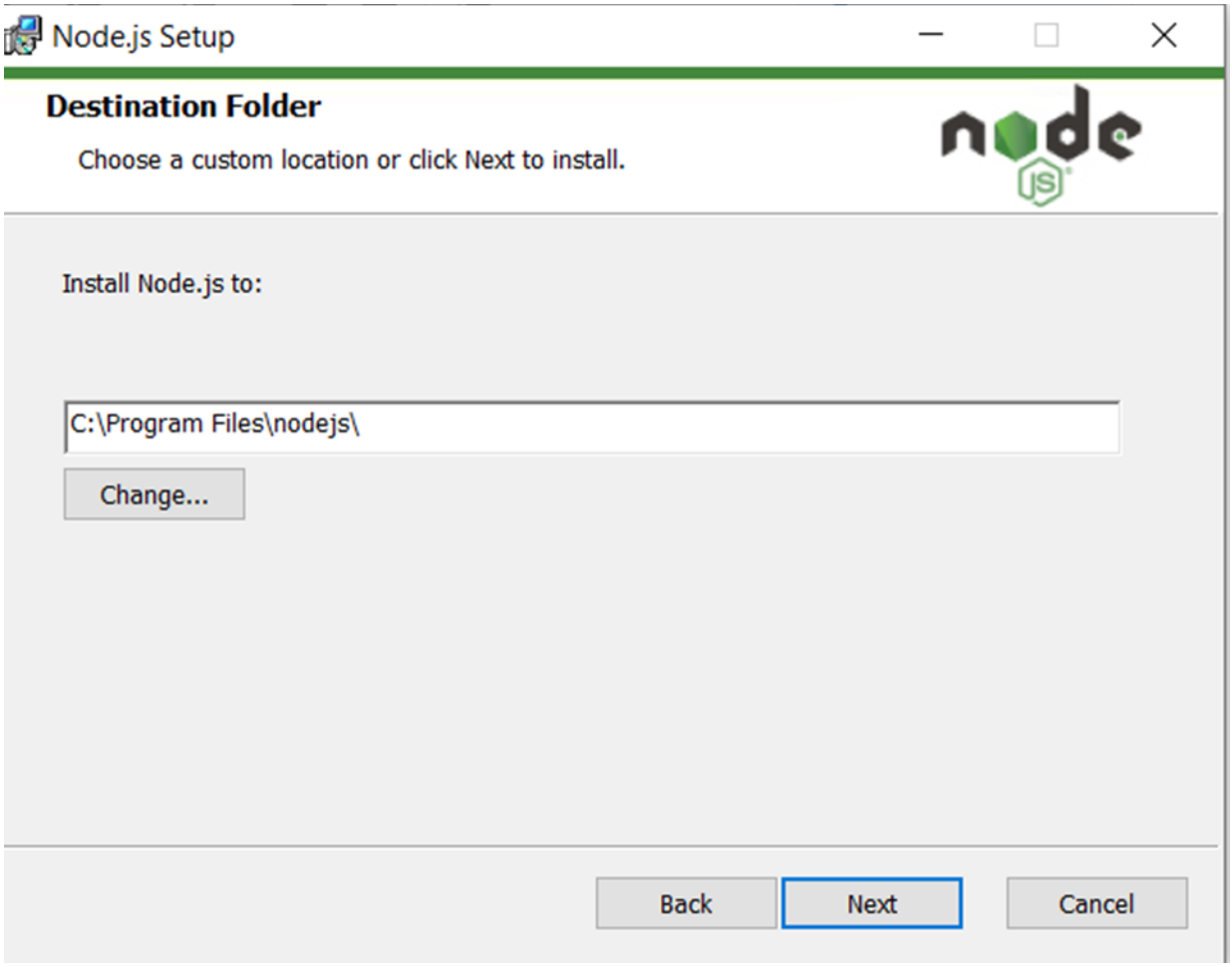
Practical 3: Installing Node.js ,Node-Red and Node Red interface

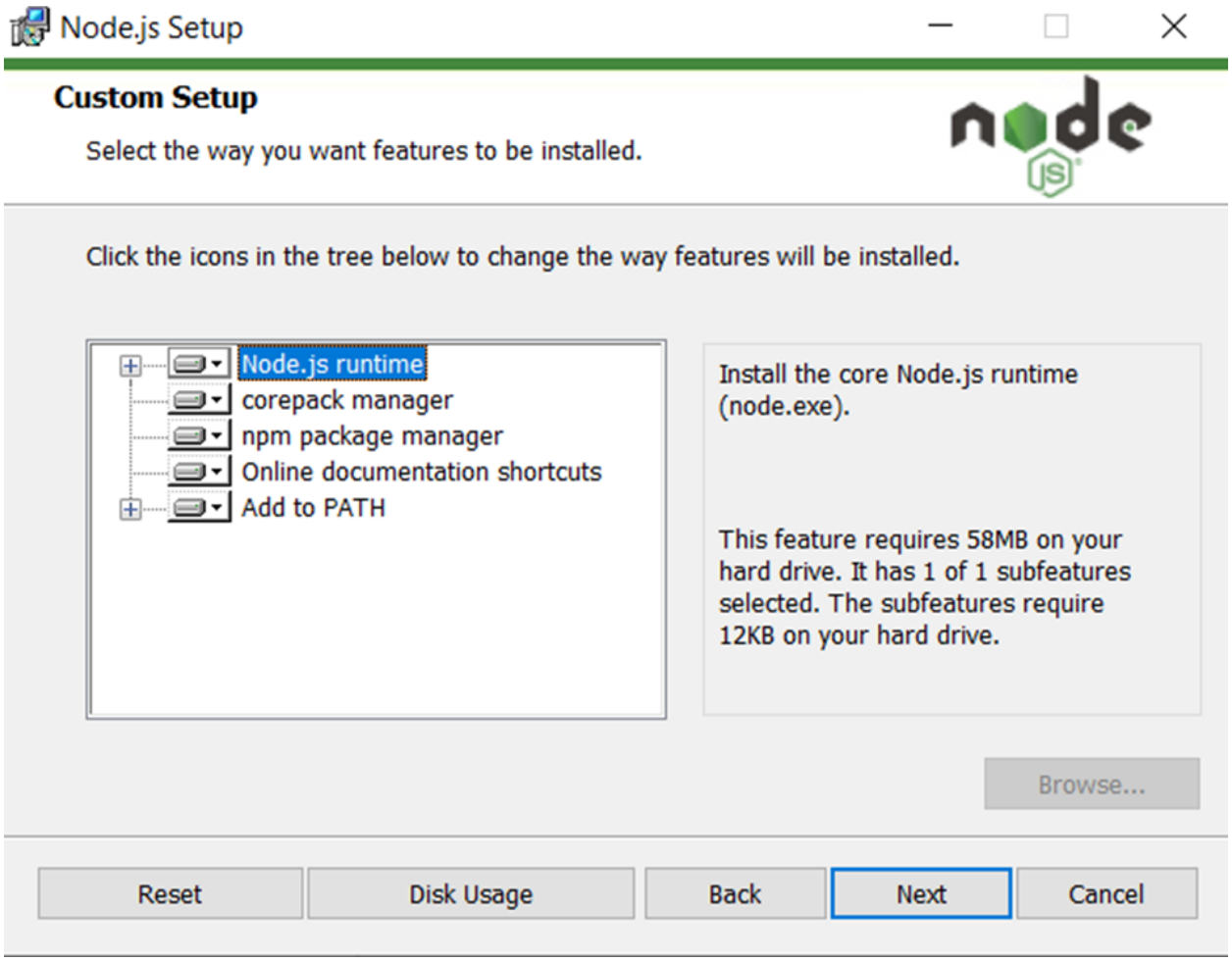
This practical mostly focuses on setting up and running Node-RED, a programming tool for integrating hardware components, APIs, and web services. It offers a browser-based editor that makes it simple to connect flows utilizing the many nodes in the palette that can be quickly deployed to its runtime.

Installing Node.js




Node.js Setup
End-User License Agreement
Please read the following license agreement carefully
Node.js is licensed for use as follows:
Copyright Node.js contributors. All rights reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so,
I accept the terms in the License Agreement
Print
Back
Next
Cancel


Node.js Setup
Destination Folder
Choose a custom location or click Next to install.
Install Node.js to:
C:\Program Files\nodejs\
Change...
Back
Next
Cancel



Installing Node-red

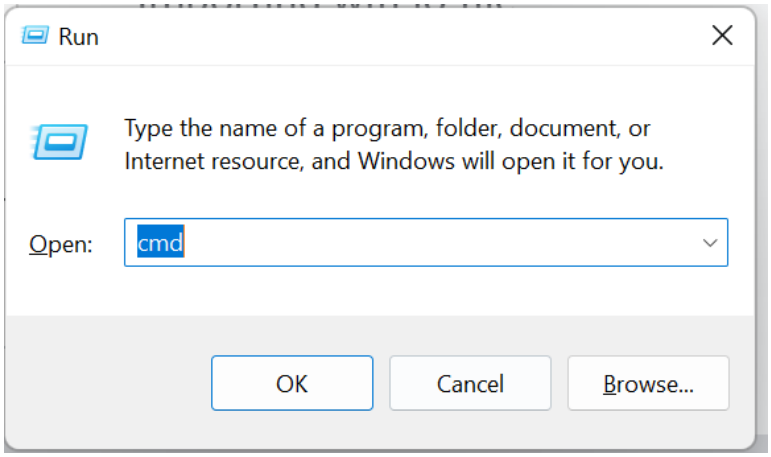
```
C:\Users\Owner>node --version && npm --version
v16.16.0
npm WARN config global `--global`, `--local` are deprecated. Use `--location=global` instead.
8.11.0
```

```
C:\Users\Owner>npm install -g --unsafe-perm node-red
npm WARN config global `--global`, `--local` are deprecated. Use `--location=global` instead.
npm WARN config global `--global`, `--local` are deprecated. Use `--location=global` instead.

added 294 packages, and audited 295 packages in 2m

38 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New minor version of npm available! 8.11.0 -> 8.15.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v8.15.0
npm notice Run npm install -g npm@8.15.0 to update!
npm notice

C:\Users\Owner>
```

Launching node red

Command Prompt → Type ‘node-red’ → Browser → http://localhost:1880/

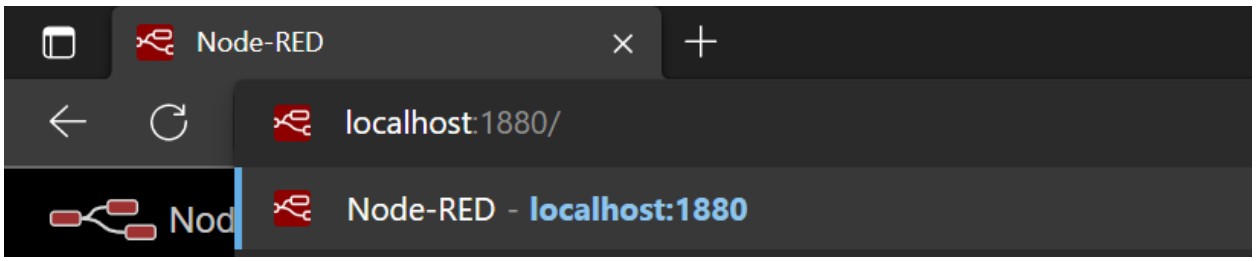
Running cmd



Typing ‘node-red’ to launch NodeRed

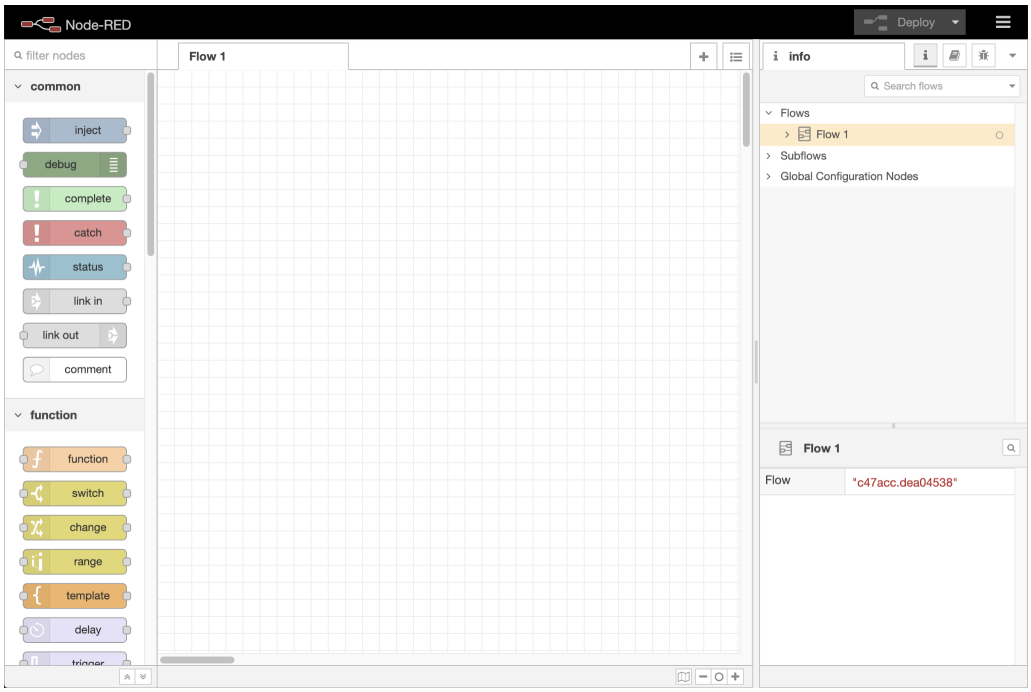
```
node-red
Microsoft Windows [Version 10.0.22000.675]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ishan>node-red
29 Jul 19:50:42 - [info]

Welcome to Node-RED
===================

29 Jul 19:50:42 - [info] Node-RED version: v3.0.1
29 Jul 19:50:42 - [info] Node.js  version: v16.16.0
29 Jul 19:50:42 - [info] Windows_NT 10.0.22000 x64 LE
29 Jul 19:50:43 - [info] Loading palette nodes
29 Jul 19:50:44 - [info] Dashboard version 3.1.7 started at /ui
29 Jul 19:50:44 - [info] Settings file  : C:\Users\ishan\.node-red\settings.js
29 Jul 19:50:44 - [info] Context store  : 'default' [module=memory]
29 Jul 19:50:44 - [info] User directory : \Users\ishan\.node-red
29 Jul 19:50:44 - [warn] Projects disabled : editorTheme.projects.enabled=false
29 Jul 19:50:44 - [info] Flows file     : \Users\ishan\.node-red\flows.json
29 Jul 19:50:44 - [info] Server now running at http://127.0.0.1:1880/
29 Jul 19:50:44 - [warn]

---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

29 Jul 19:50:44 - [info] Starting flows
29 Jul 19:50:44 - [info] Started flows
```

Running NodeRed on Port 1880 in the browser



NodeRed Components

NodeRed browser based IDE

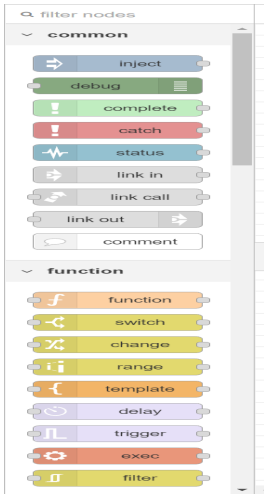


There are four sections in the editing window:

- **The Header on top.**

The header contains the deploy button, main menu, and, if user authentication is enabled, the user menu.

- **The Palette on the left.**

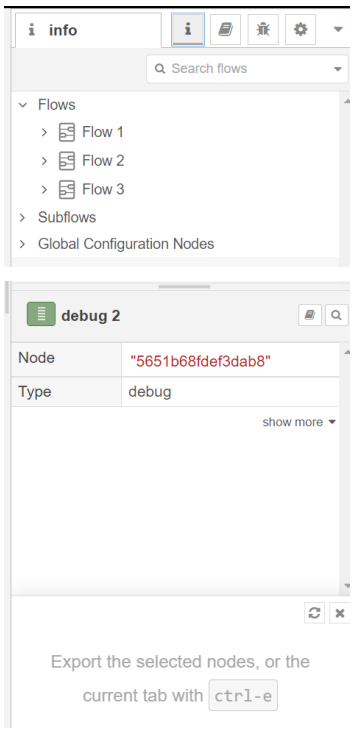


The palette contains all of the nodes that are installed and ready to utilize. They are arranged into a number of categories, with inputs, outputs, and functions at the top. Subflows appear in a category at the top of the palette.

- **The Main workspace in the middle.**

The main workspace is where flows are developed by dragging nodes from the palette and wiring them together. The workspace has a row of tabs along the top; one for each flow and any subflows that have been opened.

- **The Sidebar on the right.**



The sidebar contains panels that provide a number of useful tools within the editor.

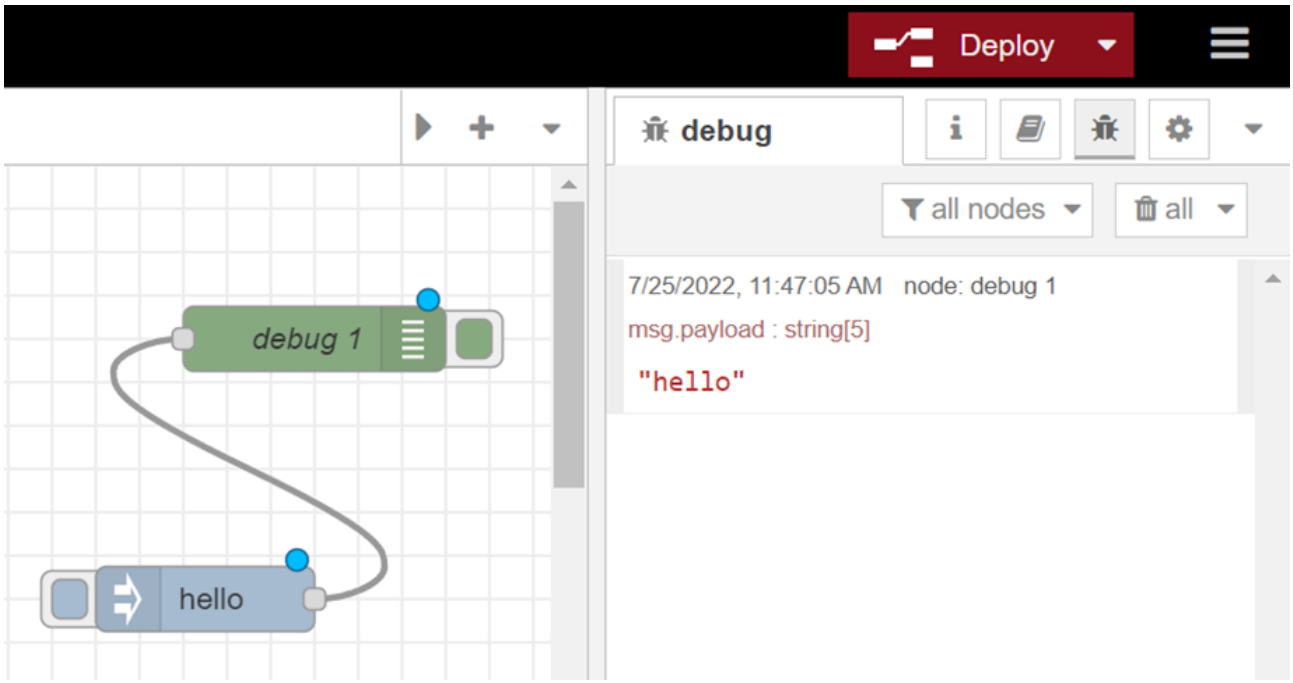
- **Information** - view information about nodes and their help
- **Debug** - view messages passed to Debug nodes
- **Configuration Nodes** - manage configuration nodes
- **Context data** - view the contents of context.

Practical 4: Introducing the inject, function, debug and switch nodes

This Practical demonstrates the fundamental NodeRed nodes and node-to-node flow.

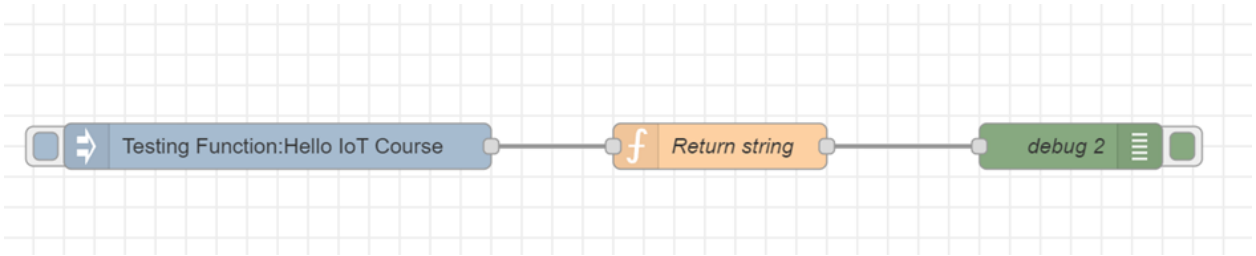
Inject Node

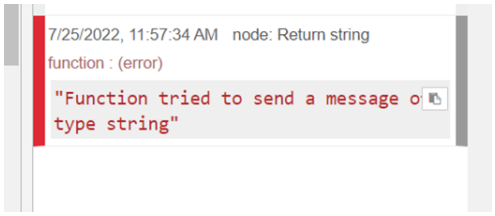
The Inject node can be used to manually trigger a flow by clicking the node's button within the editor. It can also be used to automatically trigger flows at regular intervals.



Function Node

The Function node allows JavaScript code to be run against the messages that are passed through it.

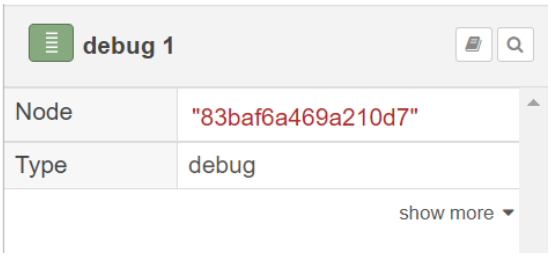




This function is returning a String object rather than a message object, as the error indicates.

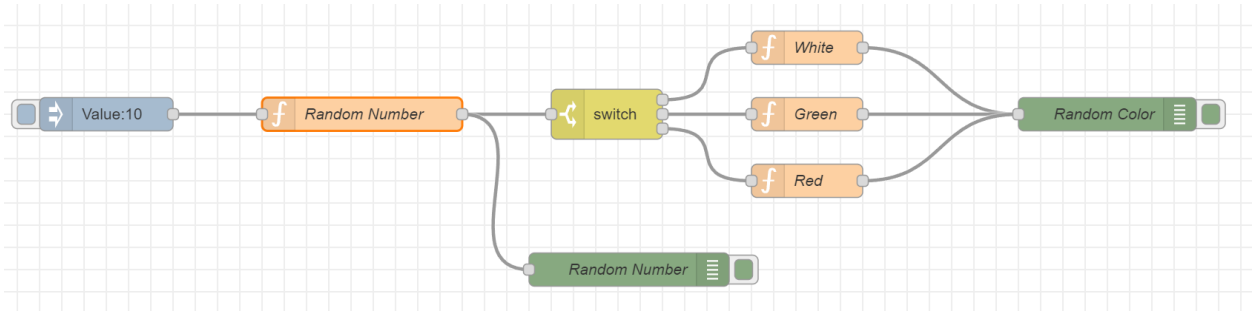
Debug Node

The Debug node can be used to display messages in the Debug sidebar within the editor.The sidebar provides a structured view of the messages it is sent, making it easier to explore the message. Alongside each message, the debug sidebar includes information about the time the message was received and which Debug node sent it. Clicking on the source node id will reveal that node within the workspace.

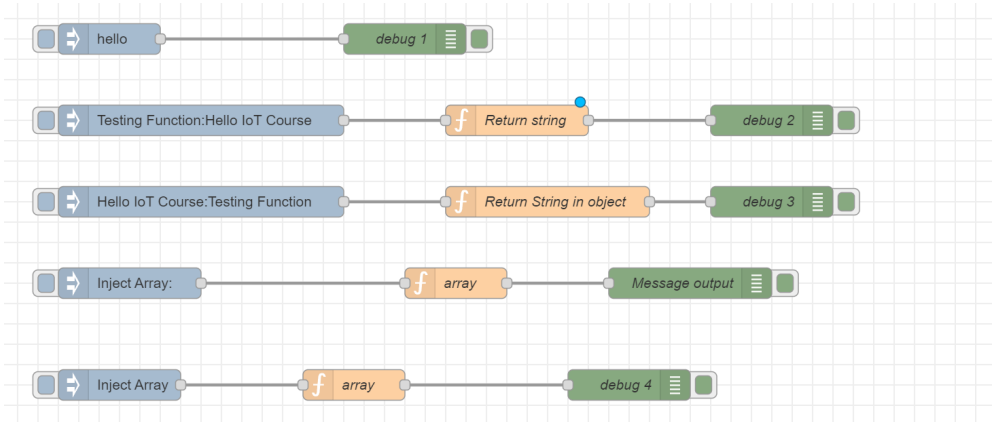


Switch Node

The Switch node allows messages to be routed to different branches of a flow by evaluating a set of rules against each message.

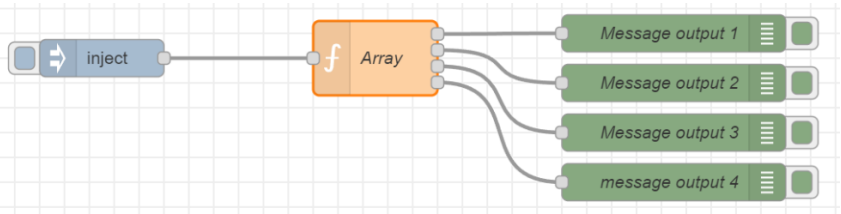


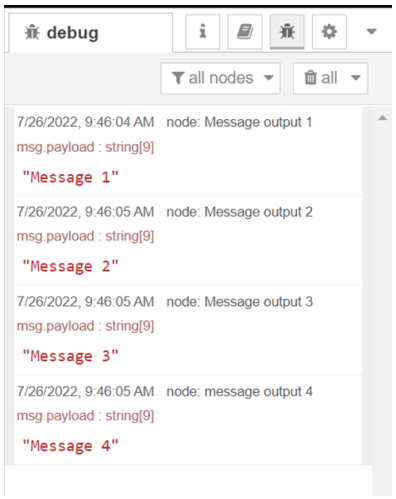
Hello IOT Course



Multi Payload Function

There might be more than one output for a function node. It comes in handy when we need to diverge or divide a single flow into several pathways.



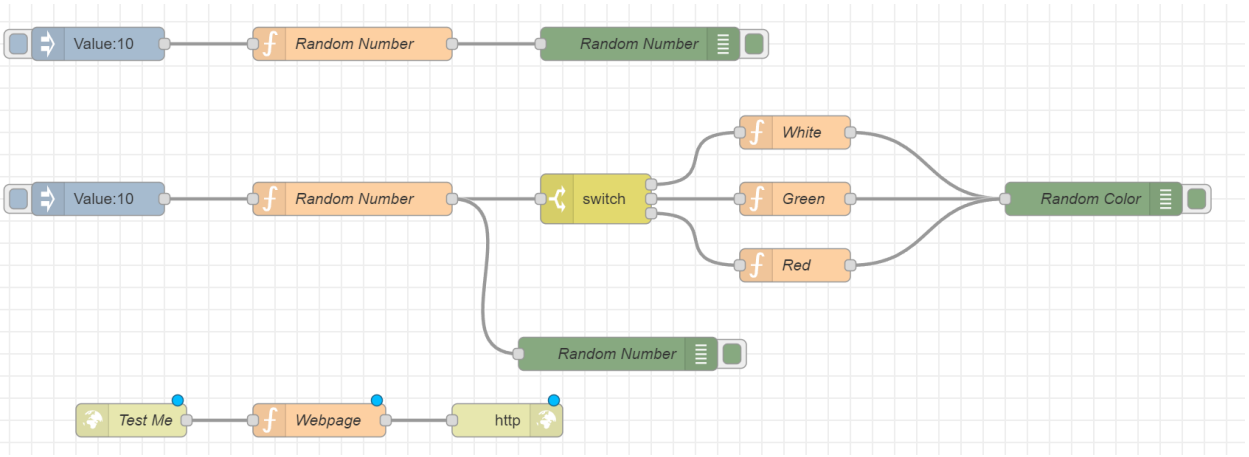



```
var msg1 = { "payload": "Message 1" }
var msg2 = { "payload": "Message 2" }
var msg3 = { "payload": "Message 3" }
var msg4 = { "payload": "Message 4" }
return [msg1, msg2, msg3, msg4];
```

Practical 5: Random number generator with selection of color. Introduce the HTTP node.

This Practical shows how to use nodes like inject, function, and switch nodes to create a random number generator with color customization. Additionally, the practical introduces the use of HTTP input and output nodes as well as a simple program for showing the flow and web pages using NodeRed.

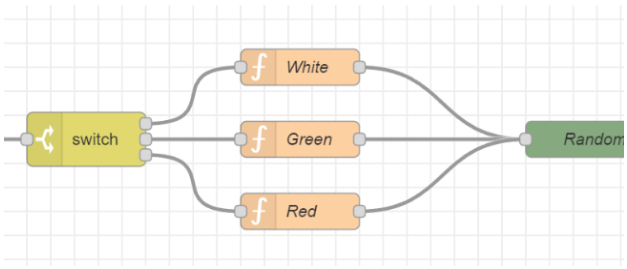
Random Number Function



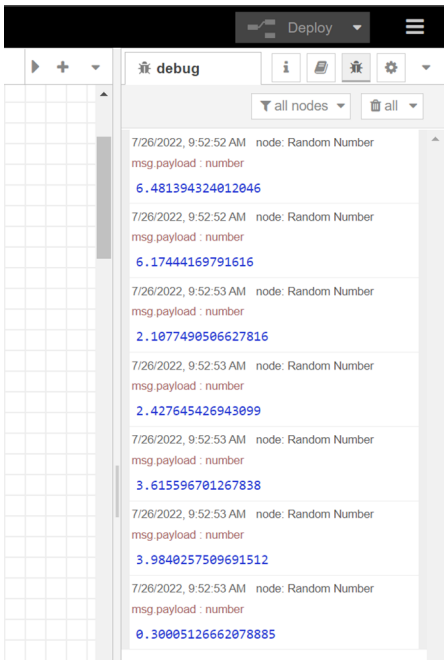

```
var x = msg.payload;
var rand = x*Math.random();
return = {"payload":rand};
```

The Math.random() function returns a floating-point, pseudo-random number in the range 0 to less than 1 (inclusive of 0, but not 1) with approximately uniform distribution over that range

Selection of Colors



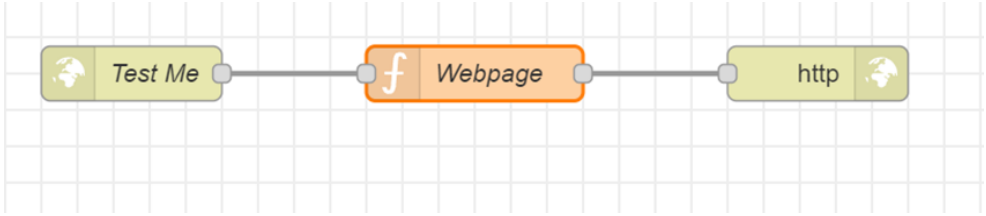

```
return {"payload":"I am White"};
return { "payload": "I am Green" };
return { "payload": "I am Red" };
```



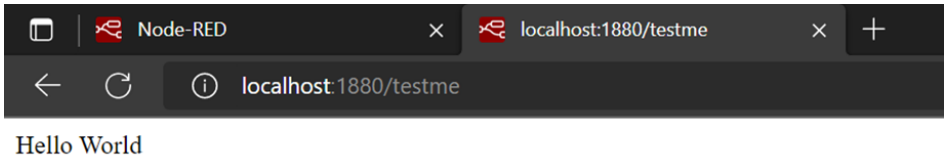
HTTP-In and HTTP-Response Node

To implement a web server with node-red you require two nodes. They are:

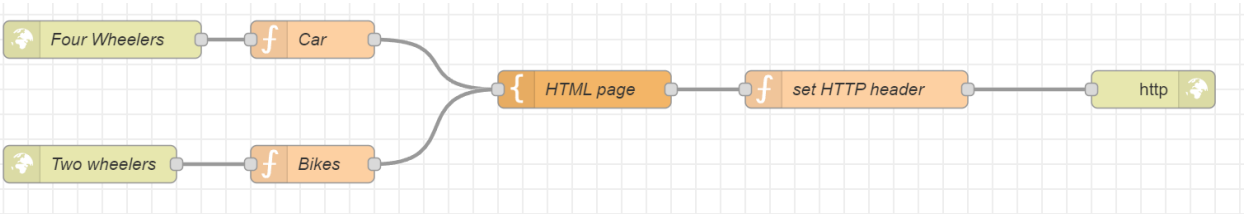
- **http-in** -accepts requests from a client.
- **http-response** -responds to requests from a client.



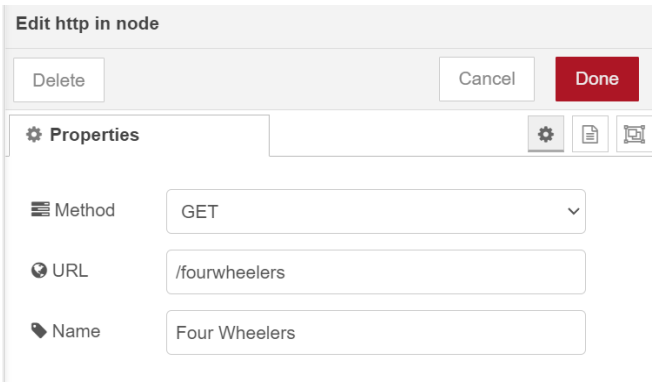

```
msg.payload="Hello World";
return msg;
```



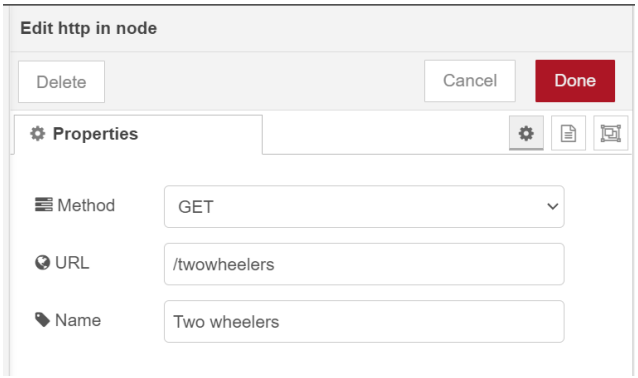
Two Wheelers and Four Wheelers WebPages



HTTP in node for Four Wheelers



HTTP in node for Two Wheelers



The objects in the Car and Bike functions are of the name-value pair type.

Car Function

```
msg.payload = {"Country":"India",
                "Make":"Maruti",
                "Cars":[{"car":"Baleno"},
                        {"car":"Ertiga"},
                        { "car": "Ignis" }]
            };
return msg;
```

Bikes Function

```
msg.payload = {
    "Country": "India",
    "Make": "Suzuki",
    "Bikes": [{ "bike": "Gixxer" },
    { "bike": "Intruder" }],
    "SuperBikes": [{ "bike": "Hayate" },
    { "bike": "Hayabusa" }]

};
return msg;
```

HTML Page Template

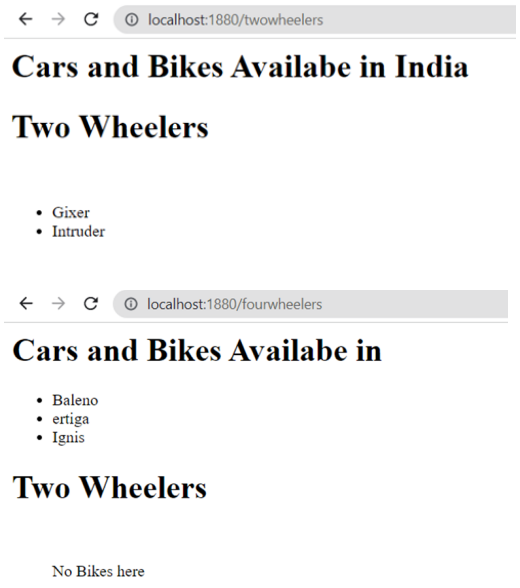
The template node, which contains HTML markup, is used to create web pages. Using Mustache templates, which is the usual setting, it may be used to dynamically insert data into a page.

```
<h1>Cars and bikes available in {{payload.Country}}</h1>

<ul>
    {{#payload.Cars}}
    <li>{{Car}}</li>
    {{/payload.Cars}}
</ul>
<h1>Two Wheelers!</h1>
```

```
<hr>

<ul>
    {{^payload.Bikes}}
    No Bikes here !!
    {{/payload.Bikes}}
    {{#payload.Bikes}}
    <li>{{bike}}</li>
    {{/payload.Bikes}}
</ul>
```

Set Http Header Function

```
msg.headers = {"Content-Type":"text/html"}
return msg;
```

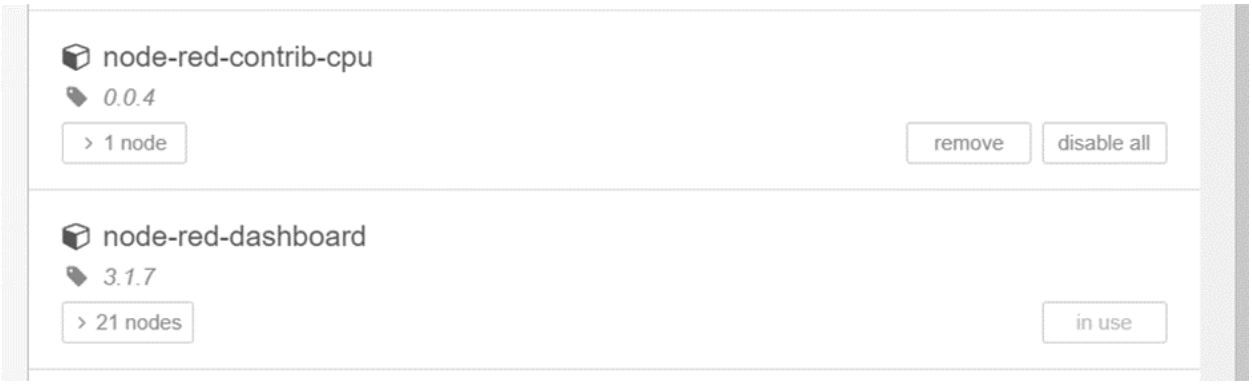


Practical 6: CPU utilization flow

This practical demonstration demonstrates the creation of a UI dashboard to monitor CPU usage. This program employs a number of nodes, including switch, function, inject, charts , and gauge nodes.

Installing node-red-contrib-cpu and node-red-dashboard

The **node-red-contrib-cpu node** is a form of convenience node that handles all calculations for real-time CPU usage tracking. The **node-red-dashboard** is a plugin that allows you to create live dashboards.



Flow of CPU usage

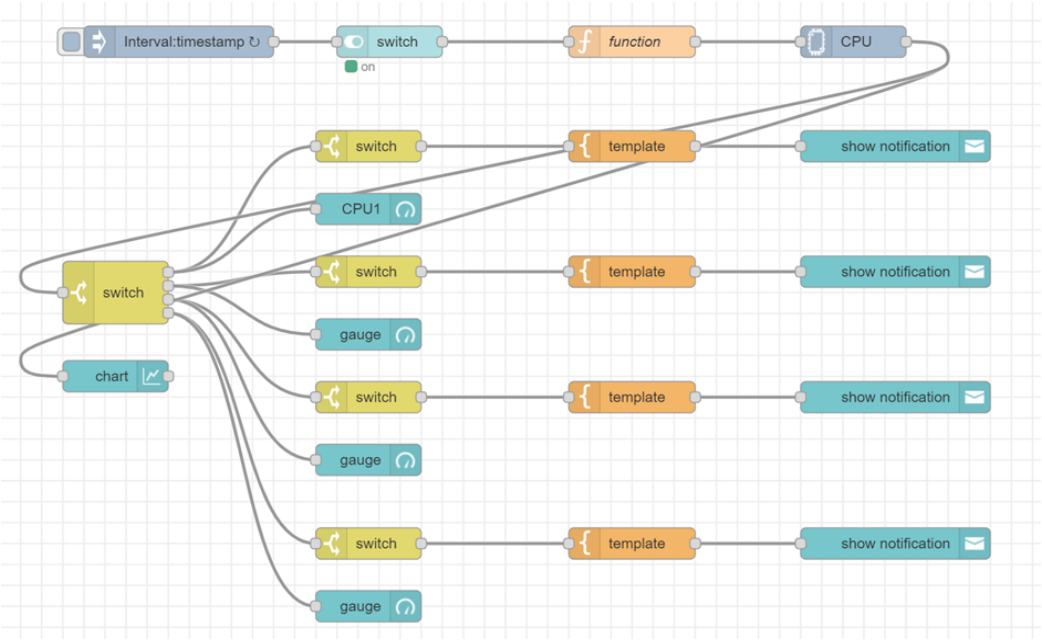
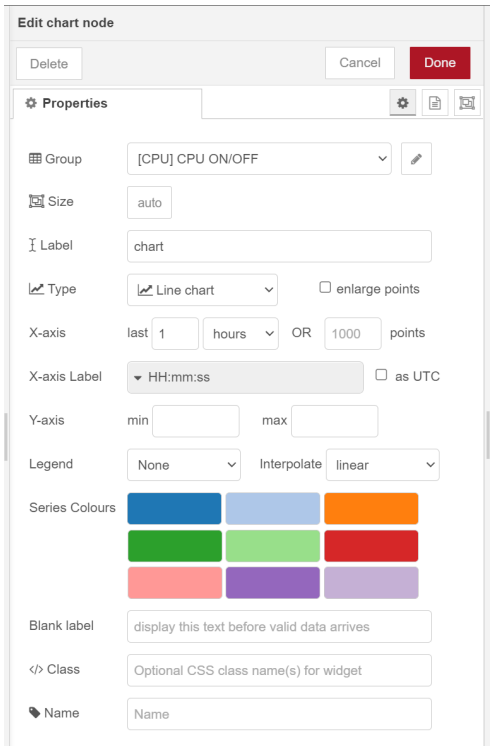
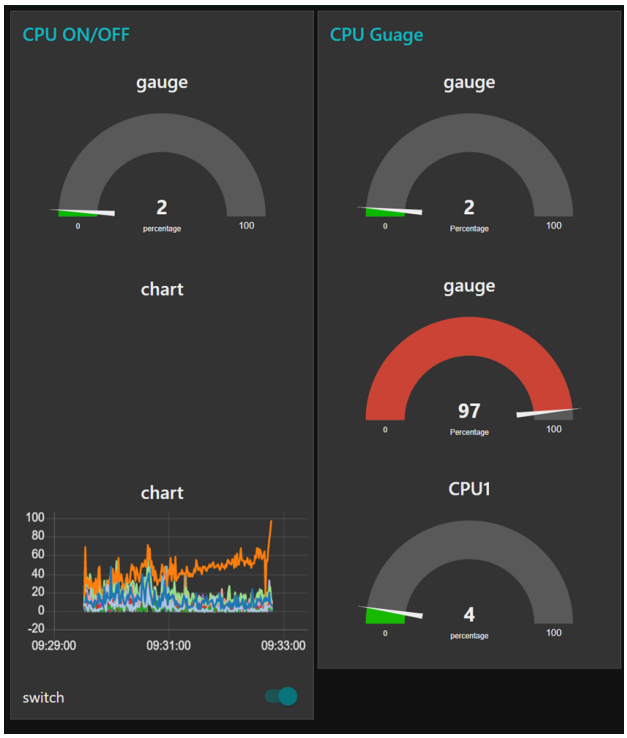


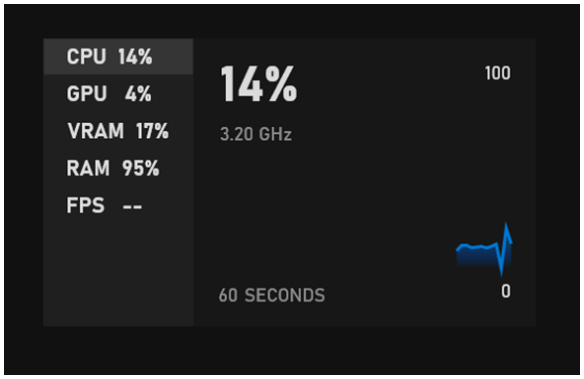
Chart Node



Connecting **CPU node** to **debug node** to see message for each core usage on **debug panel.** Connecting CPU node output to **chart** dashboard to chart a line graph of messages from CPU. Using **switch nodes** to separate data coming from all 4 cores. Creating separate **CPU Gauges** to map for each core's data.



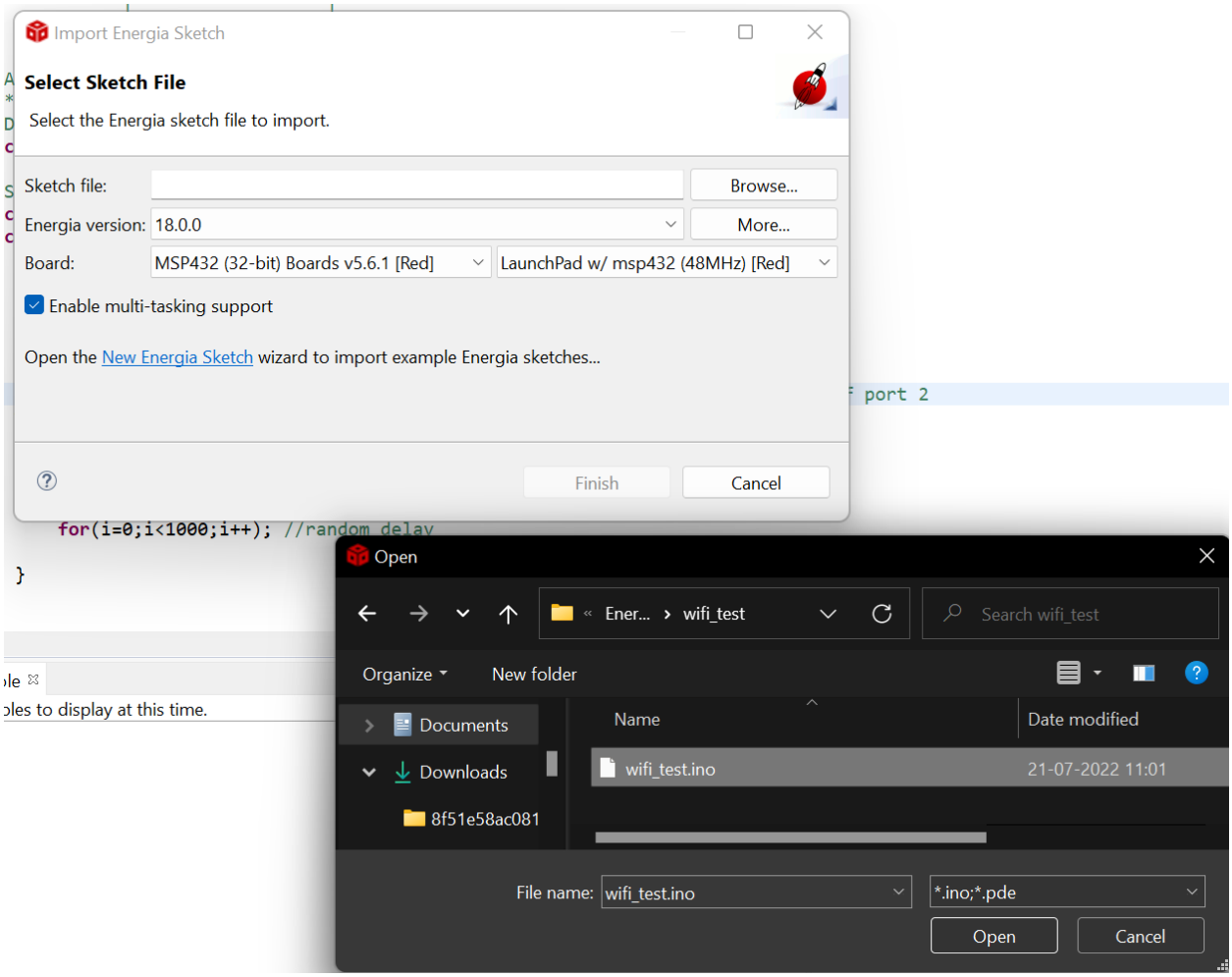
System's default CPU utilization



Practical 7: WiFi Setup

The practical demonstrates WiFi setup and configuration, after which all nearby devices will be identified by the board and presented on the emulator, i.e. PuTTY.

Importing wifi.io file



Wifi_test.ino

```
#include <WiFi.h>
#include <SPI.h>



IPAddress shieldIP, subnetMask, gatewayIP;
uint8_t rssi;
uint8_t networkID;
```

```
byte macAddr[6]; //mac address is 6 byte
byte encryptionType;

char ssid[]="Redmi";//my cell phone
//char password[]="12345678";

void setup() {
  // put your setup code here, to run once:
    Serial.begin(115200);
    Serial.print("Connecting to Wifi..");
    while(WiFi.begin(ssid)!=WL_CONNECTED)// for cell phone
      //  while(WiFi.begin(ssid, password)!=WL_CONNECTED)
        {
            Serial.println("");
            delay(1);  //delay in ms

        }
    Serial.println("");
    Serial.print("WiFi connected, Fetching WiFi Sheild's IP Address:");
    while(WiFi.localIP()==INADDR_NONE){
        Serial.print(".");// while the connection isnt established, the dot animation should continue
        delay(1);
 }
    shieldIP = WiFi.localIP();
    Serial.println(shieldIP);

    Serial.print("Access point name: ");
    Serial.println(ssid);

    Serial.print("Signal Strength: ");
    rssi = WiFi.RSSI();
    Serial.println(rssi); // RSSI - Recieved Signal Strength Indication

    uint8_t networkID = WiFi.scanNetworks();
    Serial.print("Number of access points in range:");
    Serial.println(networkID);
    for(int i=1;i<=networkID;i++){
        Serial.print("Name of the Access Points and encryption type:");
        Serial.print(WiFi.SSID(i));
        Serial.print(",");
        encryptionType = WiFi.encryptionType(i);
        // Serial.println(encryptiontype,HEX);
        Serial.println(encryptionType,DEC);
    }
```

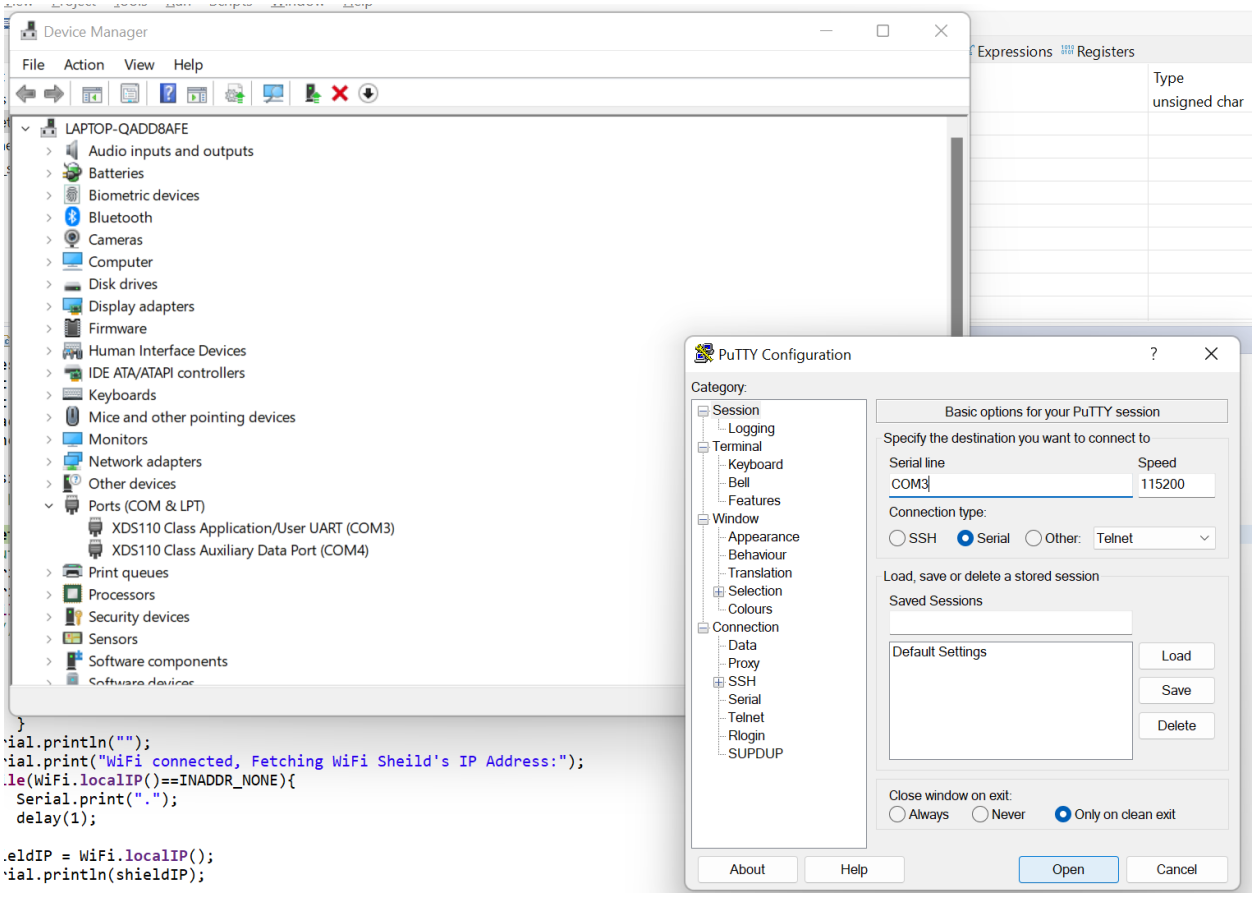
```
   subnetMask = WiFi.subnetMask();
   Serial.print("Subnet Mask: ");
   Serial.println(subnetMask);

    gatewayIP = WiFi.gatewayIP();
    Serial.print("gateway IP Address: ");
    Serial.println(gatewayIP);

    WiFi.macAddress(macAddr);
    Serial.print("Mac Address of shield:");
    for(int i =0; i< 6; i++)
    {
       Serial.print(macAddr[i],HEX);
       if(i<=4)
          Serial.print(':');
       else
          Serial.println();
}

   }

void loop() {
  // put your main code here, to run repeatedly:

}
```

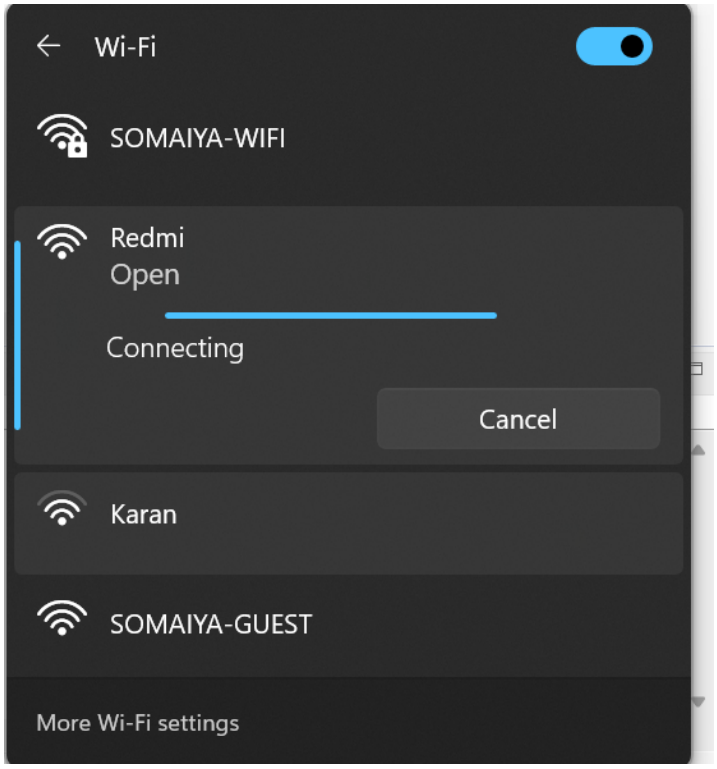
Code Explanation

- The code starts with declaration of variables where **shieldIP, subnetMask, gatewayIP** are of type **IPAddress**. A variable of this type can also be stored in an associative array either as a key or as a value.
- **Rssi** - Received Signal Strength Indication determines the signal strength.
- **networkID** shows the number of networks in the vicinity.
- **Wclient** of type **WiFiClient** Creates a client that can connect to a specified internet IP address and port as defined in client.connect().

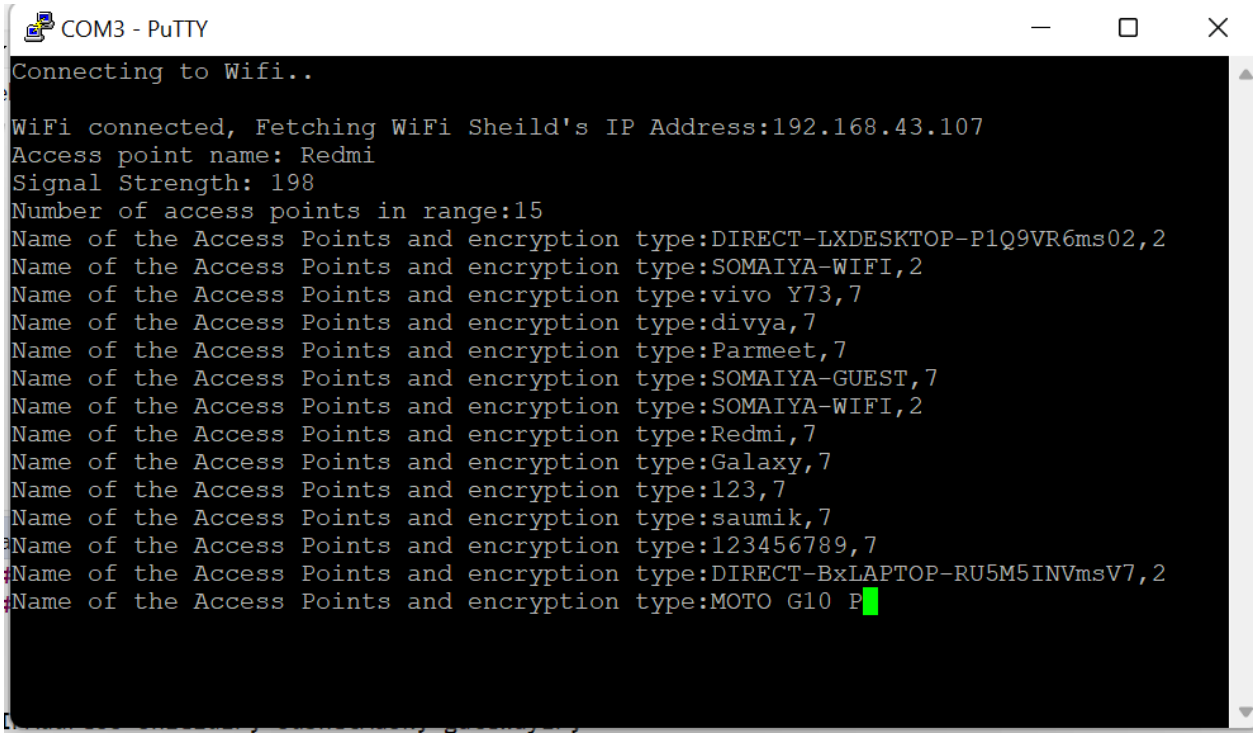
Checking the port on which the microcontroller MSP432 is connected and Configuring PuTTY

PuTTY is an emulator being used here to display the available devices in the vicinity of the MSP432 board.





Detecting available devices



Practical 8: Analog To Digital Converter

This practical demonstrates the use of ADC which is a converter which converts continuous time-varying analog signals into discrete-time digital signals so that they can easily be read by the digital devices.

Analog to Digital Converter

The Analog-to-Digital Converter allows the MSP432 Microcontroller to communicate with the real world. Basically an analogue to digital converter takes a snapshot of an analogue voltage at one instant in time and produces a digital output code which represents this analogue voltage.

Code

```
#include <stdint.h>
#include <stdbool.h>

int main(void)
{
    //Analog to digital conversion
    uint16_t cResult;  //to holde 12 bits ADC
    /* Stop Watchdog */



    MAP_WDT_A_holdTimer();
    GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0|GPIO_PIN1|GPIO_PIN2); //Pipe symbols are used for redirecting to the next port here | set output because LED shows output

GPIO_setAsPeripheralModuleFunctionInputPin(GPIO_PORT_P4,GPIO_PIN7,GPIO_TERTIARY_MODULE_FUNCTION); //potentiometer is connected to port 4 pin 7
    ADC14_initModule(ADC_CLOCKSOURCE_MCLK, ADC_PREDIVIDER_1, ADC_PREDIVIDER_1, ADC_NOROUTE); //Multiple predividers are required to get a lower number | predivider_1 is dividing the clock by 1| Nouroute specifies the external connection - using ADC to connect to the pin
    ADC14_configureSingleSampleMode(ADC_MEM6,true); // ADC_MEM6 is the memory location associated with the particular port
    ADC14_configureConversionMemory(ADC_MEM6, ADC_VREFPOS_AVCC_VREFNEG_VSS, ADC_INPUT_A6, false); //A6 is the channel | using port 3.3v for the reference voltage  with ADC_VREFPOS
    ADC14_setSampleHoldTime(ADC_PULSE_WIDTH_64, ADC_PULSE_WIDTH_64); //Sampling and Holding signal | there are two ADC_PULSE_WIDTH_64 used becuase the API requires two parameters
```

```
   ADC14_enableSampleTimer(ADC_AUTOMATIC_ITERATION); //specifies that the timer is
running automatically
   ADC14_enableModule(); //activating the module
   ADC14_enableConversion(); //final conversion
   ADC14_toggleConversionTrigger(); //Triggering the conversion continously



   while(1)
   {
     while(!ADC14_isBusy()); //while the ADC is busy the other inputs should wait
        cResult=ADC14_getResult(ADC_MEM6); //12 bits results for 12 bit ADC | values should
be continuously stored in MEM6
        P2OUT=cResult>>8;
        ADC14_toggleConversionTrigger();

   }
}
```

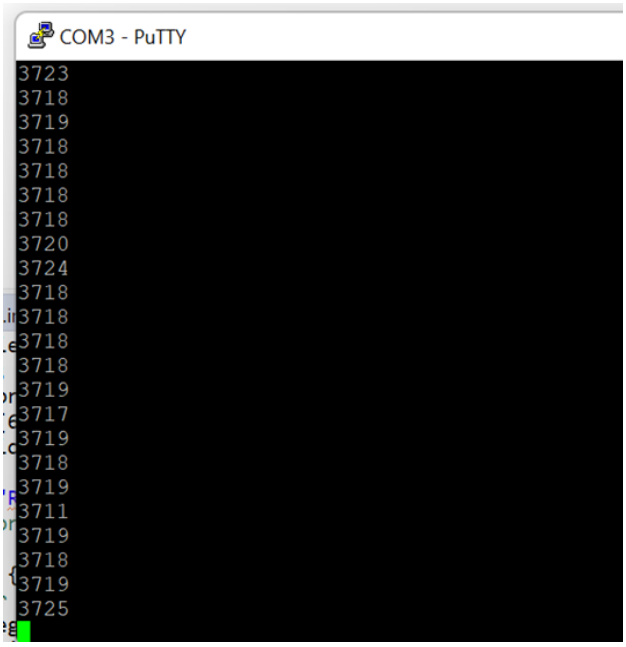
Code Explanation

- ADC is initialized and configured and is enabled for operation
- ADC clock, pre divider and clock divider are initialized through ADC14_initModule();. The divider and the predivider are set as 1 so the clock effectively is at 3 Mhz for the ADC.
- The sample and hold source and time of the ADC are configured to choose the ADC memory where the converted data will be kept.
- ADC reference voltage and ADC iterate are configured

API functions used configuring ADC

ADC14_initModule(); // to initialize ADC module
ADC14_configureSingleSampleMode();// setting up the memory location used for storing the conversion result
ADC14_configureConversionMemory(); //Configuring some more parameters of ADC
ADC14_setSampleHoldTime(); //To sample the analog pin for getting a good digital replica of analog signals
ADC_14_setResolution();//setting ADC resolution
ADC14_enableSampleTimer();// Automatically moving to the next sample after the previous sample is sampled
ADC14_enableModule(); //The ADC module is enabled
ADC14_enableConversion(); //Conversion is enabled
ADC14_toggleConversionTrigger(); //Trigger is enabled

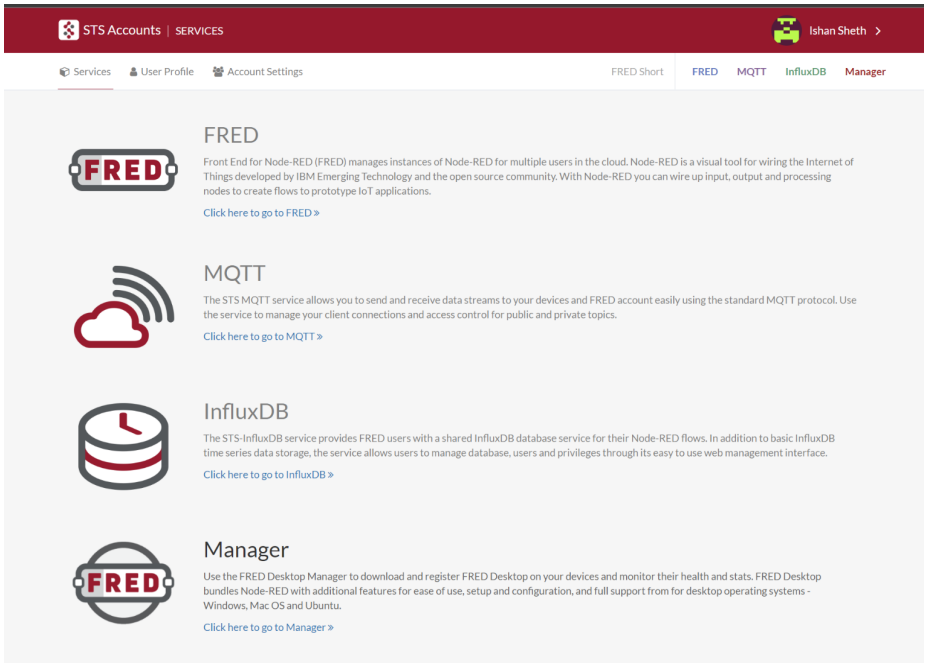
Readings from the Potentiometers



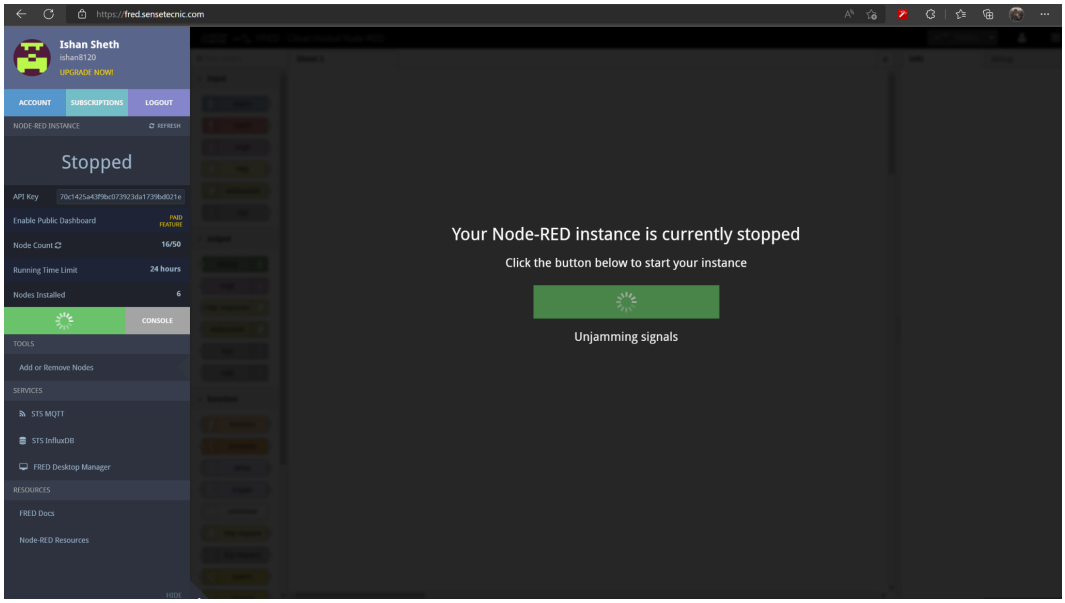
ADC	TRUTH	ADC * 1000
0.7000	885	700
0.6900	769	690
0.8300	1052	830
1.1500	1464	1150
1.3700	1735	1370
1.5000	1918	1500
1.5800	2014	1580
1.6500	2085	1650
1.6800	2140	1680
1.7100	2174	1710
1.7600	2246	1760
1.7800	2249	1780
1.8200	2308	1820
1.8400	2326	1840
1.9100	2428	1910
1.9900	2530	1990
2.0300	2566	2030
2.0600	2623	2060
2.0800	2632	2080
2.1400	2712	2140
2.1600	2751	2160
2.2000	2783	2200
2.2500	2859	2250
2.3400	2975	2340
2.4000	3124	2400
2.5400	3224	2540
2.8300	3593	2830
2.9600	3707	2960
3.2500	4089	3250
3.2600	4094	3260
lope	0.786296	
ntecept	7.695073	
orelation	0.999397	

Practical 9: Sending Raw data to the cloud

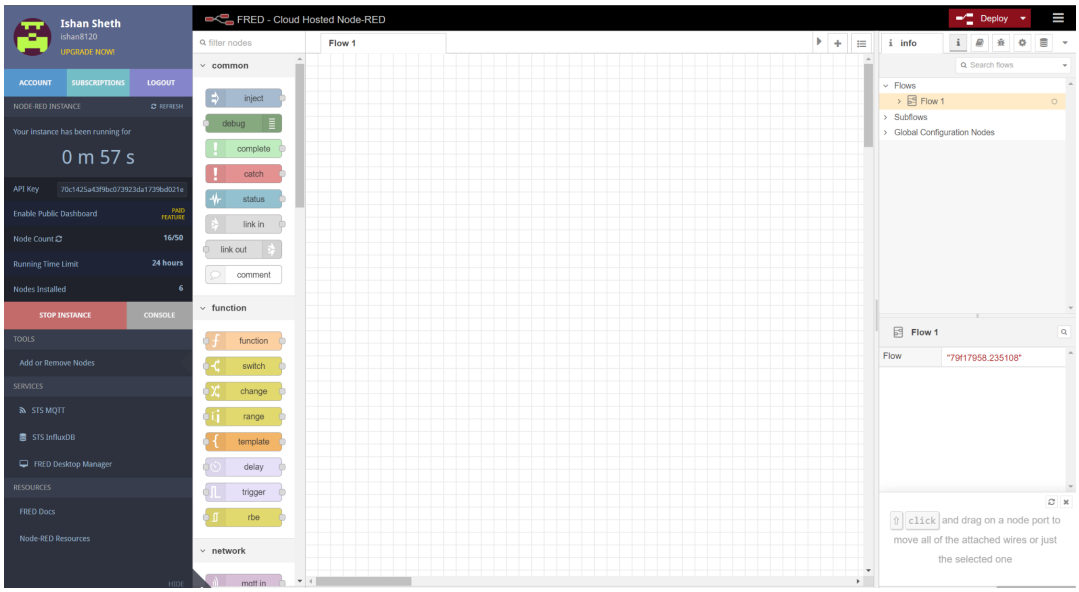
This practical deals with sending raw data to the cloud. Here, FRED is used to manage instances of Node-Red for multiple users in the cloud. MQTT is the most commonly used messaging protocol for the Internet of Things (IoT). MQTT stands for MQ Telemetry Transport. The protocol is a set of rules that defines how IoT devices can publish and subscribe to data over the Internet.



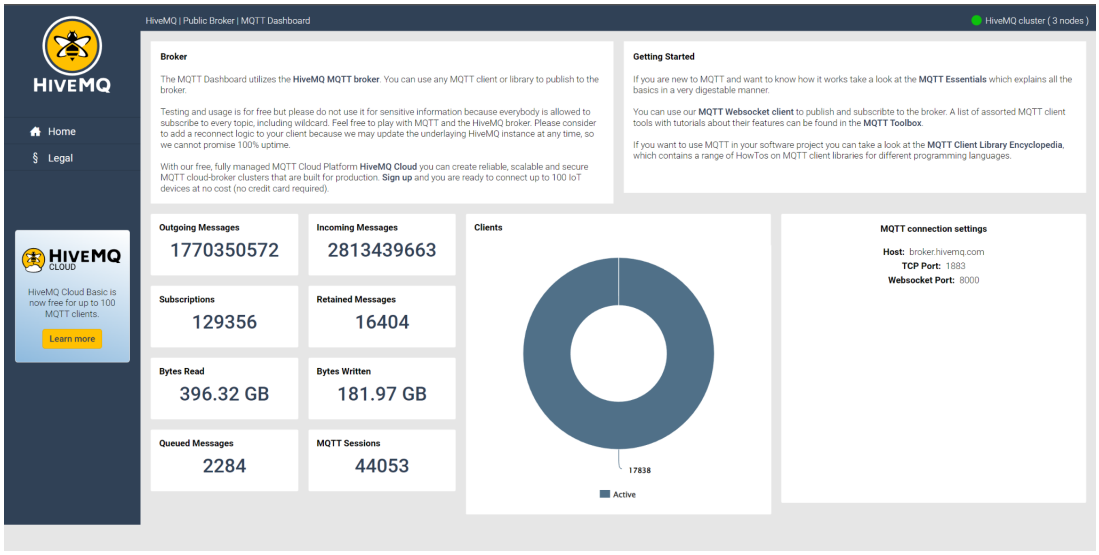
Creating Instance



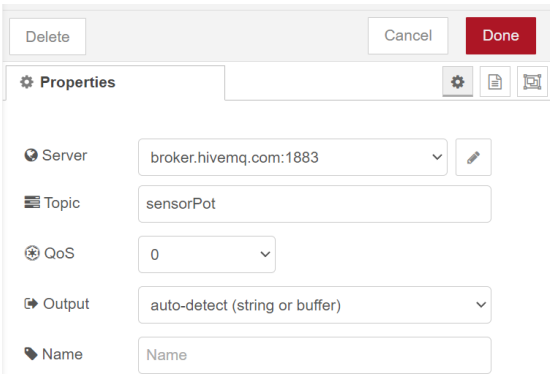
FRED IDE



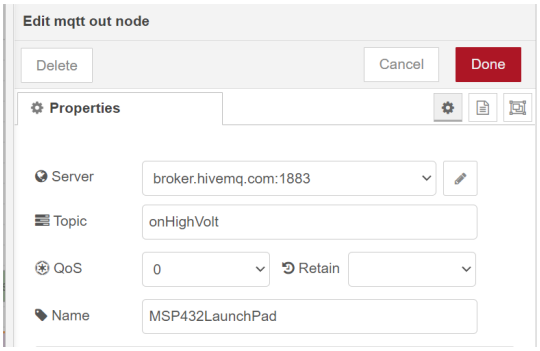
HIVEMQ Dashboard



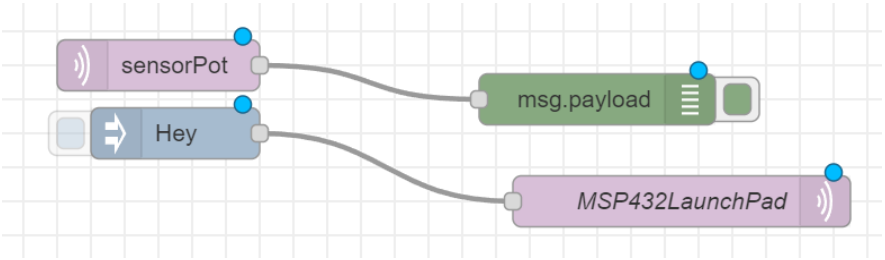
MQTT input node : SensorPOT



MQTT out node : MSP432LaunchPad



Flow between subscriber and publisher



Practical 10: Connecting MSP432 to the cloud

This practical demonstrates the connection between the MSP432 launchpad and the cloud. In this application, cloud computing is used to provide remote control of the LED on the board. The LED may be remotely controlled via a UI dashboard.

Code

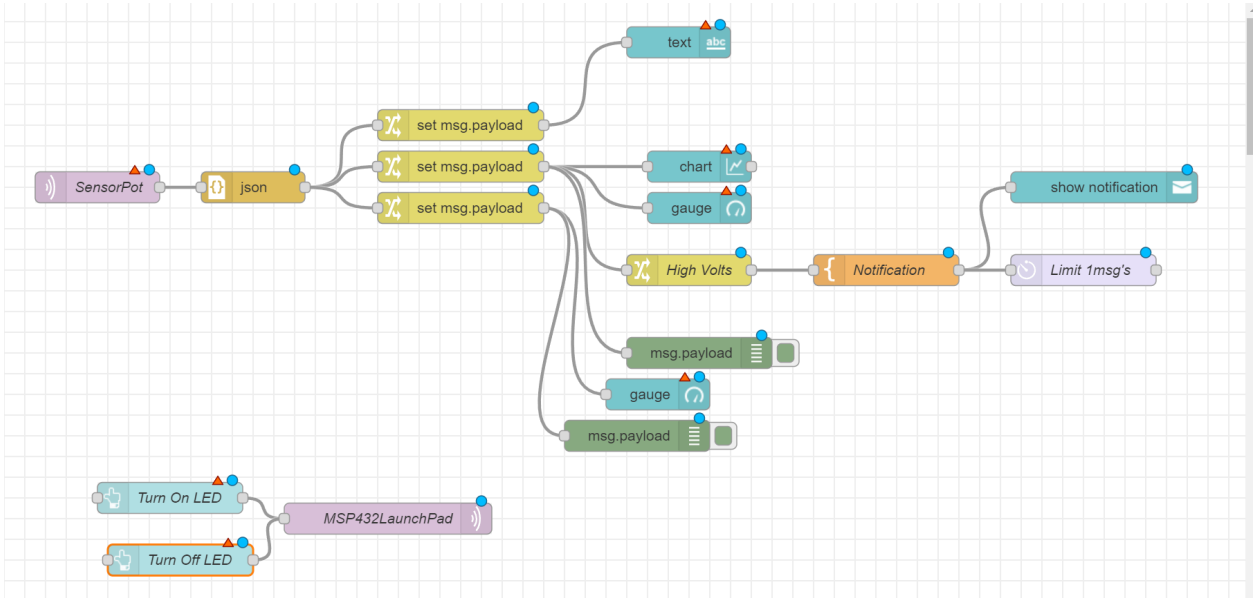
```
#include <ti/devices/msp432p4xx/driverlib/driverlib.h>
#include<WiFi.h>
#include<SPI.h>
#include<PubSubClient.h>
#include<aJSON.h>



IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6];
byte encryptionType;

char ssid[]="vivoY73";

WiFiClient wclient;

char server[]="broker.hivemq.com";
PubSubClient client(server,1883,callback,wclient);

char* jsonPayload;
aJsonStream serial_stream(&Serial);



void setup() {
  // put your setup code here, to run once:
    Serial.begin(115200);
       Serial.print("connecting to WIFI..");
       while(WiFi.begin(ssid)!=WL_CONNECTED)//for cell phone //WL_CONNECTED while it is
not connected
           //while(WiFi.begin(ssid.password)!=WL_CONNECTED)
       {
           Serial.print(".");
           delay(1);//delay in ms
       }
```

```
Serial.println("");
Serial.print("Wifi Connected , Fetching Wifi Sheild's IP address :");
while(WiFi.localIP()==INADDR_NONE){
   Serial.print(".");
   delay(1);
}
shieldIP = WiFi.localIP();
Serial.println(shieldIP);

Serial.print("Access Point name:");
Serial.println(ssid);

Serial.print("Signal Strength");//
rssi=WiFi.RSSI();//it fecthes the signal strength
Serial.println(rssi); //RSSI-Recived signal strength indication

uint8_t networkId= WiFi.scanNetworks();
Serial.print("number of access points in range:");
Serial.println(networkId);

for(int i=1;i<=networkId;i++){
      Serial.print("Name of Access points and encryption type:");
      Serial.print(WiFi.SSID(i));
      Serial.print(",");
      encryptionType=WiFi.encryptionType(i);
      Serial.println(encryptionType,HEX);
      //Serial.println(encryptionType,DEC);
   }
   subnetMask = WiFi.subnetMask();
   Serial.print("Subnet Mask:");
   Serial.println(subnetMask);

   gatewayIP=WiFi.gatewayIP();
   Serial.print("Gateway IP Address:");
   Serial.println(gatewayIP);

   WiFi.macAddress(macAddr);
   Serial.print("Mac Address of Sheild:");
   for(int i=0;i<6;i++){
      Serial.print(macAddr[i],HEX);
      if(i<=4)
         Serial.print(':');
      else
         Serial.println();
```

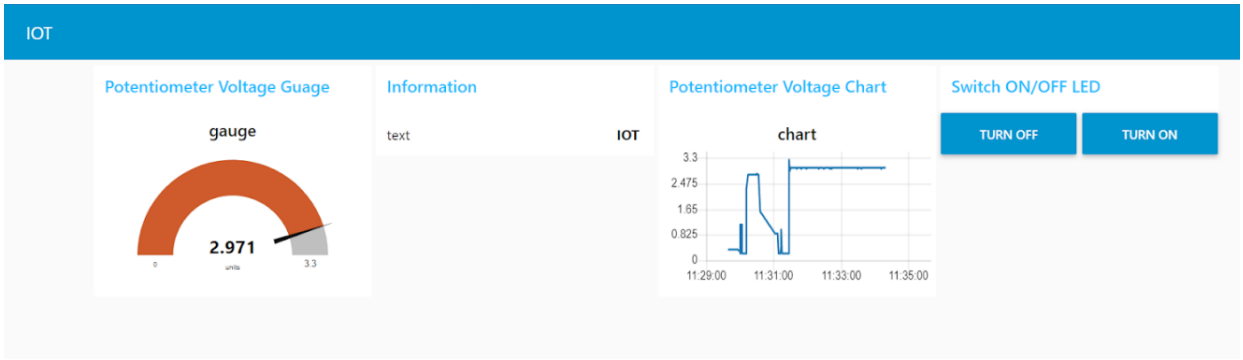
```
        }
           GPIO_setAsOutputPin(GPIO_PORT_P1,GPIO_PIN0);
           GPIO_setAsOutputPin(GPIO_PORT_P2,GPIO_PIN0|GPIO_PIN1|GPIO_PIN2);

GPIO_setAsPeripheralModuleFunctionInputPin(GPIO_PORT_P4,GPIO_PIN7,GPIO_TERTIAR
Y_MODULE_FUNCTION);//potentiometer is connect in port 4 ,
                                                                    //pin 7 becoz ADC
exists in PIN 7 of port 4 and then digital signal goes to LED

ADC14_initModule(ADC_CLOCKSOURCE_MCLK,ADC_PREDIVIDER_1,ADC_PREDIVIDER_
1,ADC_NOROUTE);//initModule requires 4 parameters//clock is started //predivider is divding
the clock to make it small//master clock is 3 MHz which is divider by 2 divders // ADC Noroute
means use ADC to connect to that pin // more the predivider more the numbers we can get
                ADC14_configureSingleSampleMode(ADC_MEM6,true);//ADC_MEM6 internal
memory for storing converted result

ADC14_configureConversionMemory(ADC_MEM6,ADC_VREFPOS_AVCC_VREFNEG_VSS,A
DC_INPUT_A6,false);//2nd parameter is 3.3v i.e the refrences voltage  , 3rd para is use that
channel 6 inside MC and store it to the mem 6  , false is it will run only once

ADC14_setSampleHoldTime(ADC_PULSE_WIDTH_32,ADC_PULSE_WIDTH_4);//it will hold
the signal for 64 clock cycle , 2 times becoz it require 2 parameter ; our MC is 32bit so it divided
into 16-16 bits channels 2nd para is useless as of now
                ADC14_setResolution(ADC_12BIT);
                ADC14_enableSampleTimer(ADC_AUTOMATIC_ITERATION);//it will run
automatically
                ADC14_enableModule();//for activating the module
                ADC14_enableConversion();//start conversion
                ADC14_toggleConversionTrigger();// trigger the conversion again and again
}

void loop() {
  // put your main code here, to run repeatedly:
    if(!client.connected())
       {
          Serial.println("Disconnected:Reconnecting...");
          if(!client.connect("subs"))
          {
             Serial.println("connection failed");
          }
          else
          {
             Serial.println("Connection success");
             if(client.subscribe("subscriber"))
```

```
        {
            Serial.println("Subscription sucessfull");
        }
      }
    }

int result,regressedData1;
    float regressedData;
    while(!ADC14_isBusy());
    result=ADC14_getResult(ADC_MEM6);
    //P2OUT=result>>8;
    //Serial.println(result);
    ADC14_toggleConversionTrigger();
    //result*slope+-itercept/1000
    regressedData=((result*0.786295966)+7.695073417)/1000;
    Serial.println(regressedData);
    regressedData1=((result*0.786295966)+7.695073417);
    P2->OUT = result>>8;

    String str=(String)regressedData1;
      int str_len=str.length()+2;
      char char_array[str_len];
      str.toCharArray(char_array,str_len);
      char pot_reading[str_len];

      if(regressedData<1)
      {
         pot_reading[0]='0';
         pot_reading[1]='.';
         for(int i=2;i<=str_len;i++)
         {
            pot_reading[i]=char_array[i-2];
         }
      }
      else
      {
         pot_reading[0]=char_array[0];
         pot_reading[1]='.';
         for(int i=2;i<=str_len;i++)
         {
            pot_reading[i]=char_array[i-1];

         }
      }
```

```
        aJsonObject *root=aJson.createObject();
            aJsonObject *d=aJson.createObject();
            aJson.addItemToObject(root,"d",d);
            aJson.addStringToObject(d,"IbmCloudPot","MSP432");
            aJson.addStringToObject(d,"potValue",pot_reading);
            jsonPayload=aJson.print(root)+'\0';//renders the jso object back to a string;
            aJson.deleteItem(d);
            aJson.deleteItem(root);
            //publish data to MQTT broker
            client.publish("publisher",jsonPayload);
            client.poll();
            delay(1000);
}

void callback(char* inTopic,byte* payload,unsigned int length)
{
   Serial.print("Message arrived on topic:");
   Serial.print(inTopic);
   Serial.print(". Message:");
   String arrivedMessage;
   for(int i=0;i<length;i++)
   {
      Serial.print((char)(payload[i]));
      arrivedMessage+=(char)payload[i];
   }
   Serial.println();
   if(arrivedMessage=="LED1ON")
      GPIO_setOutputHighOnPin(GPIO_PORT_P1,GPIO_PIN0);
   else
      if(arrivedMessage=="LED1OFF")
         GPIO_setOutputLowOnPin(GPIO_PORT_P1,GPIO_PIN0);
}
```



Dashboard for Remotely Controlling LED



Using PuTTY emulator to record the LED changes