Qualcomm Developer Project  
**QCA4020 Home Automation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribution:** |  | | |
| **Email address** | [yanggh0703@thundersoft.com](mailto:yanggh0703@thundersoft.com)  [hanfd0702@thundersoft.com](mailto:hanfd0702@thundersoft.com)  [zhouxiang0419@thundersoft.com](mailto:zhouxiang0419@thundersoft.com) | | |
| **Project Title**\* |  | | |
| **Description**\*  *High level description of the project* ***(75 words or less)*** | This project is designed to use the QCA4020 development kit to control the Sengled Zigbee bulb and Smart lock using an Android mobile phone and demonstrates smart home. You can control the status of the bulb via the app ”AzureConsole”, such as turning it on/off, changing the color temperature and so on. For “Smart\_Lock”, you can control the rotation of servo motor via the app ”AzureConsole” to simulate a smart lock. | | |
| **Images**  *Upload up to 5 images of your project*  *Please submit/send the original JPEG/PNG files for all images included in the document* | **qca4020\_devkit\_130x130.png**  qca4020_devkit_130x130  [alt tag: “**Home Automation using the QCA4020 development board.**”]  **servor\_motor\_130x130.png**  servor_motor_130x130  [alt tag: “**Home Automation using the servor motor.**”]  **zigbee\_lamp\_130x130.png**  Zigbee_Lamp_130x130  [alt tag: “**Home Automation using the zigbee lamp.**”]  **overall\_project\_130x130.png**  overall_project_130x130  [alt tag: “**Home Automation ’s assembly diagram.**”] | | |
| **Objective**   * *What inspired you to create this project?* * *What is your desired outcome?* | The main objective of this demo is to create smart home using the QCA4020 development kit. This is done by using an application on the mobile phone, running Android O, to connect to the QCA4020 and control the Sengled bulb and servo motor. | | |
| **Operation System**\*  (Android, Linux, Windows 10 IoT Core) | Android   Linux  RTOS | | Windows 10 IoT Core   Ubuntu Core |
| **Cloud Services/Platform**  AT&T M2X, AWS IoT, IBM Bluemix, IBM Watson IoT, Such as Microsoft Azure IoT) | Amazon AWS IoT   AT&T M2x   IBM Bluemix | | IBM Watson IoT  Microsoft Azure IoT  Google Cloud Platform |
| **Skill Level Required**  (Beginner, Intermediate, Advanced) | Advanced   Beginner | | Intermediate |
| **Areas of Focus**  (e.g., IoT, smart cities, smart home, robotics, hardware, gaming, healthcare, automotive, digital signage, etc.) | 3D Printing & Modeling   Alexa Voice Service   Bluetooth   Computer Vision   Digital Signage   Education   Embedded    Gaming | | Healthcare   IoT   Robotics  Security   Sensors   Smart Cities   Smart Home   Toys  ZigBee |
| **Materials Required / Parts List / Tools** | Part Name | Link | |
| [QCA4020 board](https://www.qualcomm.com/products/qca4020) | <https://www.qualcomm.com/products/qca4020> | |
| [Sengled bulb](https://www.amazon.com/Element-Plus-Sengled-2700-6500KSmartThings/dp/B01MQVYNFL/ref=sr_1_8?s=hi&ie=UTF8&qid=1523987884&sr=1-8&keywords=sengled+smart+bulb) | <https://www.amazon.com/Element-Plus-Sengled-2700-6500KSmartThings/dp/B01MQVYNFL/ref=sr_1_8?s=hi&ie=UTF8&qid=1523987884&sr=1-8&keywords=sengled+smart+bulb> | |
| [Lamp holder](https://item.m.jd.com/product/31912759280.html?ShareTm=pRYzt%2BY0bcuAYJHKosVgO%2B6g4QMPvCTgHHayaxLuN934yGW0ixCXpnTHCubwkoJjBMTrBiX5fdVZLMg0SLHZpL2ChVLi%2FgNTr2xnl%2Fn0itiT3EK87e6%2B2TteNvaxqWP5KzoZFR8WWiHUeN%2FdSAyycbow9%2FTdxahwR9r1DAg5hmY%3D&ad_od=share&) | <https://item.m.jd.com/product/31912759280.html?ShareTm=pRYzt%2BY0bcuAYJHKosVgO%2B6g4QMPvCTgHHayaxLuN934yGW0ixCXpnTHCubwkoJjBMTrBiX5fdVZLMg0SLHZpL2ChVLi%2FgNTr2xnl%2Fn0itiT3EK87e6%2B2TteNvaxqWP5KzoZFR8WWiHUeN%2FdSAyycbow9%2FTdxahwR9r1DAg5hmY%3D&ad_od=share&> | |
| Adaptor | unspecified | |
| Mobile phone | Android O | |
| Transformer(220v->110v) | unspecified | |
| **Source Code / Source Examples / Application Executable**  *Link to open source / shareable code repository* | Description | Link | |
| [Source Code](https://github.com/canyudeguang/Home_Automation) | <https://github.com/canyudeguang/Home_Automation> | |
|  |  | |
| **Additional Resources**  *List related links or resources such as websites, videos, presentations, or other materials* | Resource Title | Link or File Name (and provide file) | |
| [Video of “Home Automation”](https://pan.baidu.com/s/1dDjGoV6fmwGDlkZc9CzZ0Q) | <https://pan.baidu.com/s/1dDjGoV6fmwGDlkZc9CzZ0Q>  ps：jboo | |
|  |  | |
| **Build / Assembly Instructions** | Parts used Below are the items used in this project.  **parts.png**  parts  [alt tag: “**Parts used for the QCA4020 development kit Home Automation project.**”]   1. Mobile Phone with Android O operating system. 2. QCA4020 development board 3. Zigbee Lamp 4. Transformer/power supply to turn on the Zigbee Lamp. 5. Servo Motor   Deploying the project  1.Download code from the [github repository](https://github.com/canyudeguang/Home_Automation).  2.Compile the code and flash the image as described in “[QCLI\_demo with QCA4020 Development kit](https://developer.qualcomm.com/project/qclidemo-w-qca4020-dev-kit)” project.  3.Install the application found in the [github repository](https://github.com/canyudeguang/Home_Automation) on the Android phone.  How does it work?  The QCA4020 development board controls the on/off status, color temperature and intensity of the bulb via Zigbee protocol, and control the rotation of the servo motor to simulate a smart lock. The sections below will show you how the various controls and interfaces in this demo have been implemented, and how the QCA4020 development kit can control the bulb and servo motor via these interfaces. Initialize Zigbee protocol and build Zigbee network. Global variable "flag\_form" is used to judge whether the Zigbee network is successful or not. Add the following code to the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  int flag\_form = 0;    void init\_Zigbee()  {  d\_cmd\_ZB\_Initialize(0,NULL);    QCLI\_Parameter\_t param\_setBIB[4];  param\_setBIB[0].Integer\_Value = 0x100f;  param\_setBIB[0].Integer\_Is\_Valid = true;  param\_setBIB[1].Integer\_Value = 1;  param\_setBIB[1].Integer\_Is\_Valid = true;  param\_setBIB[2].Integer\_Value = 4;  param\_setBIB[2].Integer\_Is\_Valid = true;  param\_setBIB[3].Integer\_Value = 0;  param\_setBIB[3].Integer\_Is\_Valid = true;  d\_cmd\_ZB\_SetBIB(4, param\_setBIB);    QCLI\_Parameter\_t param\_CE[2];  param\_CE[0].Integer\_Value = 1;  param\_CE[0].Integer\_Is\_Valid = true;  param\_CE[1].Integer\_Value = 6;  param\_CE[1].Integer\_Is\_Valid = true;  d\_cmd\_ZB\_CL\_CreateEndpoint(2, param\_CE);    QCLI\_Parameter\_t param\_form[1];  param\_form[0].Integer\_Value = 1;  param\_form[0].Integer\_Is\_Valid = true;  d\_cmd\_ZB\_Form(1, param\_form);    while(true)  {  if(flag\_form == 1)  {  zigbee\_printf("Form Zigbee network success!");  break;  }  qurt\_thread\_sleep(50);  zigbee\_printf("wait for forming Zigbee network!");  }  } Zigbee devices connection The function "Connect\_ZigbeeDevice()" is to wait for connection of Zigbee bulb, "flag\_connect" is used to judge whether the bulb joins into the network successfully or not. Add the following code to the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  int flag\_connect = 0;  void Connect\_ZigbeeDevice() {  while(true)  {  if(flag\_connect == 1)  {  zigbee\_printf("Device join success!");  break;  }  qurt\_thread\_sleep(50);  zigbee\_printf("wait that blub join in Zigbee network!");  } } Building the Zigbee network "flag\_form" is set to 1 after the network architecture is built successfully. Add the following code to the file “/quartz/demo/QCLI\_demo/src/ zigbee/zigbee\_demo.c”.  extern int flag\_form; static void ZB\_Event\_CB() {  flag\_form = 1; } Zigbee network devices connection We can set "flag\_form" to 1 after the devices connect successfully.  Add the following code to the file “/quartz/demo/QCLI\_demo/src/zigbee/zdp\_demo.c”.  extern int flag\_connect; static void ZB\_ZDP\_Event\_CB(qapi\_ZB\_Handle\_t ZB\_Handle, const qapi\_ZB\_ZDP\_Event\_t \*ZDP\_Event\_Data, uint32\_t CB\_Param) {  QCLI\_Parameter\_t param[3];  param[0].Integer\_Value = 2;  param[0].Integer\_Is\_Valid = true;  param[1].Integer\_Value = ZDP\_Event\_Data?  Event\_Data.Device\_Annce.NwkAddr;  device\_addr = ZDP\_Event\_Data?  Event\_Data.Device\_Annce.NwkAddr;  QCLI\_Printf(ZDP\_Demo\_Context.QCLI\_Handle, NetworkAddress: %d\n", ZDP\_Event\_Data?Event\_Data.Device\_Annce.NwkAddr);  param[1].Integer\_Is\_Valid = true;  param[2].Integer\_Value = 1;  param[2].Integer\_Is\_Valid = true;  d\_cmd\_ZB\_AddDevice(3, param);  flag\_connect = 1; } Turning the Zigbee bulb on/off Add the function "Control\_ZigbeeLightOnOff" to turn the bulb on/off in the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  int Control\_ZigbeeLightOnOff(int on) {  if(on > 1)  {  QCLI\_Parameter\_t param\_on[2];  param\_on[0].Integer\_Value = 1;  param\_on[0].Integer\_Is\_Valid = true;  param\_on[1].Integer\_Value = 1;  param\_on[1].Integer\_Is\_Valid = true;  d\_cmd\_ZCL\_OnOff\_On(2, param\_on);  }  else{  QCLI\_Parameter\_t param\_off[2];  param\_off[0].Integer\_Value = 1;  param\_off[0].Integer\_Is\_Valid = true;  param\_off[1].Integer\_Value = 1;  param\_off[1].Integer\_Is\_Valid = true;  d\_cmd\_ZCL\_OnOff\_Off(2, param\_off);  }   return 0; } Adjust the intensity of the light bulb To dim the light bulb, add the function "Control\_ZigbeeLightLevel" to the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  int Control\_ZigbeeLightLevel(int level) {  QCLI\_Parameter\_t param\_level[5];  param\_level[0].Integer\_Value = 1;  param\_level[0].Integer\_Is\_Valid = true;  param\_level[1].Integer\_Value = 1;  param\_level[1].Integer\_Is\_Valid = true;  param\_level[2].Integer\_Value = 1;  param\_level[2].Integer\_Is\_Valid = true;  param\_level[3].Integer\_Value = level;  param\_level[3].Integer\_Is\_Valid = true;  param\_level[4].Integer\_Value = 5;  param\_level[4].Integer\_Is\_Valid = true;  d\_cmd\_ZCL\_LevelControl\_MoveToLevel(5, param\_level);   return 0; } Adjust color temperature of the light bulb To control the color temperature of the bulb, add the function "Control\_ZigbeeColorTemperature" to the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  int Control\_ZigbeeColorTemperaturn(int temp) {  QCLI\_Parameter\_t param\_temp[4];  param\_temp[0].Integer\_Value = 1;  param\_temp[0].Integer\_Is\_Valid = true;  param\_temp[1].Integer\_Value = 1;  param\_temp[1].Integer\_Is\_Valid = true;  param\_temp[2].Integer\_Value = temp;  param\_temp[2].Integer\_Is\_Valid = true;  param\_temp[3].Integer\_Value = 5;  param\_temp[3].Integer\_Is\_Valid = true;  d\_cmd\_ZCL\_ColorControl\_MoveToColorTemp(4, param\_temp);  return 0; } Saving Internet address of bulb Here is how to save the network address of the bulb, the code is located in the file "/quartz/demo/QCLI\_demo/src/zigbee/zigbee\_demo.c".  uint64\_t device\_addr = 0;  static QCLI\_Command\_Status\_t cmd\_ZB\_AddDevice(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {  (Verify\_Integer\_Parameter(&Parameter\_List[0], QAPI\_ZB\_ADDRESS\_MODE\_GROUP\_ADDRESS\_E, QAPI\_ZB\_ADDRESS\_MODE\_EXTENDED\_ADDRESS\_E)) &&  (Hex\_String\_To\_ULL(Parameter\_List[1].String\_Value, &DevAddr)))  (Verify\_Integer\_Parameter(&Parameter\_List[0], QAPI\_ZB\_ADDRESS\_MODE\_GROUP\_ADDRESS\_E,API\_ZB\_ADDRESS\_MODE\_EXTENDED\_ADDRESS\_E))/\*&&(Hex\_String\_To\_ULL(Parameter\_List[1].String\_Value, &DevAddr))\*/)  {  DevAddr = device\_addr;   } } Function encapsulation We need to do function encapsulation process. The code related to color temperature is located in the file "/quartz/demo/QCLI\_demo/src/zigbee/clusters/zcl\_colorcontrol\_demo.c".  QCLI\_Command\_Status\_t d\_cmd\_ZCL\_ColorControl\_MoveToColorTemp(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List);    QCLI\_Command\_Status\_t d\_cmd\_ZCL\_ColorControl\_MoveToColorTemp(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {  return cmd\_ZCL\_ColorControl\_MoveToColorTemp(Parameter\_Count, Parameter\_List);  }  The code related to level control is located in the file "/quartz/demo/QCLI\_demo/src/zigbee/clusters/zcl\_levelcontrol\_demo.c".  QCLI\_Command\_Status\_t d\_cmd\_ZCL\_LevelControl\_MoveToLevel(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List); QCLI\_Command\_Status\_t d\_cmd\_ZCL\_LevelControl\_MoveToLevel(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {  return cmd\_ZCL\_LevelControl\_MoveToLevel(Parameter\_Count, Parameter\_List);  }  param\_temp[3].Integer\_Is\_Valid = true;  d\_cmd\_ZCL\_ColorControl\_MoveToColorTemp(4, param\_temp);  return 0;  The code related to color on or off status is located in the file "/quartz/demo/QCLI\_demo/src/zigbee/clusters/zcl\_onoff\_demo.c"  QCLI\_Command\_Status\_t d\_cmd\_ZCL\_OnOff\_On(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List);  QCLI\_Command\_Status\_t d\_cmd\_ZCL\_OnOff\_Off(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List);    QCLI\_Command\_Status\_t d\_cmd\_ZCL\_OnOff\_On(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {    return cmd\_ZCL\_OnOff\_On(Parameter\_Count, Parameter\_List);  }    QCLI\_Command\_Status\_t d\_cmd\_ZCL\_OnOff\_Off(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {  return cmd\_ZCL\_OnOff\_Off(Parameter\_Count, Parameter\_List);  }  The code related to endpoint creation is located at "/quartz/demo/QCLI\_demo/src/zigbee/zcl\_demo.c".  QCLI\_Command\_Status\_t d\_cmd\_ZB\_CL\_CreateEndpoint(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List);    QCLI\_Command\_Status\_t d\_cmd\_ZB\_CL\_CreateEndpoint(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List)  {  return cmd\_ZB\_CL\_CreateEndpoint(Parameter\_Count, Parameter\_List);  }  Encapsulate below functions "d\_cmd\_ZB\_AddDevice", “d\_cmd\_ZB\_AddDevice", "d\_cmd\_ZB\_SetBIB", "d\_cmd\_ZB\_Initialize".  QCLI\_Command\_Status\_t d\_cmd\_ZB\_AddDevice(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List); QCLI\_Command\_Status\_t d\_cmd\_ZB\_AddDevice(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List); QCLI\_Command\_Status\_t d\_cmd\_ZB\_SetBIB(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List); QCLI\_Command\_Status\_t d\_cmd\_ZB\_Initialize(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List); QCLI\_Command\_Status\_t d\_cmd\_ZB\_Initialize(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List) { return cmd\_ZB\_Initialize(Parameter\_Count, Parameter\_List); } QCLI\_Command\_Status\_t d\_cmd\_ZB\_AddDevice(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List) {  return cmd\_ZB\_AddDevice(Parameter\_Count, Parameter\_List); } QCLI\_Command\_Status\_t d\_cmd\_ZB\_Form(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List) {  return cmd\_ZB\_Form(Parameter\_Count, Parameter\_List); } QCLI\_Command\_Status\_t d\_cmd\_ZB\_SetBIB(uint32\_t Parameter\_Count, QCLI\_Parameter\_t \*Parameter\_List) {  return cmd\_ZB\_SetBIB(Parameter\_Count, Parameter\_List); } Control servo motor through PWM Servo motor will rotate different angles at different PWM situations.In this project, we use the angle “-45” and “0” to simulate the lock “open” and “close”. To do this, add the function "open\_lock" and “close\_lock” to the file “/quartz/demo/QCLI\_demo/src/qcli/pal.c”.  void open\_lock()  {  QCLI\_Parameter\_t param[5];  param[0].Integer\_Value = 5000;  param[0].Integer\_Is\_Valid = true;  param[1].Integer\_Value = 500;  param[1].Integer\_Is\_Valid = true;  param[2].Integer\_Value = 2000;  param[2].Integer\_Is\_Valid = true;  param[3].Integer\_Value = 1;  param[3].Integer\_Is\_Valid = true;  param[4].Integer\_Value = 1;  param[4].Integer\_Is\_Valid = true;  pwm\_driver\_test(5, param);  }  void close\_lock()  {  QCLI\_Parameter\_t param[5];  param[0].Integer\_Value = 5000;  param[0].Integer\_Is\_Valid = true;  param[1].Integer\_Value = 750;  param[1].Integer\_Is\_Valid = true;  param[2].Integer\_Value = 2000;  param[2].Integer\_Is\_Valid = true;  param[3].Integer\_Value = 1;  param[3].Integer\_Is\_Valid = true;  param[4].Integer\_Value = 1;  param[4].Integer\_Is\_Valid = true;  pwm\_driver\_test(5, param);  }  Now that you understand how controlling the light bulb is implemented in the project, below are some usage instructions to test the project. | | |
| **Usage Instructions** | 1. Power on the QCA4020 via the power button, and the QCA4020 will build the Zigbee network automatically. 2. Power on the Zigbee bulb, and it will join the Zigbee network automatically. 3. Use the “Azure\_console” app on the mobile phone to:    1. Control on or off status of bulb via clicking “Open” or “Close” button    2. Control color temperature via setting the values ranged from “0x0000~0xFFFF”    3. Control dim via setting the values ranged from “0~255”    4. Control servo motor via clicking “Open” or “Close” button | | |
| **Contributor(s) Info**  *Feel free to include headshots!* | Name | Title  Company | |
| Yang  [yanggh0703@thundersoft.com](mailto:yanggh0703@thundersoft.com) | Thundersoft | |
| Jay  [zhouxiang0419@thundersoft.com](mailto:zhouxiang0419@thundersoft.com) | Thundersoft | |
| Scott  [hanfd0702@thundersoft.com](mailto:hanfd0702@thundersoft.com) | Thundersoft | |

*By submitting your content (“Submission”), you are granting Qualcomm a royalty-free, perpetual, non-exclusive, unrestricted, worldwide license to: (a) post, use, copy, sublicense, adapt, transmit, publicly perform or display any such Submission, (b) use, reproduce, modify, adapt, publish, translate, create derivative works from, distribute, perform, play, host, communicate, make available and publish your Submission without restriction and (c) sublicense to third parties the unrestricted right to exercise any of the foregoing rights granted with respect to the Submission. The foregoing grants shall include the right to exploit any ideas, concepts, intellectual property, or proprietary rights in such Submission, including but not limited to rights under copyright, trademark, servicemark or patent laws under any relevant jurisdiction without Qualcomm owing any monies to you whatsoever. You represent and warrant that you own all right, title and interest in and to the Submission, or you have been granted sufficient rights in and to the Submission allowing the foregoing use of such Submission.*