A Secure Memory Interface for Smart Home Application

Shriram M
Student
Dept. of Electronics and Communication Engineering,
R V College of Engineering
Bengaluru, India

Deekshit B N
Student
Dept. of Electronics and Communication Engineering,
R V College of Engineering
Bengaluru, India

Roopa J
Assistant Professor
Dept. of Electronics and Communication Engineering,
R V College of Engineering
Bengaluru, India

Jaya Kathuria Bindra
Manager Applications Engineering,
Cypress Semiconductor
Technology India Pvt. Ltd.
Bengaluru, India

***Abstract*— Memory devices have been indispensable components of embedded systems. As many embedded devices are portable, they are battery-powered. As a result, the components of these systems need to optimize their power consumption both in hardware and firmware. External memory devices are instrumental in reducing the power consumption and the die size of microcontrollers in embedded devices. With the rapid increase in the number of connected devices in the world of Internet of Things (IoT), the necessity of security in IoT devices has increased. The paper presents a secure memory interface using Cypress Semiconductor's PSoC 6 device. The proposed system implements the memory interface in a Smart Home network that is integrated with the Amazon Web Services (AWS) IoT Core cloud platform.**

Keywords*— *Memory interface, IoT, PSoC, AWS IoT Core, MQTT, Smart Home

I. INTRODUCTION

Recent advancements in semiconductor technology have given rise to cost-effective solutions which directly integrate wireless connectivity in embedded processors and sensors. This leads to a great interest in the Internet of Things (IoT), defined as the interconnection of everyday objects to the internet. The IoT is now considered to be a ready technology for the consumer electronics market [1]. Fields like the smart home and the smart city have been considered as the market segments with very high potential for IoT deployment, which give rise to many emerging processes such as home automation and energy management.

Smart Home is one of the applications that is rapidly growing in the IoT market segment. Smart Home devices are gaining popularity in the past few years owing to their potential applications that makes day-to-day life very easy and productive for the customers. With Smart Home IoT applications, there is a need for security, low power consumption and seamless wireless connectivity [2]. Cypress Semiconductor's latest Programmable System-On-Chip – PSoC 6 offers the industry's lowest power and extremely flexible Microcontroller Unit (MCU) architecture. It is uniquely designed and developed for battery-powered and secure IoT devices.

The memory devices for embedded systems can be either internally built with the microcontroller unit (MCU) or externally interfaced to the MCUs. The necessary amount of memory for storing the firmware and data will be available in the internal memory. The external memory devices provide flexibility in the design of complex systems. They help in the low power consumption of the overall system. They also help in reduction of the die size of the MCUs.

There are several memory interfacing schemes to integrate external memory devices with the core MCUs. The proposed system in this paper implements a serial memory interface (SMIF) with cryptographic functions for security in embedded devices.

II. TYPES OF MEMORY INTERFACES

Embedded systems typically use various kinds of memory devices for various tasks such as storage of firmware, user data, debug verbose, etc. Generally, an embedded Central Processing Unit (CPU) comprises of two types of memory devices built into the system which are the Random Access Memory (RAM) and the Read Only Memory (ROM). There are hybrid memory devices which serve as secondary memory devices which may be present outside the CPU system.

The primary memory of an embedded system often comprises of the RAM and the ROM built into the CPU which is often packaged within the microcontroller. The RAM is a volatile memory device which has higher read and write speeds which are comparable to that of the CPU. The ROM, on the other hand, is a non-volatile memory that is used to store critical data. The choice of the size and speed of the RAM and ROM depends on the application of the embedded system. Apart from these built-in primary memory devices, external / secondary memory devices are used in an embedded system to provide additional memory capacity for storage of user data, debug information, etc.

The Secondary memory of embedded devices are used for long term storage of data. It needs to be non-volatile in nature in order to store data in the absence of power. The Electrically erasable programmable read only memory (EEPROM) has been the most popular secondary memory device in embedded systems. EEPROMs offer very simple and inexpensive ways to expand the memory capacity of the embedded system. A relatively new technology called the Flash memory technology then dominated the memory industry. The Flash memory devices were very inexpensive and offered very easy and simple method of interfacing with the CPU. They can be interfaced either in serial or parallel.

The memory devices with parallel interfaces offer faster data transfer as they utilize multiple data lines and hence provide superior performance. But the issue with the parallel memory devices is that they require a greater number of General Purpose Input-Output (GPIO) pins. Another disadvantage is that such memory devices typically have a significantly larger footprint. Hence, it is very complicated to use parallel memory devices in large designs as interfacing multiple such chips becomes very difficult.

The serial memory devices, on the other hand, offer very simple interfacing and the number of GPIO pins used is very less. The actual number of pins used depends on the serial protocol used. Most memory manufacturers offer serial memory devices that support protocols like I2C, SPI and Microwire. The MCUs generally provide support for multiple protocols to offer flexibility in designs.

The SPI-based memory devices offer reasonable interface speed for data logging and other storage applications. But higher memory access speeds are needed for direct external code execution. This allows the MCU to execute firmware from the external memory device. These requirements have forced MCU manufacturers to provide alternatives to SPI such as the Dual-SPI, Quad-SPI, and Octal-SPI.

The proposed memory interface implements a Quad-SPI memory device with Cypress Semiconductor’s Programmable System-on-Chip (PSoC) 6 device. The overview of the PSoC 6 is described in the next section.

III. OVERVIEW OF THE PSOC 6

The Programmable System-on-chip (PSoC) is a flexible, scalable and programmable platform architecture which encompasses a family of configurable embedded system microcontrollers with Arm Cortex Central Processing Units (CPU) [3]. The PSoC 6 product family, optimized for ultra-low power consumption by the use of 40-nm platform, is a culmination of Arm MCU with digital programmable logic, flash technology, digital-to-analog and analog-to-digital conversion, standard communication and timing peripherals. The PSoC 6 also provides wireless connectivity with the use of Bluetooth Low Energy (BLE) technology.

The PSoC 6 comprises of many peripheral hardware blocks like the Serial Communication Block (SCB) which can be configured to implement Universal Asynchronous Receiver-Transmitter (UART), Inter-Integrated Circuit (I2C) and Serial Peripheral Interface (SPI) protocols. A BLE subsystem is provided which is BLE 5.0 compliant which is also known as Bluetooth Smart.

There is a timing and pulse-width modulation (PWM) block which can implement timer, counter and PWM functions. There are 78 programmable General Purpose Input-Output (GPIO) pins which can have programmable drive modes, strengths and slew rates. There are other peripheral blocks such as the audio subsystem, programmable 12-bit analog block with analog-to-digital and digital-to-analog convertors. The PSoC 6 also features a CapSense block that enables capacitive sensing with automatic hardware tuning features.

IV. DESIGN AND IMPLEMENTATION

The section presents the overview of the SMIF hardware block of the PSoC 6, implementation of SMIF for Smart Home application, architecture of the proposed Smart Home network and its integration with the AWS cloud service.

A. Overview of the PSoC 6 SMIF block

The PSoC 6 features several peripheral hardware blocks apart from the dual-core MCU architecture. The hardware blocks function concurrently in conjunction with the dual Arm cores, but the hardware is configured and setup by the MCUs. The function of these blocks can be dynamically controlled during runtime.

The Serial Memory Interface is one of the peripheral hardware blocks of PSoC 6 that provides an interface between PSoC 6 MCU and external memory devices. The SMIF block is capable of interfacing memory devices that support single-SPI, dual-SPI, quad-SPI or octal-SPI. Its primary use case is to setup the external memory devices and use the hardware to map it to the PSoC 6 MCU memory space. The SMIF operates in two modes – Memory Mapped Input-Output (MMIO) and eXecute In Place (XIP).

The MMIO mode enables the SMIF to act as a simple communication block for transfer of SPI commands for data read or write operations. The XIP mode is an important mode of operation that allows the bus masters in PSoC 6 to interact directly with the SMIF for memory related operations as the external memory is mapped to the internal memory address space of PSoC 6. In order to enable faster access of external memory in XIP mode, the PSoC 6 provides 4-KB read cache in XIP mode. The SMIF block provides on-the-fly 128-bit encryption and decryption by implementing the Advanced Encryption Standard (AES) algorithm in hardware. The advantage of having on-the-fly encryption and decryption is that there is no wastage of CPU time in executing the AES algorithms as the encryption and decryption is performed by a hardware block which runs concurrently along with the Arm MCUs.

B. Implementation of SMIF for Smart Home application

An external NOR flash memory is interfaced with the PSoC 6 with the help of the SMIF block using the Quad-SPI (QSPI) interface. The interfacing diagram of the external memory device is as shown in Fig. 1. The device uses four data lines along with clock and chip select lines.

The developed firmware makes use of the different SPI commands, as specified by the datasheet of the memory device, to execute memory read, write and erase operations. The firmware operates the SMIF block in MMIO mode to write and erase data on the flash memory and XIP mode to read data from the memory.

The project aims to implement a secure memory interface for Smart Home application. To address the security concerns, the SMIF provides a cryptography block. The firmware is developed to utilize this security feature and it implements AES–128 encryption algorithm. The memory interface is now secure and ensures data integrity. For the required IoT application, a data logging system to periodically log the sensor and verbose data with encryption feature is developed.

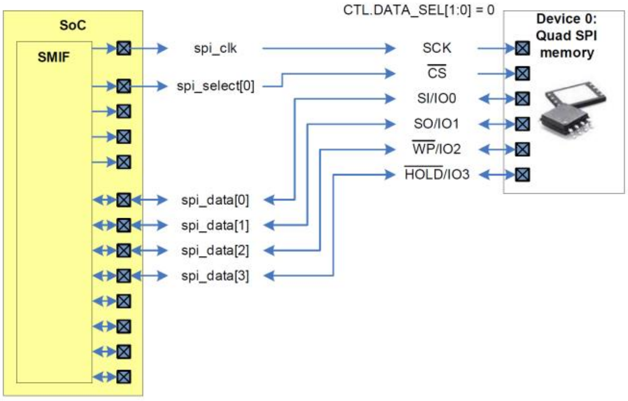


Fig. 1. Connection diagram of the memory device with PSoC 6 device [4]

C. *Smart Home architecture*

The IoT systems comprise of numerous end devices that are either directly or indirectly connected to the Internet with the help of gateways. The project implements a Smart Home network comprising of three IoT enabled end devices that will communicate with a gateway which interacts with a secure cloud database. The block diagram of the Smart Home architecture is as shown in Fig. 2.

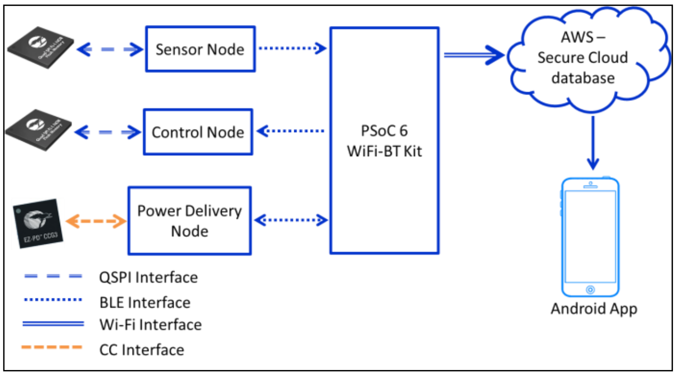


Fig. 2. Architecture of the proposed Smart Home network

The proposed architecture shows three IoT nodes – a sensor node, a control node and a power delivery node. These nodes form a network by communication with a gateway. The gateway in this architecture is the PSoC 6 WiFi-BT prototyping board which features both Wi-Fi and Bluetooth radio connectivity. The communication between the nodes and the gateway is established through Bluetooth low Energy (BLE) technology. The gateway interacts with the AWS secure cloud database using Wi-Fi technology. The data from the cloud database can be retrieved with the help of a Graphical User Interface (GUI) which can be a web interface or an Android app.

The sensor node comprises of three fundamental components. They are the sensor, processing unit, and radio module for communication. The PSoC 6 development kit includes the PSoC 6 MCU device that supports BLE which is the preferred communication technology in the proposed project. The PSoC 6 development kit covers the processing and radio communication units. The sensor that is employed in the project is the temperature sensor. A thermistor is used as the sensing element. Its resistance depends on the temperature; generally, the resistance decreases with rise in temperature. The resistance is measured using voltage divider concept. The node monitors the temperature at frequent intervals of time and logs the data in the external flash memory using the designed secure memory interface. At specified intervals, the logged data is fetched and sent over the BLE link to the gateway for cloud storage [5].

The control node is developed as a proof of concept that the data can be sent from the user to the Smart Home network. Hence, a simple indicator like a Light Emitting Diode (LED) is controlled which shows switching of any appliance. The PSoC 6 MCU periodically listens to the BLE link for commands. If the user sends the LED control command through the GUI, the required change is reflected in the AWS cloud database and then the gateway is notified by the MQTT broker [6]. The gateway then pings the control node though BLE. As the control node is checking for commands from the gateway, it receives the ping and then processes it to control the state of the LED.

The third Smart Home node is the fast charging node using which the devices are charged at higher speeds. The process of transferring higher power to end devices is known as Power Delivery in this context. In the project, a USB Type-C port is employed to charge the other Type-C devices at a higher power rating when compared to conventional USB Type-A and Type-B ports. Power delivery is implemented using Cypress' CY8CKIT-062-BLE PSoC 6 development board which is also called the PSoC 6 BLE Pioneer kit.

D. *Integration of the Smart Home network with the AWS cloud service*

The Amazon Web Services (AWS) offers numerous enterprise cloud-based services that are used by many organizations for secure cloud interface. The AWS IoT Core is one such cloud service that enables the connected devices to interact with cloud applications and other devices in a simple and secure manner. It can support billions of devices and trillions of messages. It can also process and route the messages to AWS endpoints and other devices. It also offers interoperability with multiple AWS services like Amazon Kinesis, AWS Lambda, Amazon S3, Amazon DynamoDB, and CloudWatch. The main application of the AWS IoT Core platform is to build IoT solutions that acquire, process, analyze and act on the data generated by connected "things", without having to manage any infrastructure.

The integration of the Smart Home network with the AWS cloud platform needs a user account which can be created for free from the AWS webpage and it offers a free tier mode where up to sixty AWS services can be evaluated for up to twelve months. In case of the AWS IoT Core service, there is an upper limit on the number of messages that can be transferred through the AWS cloud database. The AWS IoT Core service implements an MQTT broker and any other device can communicate with this broker with the help of root certificate authority (CA) certificate, RSA private key, and RSA public key. Once the MQTT client is connected to the AWS server, it can subscribe and publish messages under specified topics [7]. The MQTT publish subscribe architecture is as shown in Fig. 3.

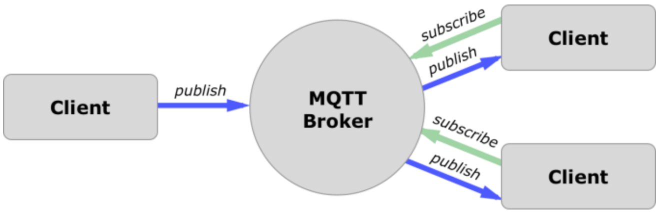


Fig. 3. MQTT Publish Subscribe architecture

The firmware for the PSoC 6 gateway needs to support the MQTT protocol and it contains all the necessary authentication parameters in order to establish communication with the MQTT broker in the AWS server. The gateway needs to publish data that arrives from the sensor node and it needs to subscribe to a topic on which the user publishes commands that operate the control node. Additionally, the PSoC 6 gateway device may publish messages on other topics to send system details and debug information in case of failure.

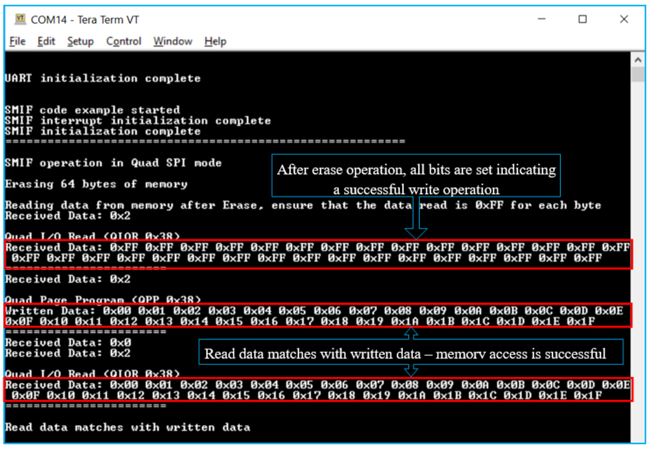
V. RESULTS

The section presents the implementation results of the project. The project is split into implementation of the memory interface, formation of Smart Home network, and the integration of the network with the AWS IoT Core cloud platform.

A. Memory Interface with cryptography

The results of the memory interface tests are as shown in Fig. 4. The test is carried out by first erasing a specified amount of data and then reading the same address to check if all the erased bits are set, i.e. the erased bytes should be 0xFF. If any bit is reset, there is a failure in SMIF implementation. In the project, all the bits were set, as shown in Fig. 4, and it can be concluded that the erase command works without any errors.

The second part of the test is used to verify the write functionality. A specified sequence of bytes (0x00, 0x01, 0x02, 0x03 …, 0x1F in this case) is written using the Quad page program obtained is compared against the written data. As shown in Fig. 4, the read data matches with the written data. Hence, the SMIF functionality in MMIO mode has been implemented successfully.



```
UART initialization complete

SMIF code example started
SMIF interrupt initialization complete
SMIF initialization complete
============================================================

SMIF operation in Quad SPI mode

Erasing 64 bytes of memory

Reading data from memory after Erase, ensure that the data read is 0xFF for each byte
Received Data: 0x2

Quad I/O Read (QIOR 0x38)
Received Data: 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF
 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF
=======================
Received Data: 0x2

Quad Page Program (QPP 0x38)
Written Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
=======================
Received Data: 0x0
Received Data: 0x2

Quad I/O Read (QIOR 0x38)
Received Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
 0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
=======================

Read data matches with written data
```

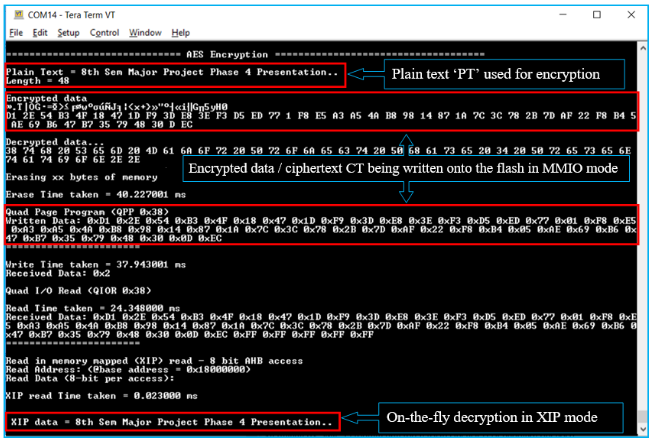
After erase operation, all bits are set indicating a successful write operation

Read data matches with written data – memorv access is successful

Fig. 4. Test results for SMIF operation in MMIO mode

After successful implementation of the basic SMIF functionality, the memory interface was extended for IoT application. Data logging system and cryptographic features were implemented during the process.

The AES-128 cryptographic algorithm is employed for encryption and it is implemented in hardware in PSoC 6. The encrypted data is written into the flash memory using SMIF operating in MMIO mode. The decryption, on the other hand, happens automatically in XIP mode without any additional instructions. This feature is called “On-the-fly decryption”. The functional test results for the AES encryption and on-the-fly decryption are as shown in Fig. 5.



```
============================ AES Encryption ====================================
Plain Text = 8th Sem Major Project Phase 4 Presentation..
Length = 48

Encrypted data
».T|ÖG·=Σ)≤ ╒øw°σúÑJ╕¦<x+)»"°┤«i╢|Gη5yH0
D1 2E 54 B3 4F 18 47 1D F9 3D E8 3E F3 D5 ED 77 1 F8 E5 A3 A5 4A B8 98 14 87 1A 7C 3C 78 2B 7D AF 22 F8 B4 5
 AE 69 B6 47 B7 35 79 48 30 D EC

Decrypted data...
38 74 68 20 53 65 6D 20 4D 61 6A 6F 72 20 50 72 6F 6A 65 63 74 20 50 68 61 73 65 20 34 20 50 72 65 73 65 6E
74 61 74 69 6F 6E 2E 2E

Erasing xx bytes of memory

Erase Time taken = 40.227001 ms

Quad Page Program (QPP 0x38)
Written Data: 0xD1 0x2E 0x54 0xB3 0x4F 0x18 0x47 0x1D 0xF9 0x3D 0xE8 0x3E 0xF3 0xD5 0xED 0x77 0x01 0xF8 0xE5
 0xA3 0xA5 0x4A 0xB8 0x98 0x14 0x87 0x1A 0x7C 0x3C 0x78 0x2B 0x7D 0xAF 0x22 0xF8 0xB4 0x05 0xAE 0x69 0xB6 0x
47 0xB7 0x35 0x79 0x48 0x30 0x0D 0xEC
=======================

Write Time taken = 37.943001 ms
Received Data: 0x2

Quad I/O Read (QIOR 0x38)

Read Time taken = 24.348000 ms
Received Data: 0xD1 0x2E 0x54 0xB3 0x4F 0x18 0x47 0x1D 0xF9 0x3D 0xE8 0x3E 0xF3 0xD5 0xED 0x77 0x01 0xF8 0xE
5 0xA3 0xA5 0x4A 0xB8 0x98 0x14 0x87 0x1A 0x7C 0x3C 0x78 0x2B 0x7D 0xAF 0x22 0xF8 0xB4 0x05 0xAE 0x69 0xB6 0
x47 0xB7 0x35 0x79 0x48 0x30 0x0D 0xEC 0xFF 0xFF 0xFF 0xFF 0xFF
=======================

Read in memory mapped (XIP) read - 8 bit AHB access
Read Address: (@base address = 0x18000000)
Read Data (8-bit per access):

XIP read Time taken = 0.023000 ms

 XIP data = 8th Sem Major Project Phase 4 Presentation..
```

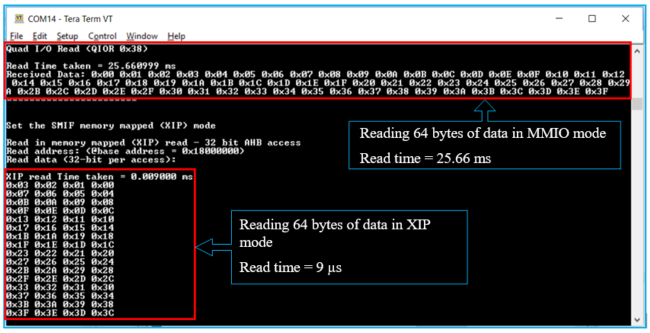
Plain text ‘PT’ used for encryption

Encrypted data / ciphertext CT being written onto the flash in MMIO mode

On-the-fly decryption in XIP mode

Fig. 5. Implementation of AES Encryption and on-the-fly decryption

The overall performance of the system was evaluated by observing the read, write and erase times for the memory for varying packet sizes. Particularly, the comparison between the read times of various packet sizes between MMIO and XIP mode shows that using XIP mode for read operation can improve the speeds by an order of magnitude of 3. The result for the memory access for 64 bytes in MMIO and XIP modes is as shown in Fig. 6.



```
Quad I/O Read (QIOR 0x38)

Read Time taken = 25.660999 ms
Received Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10 0x11 0x12
 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29
A 0x2B 0x2C 0x2D 0x2E 0x2F 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
=======================

Set the SMIF memory mapped (XIP) mode

Read in memory mapped (XIP) read - 32 bit AHB access
Read address: (@base address = 0x18000000)
Read data (32-bit per access):

XIP read Time taken = 0.009000 ms
0x03 0x02 0x01 0x00
0x07 0x06 0x05 0x04
0x0B 0x0A 0x09 0x08
0x0F 0x0E 0x0D 0x0C
0x13 0x12 0x11 0x10
0x17 0x16 0x15 0x14
0x1B 0x1A 0x19 0x18
0x1F 0x1E 0x1D 0x1C
0x23 0x22 0x21 0x20
0x27 0x26 0x25 0x24
0x2B 0x2A 0x29 0x28
0x2F 0x2E 0x2D 0x2C
0x33 0x32 0x31 0x30
0x37 0x36 0x35 0x34
0x3B 0x3A 0x39 0x38
0x3F 0x3E 0x3D 0x3C
```

Reading 64 bytes of data in MMIO mode

Read time = 25.66 ms

Reading 64 bytes of data in XIP mode

Read time = 9 µs

Fig. 6. Memory access times in MMIO and XIP modes

Since the XIP mode maps the external memory to PSoC’s internal address space, it has significantly higher memory access speeds when compared to the MMIO mode. For example, to access 64 bytes of data, it takes about 26 ms in MMIO mode whereas it takes just 9 µs in XIP mode as shown in Fig. 6.

B. Smart Home network

There were three IoT nodes that were integrated to form a Smart Home network with the help of a central gateway device that will bridge the communication between the Smart Home nodes and the secure cloud database. The hardware setup for the Smart Home architecture is as shown in Fig. 7.

The sensor node is implemented using the PSoC 6 BLE Pioneer kit which uses a thermistor as the sensing element. The PSoC 6 prototyping kit is used as the control node and the PSoC 6 WiFi-BT Pioneer kit acts as the power delivery node. The PSoC 6 4343W prototyping kit is the gateway device for the Smart Home network.

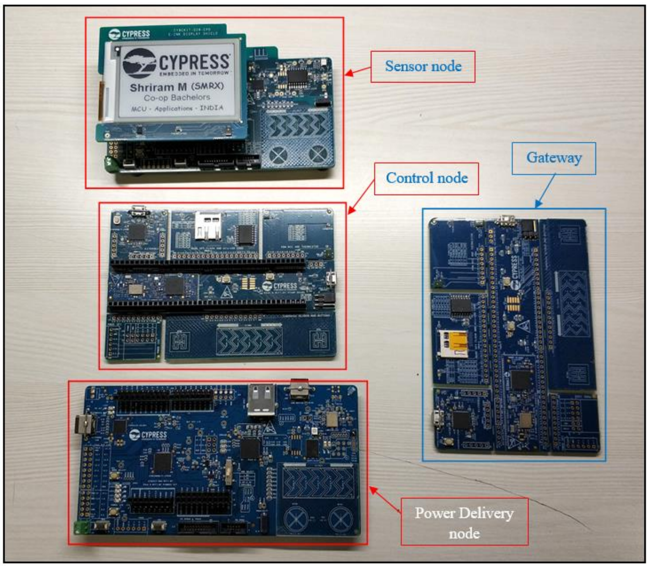
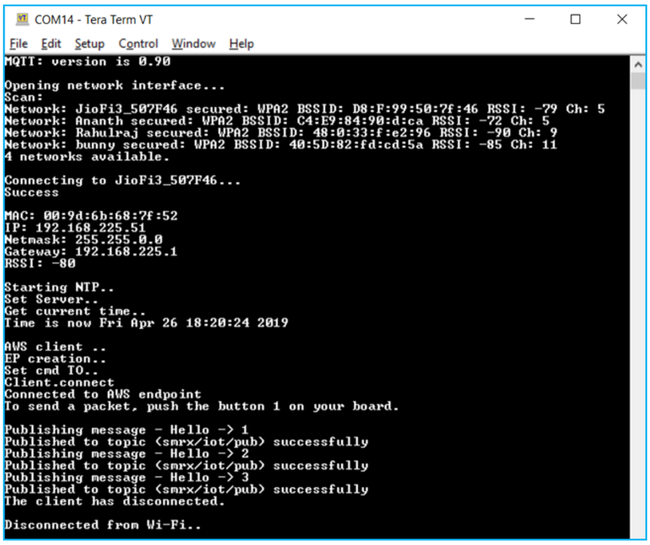


Fig. 7. Hardware setup for the Smart Home system

C. *Integration of Smart Home network with AWS cloud*

The end devices communicate with the gateway using BLE technology and the gateway communicates with the AWS IoT Core cloud platform with the help of Wi-Fi. The gateway implements MQTT protocol to co-ordinate transfer of messages with the AWS IoT cloud database. The results of the integration of the Smart Home network with the cloud platform are as shown in Fig. 8 and Fig. 9.

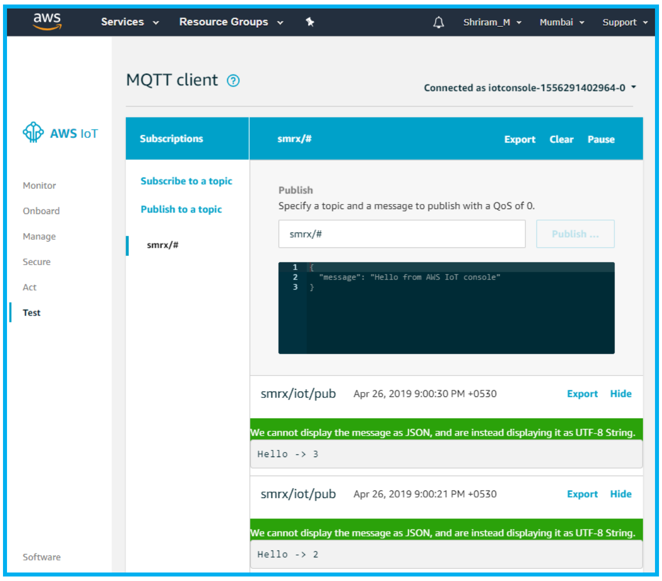
The Fig. 8 shows the serial output data from the PSoC 6 gateway acting as an AWS IoT client. It is to be noted that the messages like “Hello -> 1” are published by the client with the topic “smrx/iot/pub”.



```
MQTT: version is 0.90

Opening network interface...
Scan:
Network: JioFi3_507F46 secured: WPA2 BSSID: D8:F:99:50:7f:46 RSSI: -79 Ch: 5
Network: Ananth secured: WPA2 BSSID: C4:E9:84:90:d:ca RSSI: -72 Ch: 5
Network: Rahulraj secured: WPA2 BSSID: 48:0:33:f:e2:96 RSSI: -90 Ch: 9
Network: bunny secured: WPA2 BSSID: 40:5D:82:fd:cd:5a RSSI: -85 Ch: 11
4 networks available.

Connecting to JioFi3_507F46...
Success

MAC: 00:9d:6b:68:7f:52
IP: 192.168.225.51
Netmask: 255.255.0.0
Gateway: 192.168.225.1
RSSI: -80

Starting NTP..
Set Server..
Get current time..
Time is now Fri Apr 26 18:20:24 2019

AWS client ..
EP creation..
Set cmd TO..
Client.connect
Connected to AWS endpoint
To send a packet, push the button 1 on your board.

Publishing message - Hello -> 1
Published to topic <smrx/iot/pub> successfully
Publishing message - Hello -> 2
Published to topic <smrx/iot/pub> successfully
Publishing message - Hello -> 3
Published to topic <smrx/iot/pub> successfully
The client has disconnected.

Disconnected from Wi-Fi..
```

Fig. 8. Serial output data from PSoC 6 AWS IoT client

In order to verify whether the MQTT client has successfully published the message or not, a subscriber is dynamically created from the AWS IoT Core web interface. The results obtained from the AWS IoT Core webpage are as shown in Fig. 9. The subscriber will receive the messages that are under the subscribed topics.



MQTT client

Connected as iotconsole-1556291402964-0

smrx/#

Publish

Specify a topic and a message to publish with a QoS of 0.

smrx/#

```
{
  "message": "Hello from AWS IoT console"
}
```

smrx/iot/pub Apr 26, 2019 9:00:30 PM +0530

We cannot display the message as JSON, and are instead displaying it as UTF-8 String.

```
Hello -> 3
```

smrx/iot/pub Apr 26, 2019 9:00:21 PM +0530

We cannot display the message as JSON, and are instead displaying it as UTF-8 String.

```
Hello -> 2
```

Fig. 9. MQTT data log from AWS IoT Core service webpage

In the project, the gateway is subscribed to the topic on which the user sends commands to the control node and the end user is subscribed to the topics on which the sensor and power delivery nodes publish data. The user can also subscribe to the topics on which the system information is sent in order to debug the system.

VI. CONCLUSION AND FUTURE SCOPE

With the advent of the IoT, there are plenty of embedded devices that are connected to the Internet. With the current infrastructure, it is feasible only for a fraction of those IoT devices to be directly connected to the Internet. Hence, other devices need to communicate with a central device which can act as a gateway between the device and the Internet. As these end devices are typically battery operated, they need to consume very less power. In order to optimize power consumption and reduce the die size of the MCU, external memory interfaces are very essential for embedded IoT systems for a variety of applications.

The project implements a secure memory interface for Smart Home application. The interface has been designed to be reliable and secure by implementing AES cryptographic algorithm. With the designed memory interface, the memory read time was obtained to be 2 – 10 µs and an improvement of about 1000 times over the conventional SPI memory read operations was achieved. A Smart Home network has been implemented by applying the secure memory. The IoT nodes form a network with a gateway and then the data is communicated with AWS IoT Core database. The user can then interact with the network using a web interface.

Development of a Graphical User Interface (GUI) would greatly improve the user experience with the Smart Home. It provides the scope for future work where the Smart Home experience is enhanced, and many home appliances could be integrated with the Smart Home network.

REFERENCES

[1] F. K. Santoso and N. C. H. Vun, "Securing IoT for smart home system," 2015 International Symposium on Consumer Electronics (ISCE), Madrid, 2015, pp. 1-2.

[2] P. K. Chouhan, S. McClean and M. Shackleton, "Situation Assessment to Secure IoT Applications," 2018 Fifth International Conference on Internet of Things: Systems, Management and Security, Valencia, 2018, pp. 70-77.

[3] Cypress Semiconductor, “PSoC 6 MCU: PSoC 63 with BLE Datasheet”, PSoC 6 datasheet, May 2018 [Revised Jul. 2018].

[4] Cypress Semiconductor, ” PSoC 63 with BLE Architecture Technical Reference Manual”, PSoC 6 TRM, Aug. 2017 [Revised Apr. 2018].

[5] M. Collotta and G. Pau, "A Novel Energy Management Approach for Smart Homes Using Bluetooth Low Energy," in IEEE Journal on Selected Areas in Communications, vol. 33, no. 12, pp. 2988-2996, Dec. 2015.

[6] N. Imtiaz Jaya and M. F. Hossain, "A Prototype Air Flow Control System for Home Automation Using MQTT Over Websocket in AWS IoT Core," 2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Zhengzhou, China, 2018, pp. 111-1116.

[7] D. Kang et al., "Room Temperature Control and Fire Alarm/Suppression IoT Service Using MQTT on AWS," 2017 International Conference on Platform Technology and Service (PlatCon), Busan, 2017, pp. 1-5.