IoT-Enabled Door Lock System

Article *in* International Journal of Advanced Computer Science and Applications · January 2019

DOI: 10.14569/IJACSA.2019.0100556

5 authors, including:

Trio Adiono
Bandung Institute of Technology
269 PUBLICATIONS **1,444** CITATIONS
SEE PROFILE

Syifaul Fuada
Universitas Pendidikan Indonesia
199 PUBLICATIONS **960** CITATIONS
SEE PROFILE

Sinantya Feranti Anindya
Bandung Institute of Technology
11 PUBLICATIONS **102** CITATIONS
SEE PROFILE

Irfan Gani Purwanda
Bandung Institute of Technology
13 PUBLICATIONS **83** CITATIONS
SEE PROFILE

IoT-Enabled Door Lock System

Trio Adiono[1], Syifaul Fuada[2]
School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Jln. Ganesha No.10, Gd. Ahmad Bakrie (LABTEK VIII) Lt. IV, ZIP 40116, Indonesia[1]
Program Studi Sistem Telekomunikasi, Universitas Pendidikan Indonesia[2]

Sinantya Feranti Anindya[3], Irfan Gani Purwanda[4]
Maulana Yusuf Fathany[5]
University Center of Excellence on Microelectronics, Institut Teknologi Bandung, Jln. Tamansari No.126, IC Design Laboratory, Gd. PAU Lt. IV
ITB Campus, 40132

***Abstract*—This paper covers the design of a prototype for IoT and GPS enabled door lock system. The aim of this research is to design a door lock system that does not need manual input from user for convenience purpose while also remaining secure. The system primarily consists STM32L100 microcontroller as its core, TIP102 transistor that controls 12 V_{DC} solenoid, and Xbee module to communicate with the smart home's host and receive status regarding user's GPS position. The system is tested by measuring the user's distance from the predetermined location using GPS coordinate captured by an Android application, which serves to test whether the system is able to operate as intended and measure the device's power usage. The test result shows that the device is able to work based on GPS coordinate data received, using 42.3 mA and 587 mA current in idle and active modes, respectively.**

***Keywords*—Internet of Things; smart home; smart lock**

I. Introduction

The Internet of Things (IoT) is a concept revolving around global information network that consists 'things' such as smart devices, sensors and actuators, and even smaller networks with their own identities and ability to self-configure and make their own decisions within certain extent, whether individually or collectively [1-2]. The advent of IoT foreshadows the future where everything and everyone is connected to the Internet through all devices they possess, whether from computer, smart phone, and other consumer devices. Objects within IoT network can also communicate using various communication technologies such as WiFi, Bluetooth, near-field communication, and many more.

One of the most common applications of the IoT technology is for supporting a smart home system. Recent smart home systems make good model on how an IoT architecture behaves, as objects in a house are connected to a gateway in wireless manner to communicate whether with each other and the house's inhabitants. As a smart home system is intended to enhance the inhabitants' living quality in terms of comfort of convenience, the IoT is utilized to at least enable easier control and monitoring over the home devices.

A smart home system can be categorized into at least one type based on its functionality: health care, entertainment, energy, and/or security [3]. Among these functionalities, security becomes one of the most crucial factors behind installation of the system. Enhancing home security using smart home can be done in a number of ways, including but not limited to installation of smart, customized door lock. Several examples of smart door lock implementation have already existed, such as camera-based door security systems [4-6], passwords [6-7], smart card [8], and proximity or location detection [9]. Each of the aforementioned methods has its own strengths and weaknesses, such as interoperability value of devices.

In this research, a GPS-based smart door lock is designed. The ultimate aim of this research is to design a door lock system that does not need manual input from user for convenience purpose while also remaining secure. However, for this publication, the scope is limited for examining the feasibility of utilizing GPS for location-based lock to achieve the aforementioned goal. Furthermore, the scope of this paper will be mostly limited to the lock hardware side, while the server and the Android application code handling the GPS tracking will only be briefly explained. This device is part of MINDS smart home system we designed, which also include RGB lamp [10], curtain controller [11], humidity and temperature sensor [12], fan controller [13], and infrared remote [14]. The smart home is controlled using Android-based application detailed in [15]. This paper details the design of the smart lock design and implementation, as well as the testing result.

This paper is divided into following sections: Section I presents the research background. Section II discusses the methodology that cover: highlighted research limitation, specification of developed system, hardware, system workflow, design of protocol, android and server. Section III reports the research result while the research conclusion and future work are given in Section IV and Section V, respectively.

II. System Design

A. Research Limitation

As mentioned above, this work is more focuses on the hardware part including defined diagram block, electronic circuit and its assembly, whereas the software part including system flowchart, android apps, and server are presented briefly, and it will be elaborated in detail for other publications. The test limit on the functional test (locked and unlocked condition) and power consumption measurement using digital multimeter. Further performance metrics will be explored in future.

This work is one part of the big project entitled "Perangkat *Internet-of-Things* untuk Sistem Rumah Cerdas" that was funded by the KEMRISTEK-DIKTI for Desentralisasi scheme (009/SP2H/LT/DRPM/IV/2017).

B. System Specification

As the designed system is part of MINDS smart home system, the overall architecture is similar to other devices designed for the aforementioned smart home. The smart door lock is one of the nodes supported by MINDS system, which is connected to a Raspberry Pi-based host that serves as a 'bridge' for the nodes to receive commands from user device in the form of Android-based smart phone. As for the smart phone, it is connected to the host through either Bluetooth or the Internet (via cloud server). The architecture of the MINDS system is depicted in Fig. 1.

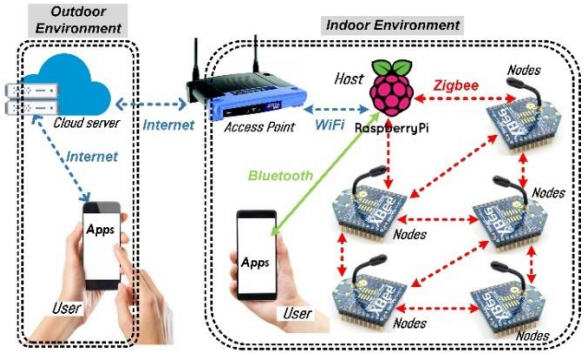


Fig. 1. Architecture of MINDS Smart Home System.

C. Hardware Design

The door lock works like an on/off switch, which is controlled based on user's proximity from the door. The lock system utilizes a battery-powered STM32L100 microcontroller as its core, which is used for controlling 12 V_{DC}/630 mA solenoid and an Xbee module to receive signals from the central host. The solenoid is controlled using TIP 102 transistor by utilizing its cut-off and saturation modes to switch the solenoid's state. Furthermore, the system is equipped with components such as DC power jack, microUSB port for development purpose, switch, and reset and mode buttons. The block diagram of the door lock's hardware is depicted in Fig. 2, while the circuit diagram for the solenoid control is depicted in Fig. 3.

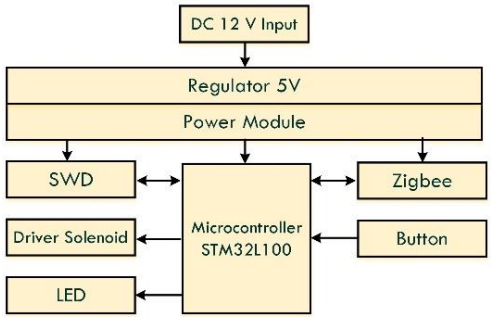


Fig. 2. Block Diagram of the Door Lock System.

As the transistor is required to work in saturation and cut-off region, the value of components such as resistor must be determined in such way so they can support these operation regions. According to TIP 102's datasheet, the base current (IB) for saturation region is 1.9 mA. As such, to achieve the current, the equation (1) is used for calculating the base resistor value.

$$I_B = \frac{Vout - VBE(on)}{RBasis} \quad (1)$$

The value of the V_{out} is based on STM32L100's output port voltage, which is 3.3 V_{DC}, while the voltage between base and emitter ports of the transistor is based on the transistor's Darlington configuration, hence 1.4 V_{DC} (0.7 V_{DC}+0.7 V_{DC}). Based on the calculation, it can be inferred that 1kΩ resistor is required to make the base current into 1.9 mA. Aside from the base current, the maximum power the transistor can handle also needs to be considered, especially when the solenoid is active. According to the datasheet, the transistor is able to handle 80 watt of power dissipation. Assuming the V_{CE} during saturation is 2 V_{DC}, then the solenoid power is:

$$P(transistor) = V_{CE}(sat) * I_C = 2V * 630mA = 1.26Watt$$

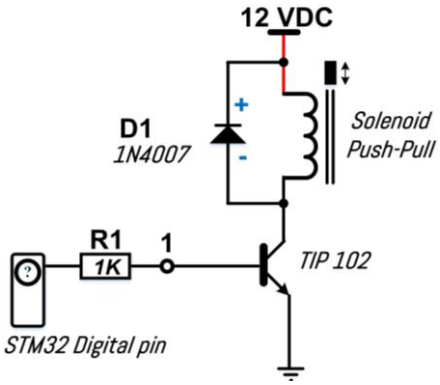


Fig. 3. Circuit Diagram for the Solenoid Control.

D. System Workflow

This section describes two separate programs required to operate the lock: the firmware inside the microcontroller and the mobile phone (Android) software to transmit GPS coordinate of the user to the host. The flowchart for the firmware is depicted in Fig. 4, while the flowchart for the Android application is depicted in Fig. 5.

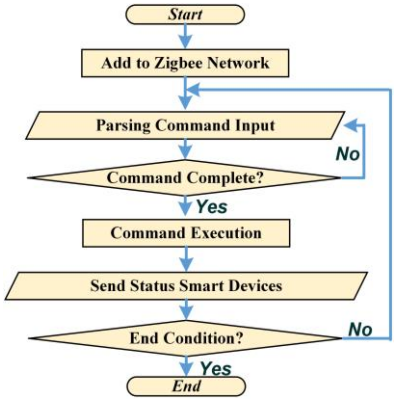


Fig. 4. Flowchart for Firmware for the Door Lock Control.

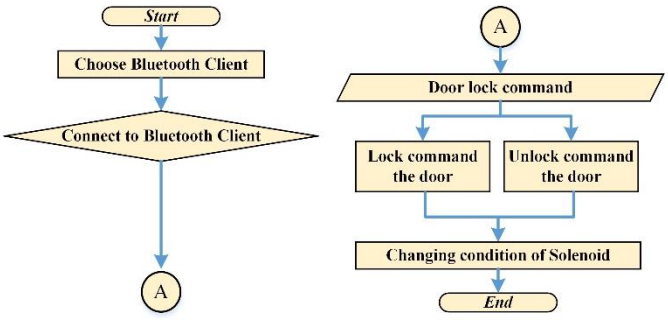


Fig. 5. Flowchart for Android Application for Door Lock Control, Reproduced from [15] under Permission.

Header (3 Bytes)			Address (2 Bytes)		Packet Init (1 Byte)	Data Payload (1 Byte)	Checksum (1 Byte)
1	2	3	Device	Equip			
50	4D	45	07	C1	80	01	
50	4D	45	07	C1	80	00	

Fig. 6. Packet Data Structure for Door Lock Control, Reproduced from [16-17] under Permission.

E. Protocol Design

The message protocol used to transmit GPS data to the host is based on the protocol elaborated in [16]. The specific structure of the message for door lock control is depicted in Fig. 6 (consisting the packet header, address, packet init, Data payload, and checksum).

F. Android Application and Server Design

The door lock works by having the Android application that serves as user interface of the entirety of MINDS system to periodically detect GPS coordinate of the user, then send it to the server (and host) to be compared with the house's coordinate. If the distance is 10 meters or less, the application will send command to both the server to unlock the house, which will be relayed to the host then to the lock hardware. Likewise, if the user's distance with the house is more than 10 meters, then the system will be locked. The system will send a warning if an unauthorized attempts to access the device within the house is made while the house is locked.

III. Results and Discussion

A. System Implementations

Based on the block diagram depicted in Fig. 2 earlier, the printed-circuit board for the controller is designed with the result depicted in Fig. 7. This controller design is similar to the controller from previous project [13]. The reasoning behind the similar design is to add interoperability value to the system while also enhancing its performance.

To control the solenoid lock, the 0th pin of STM32L1000's GPIO-B port is used as the output port. The software implementation for the port configuration is elaborated in Table I, while the control process is elaborated in Table II.

As previously explained, this system works together with the host and MINDS Android application. Fig. 8 displays the comparison between the MINDS application during normal (unlocked) condition and locked condition. The application captures the user's location every 5 minutes using GPS, then send the result to the server (and host), after which the host will send command to the lock device. If the house is supposed to be locked, then a warning will be sent if there is any unauthorized attempt to control the device. The code implementation for the GPS tracking is depicted in Table III, while the code implementation for the control and warning by host is depicted in Tables IV, V and VI.

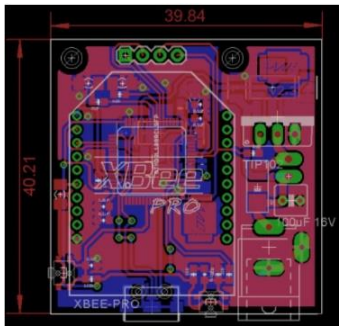


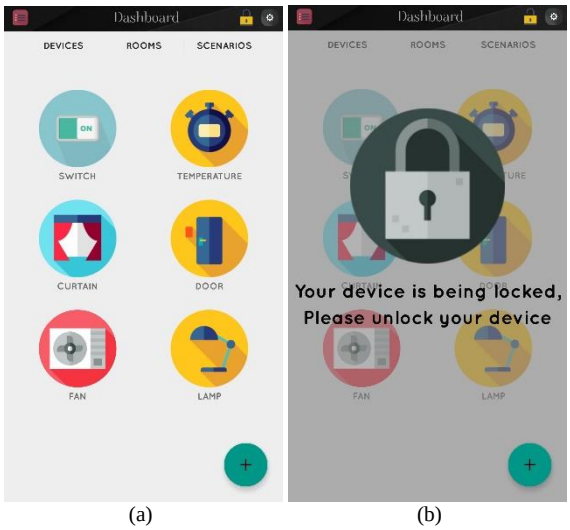
Fig. 7. The Appearance of PCB DC12V Controller for MINDS Door Lock Device at Actual Size (40.21 mm x 39.84 mm).

TABLE I. GPIO-B Port Configuration in STM32L100 Microcontroller

```
// GPIO Init
GPIO_InitTypeDef GPIO_InitStructure;
RCC_AHBPeriphClockCmd( RCC_AHBPeriph_GPIOB, ENABLE); // Enable GPIOB clock
// GPIO Configuration
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_Init(GPIOB, &GPIO_InitStructure);
```

TABLE II. Software Implementation for the Door Lock On/Off Control

```
if(rawlock==0x00) GPIO_LOW(GPIOB,GPIO_Pin_0);
else if(rawlock==0x01) GPIO_HIGH(GPIOB,GPIO_Pin_0);
```



(a) (b)

Fig. 8. The Appearance of MINDS Application During: (a) Normal, Retrieved from [18] under Permission and (b) Locked Conditions.

TABLE III. CODE IMPLEMENTATION FOR GPS TRACKING IN ANDROID APPLICATION

```
if (head.equals("LC")) {
    if (SensorService.latcur != 0.0 && SensorService.lngcur != 0.0) {
        try {                                   // send position
                AMQP.BasicProperties props = new
                AMQP.BasicProperties
                .Builder()
                .correlationId(SensorService.phone_id)
                .build();
                JSONObject obj = new JSONObject();
                obj.put("head", "LC");
                obj.put("homeid", SensorService.home_id);
                obj.put("lat", Double.toString(SensorService
                                                .latcur));
                obj.put("lng", Double.toString(SensorService
                                                .lngcur));
                        StringWriter toSent = new StringWriter();
                        obj.writeJSONString(toSent);
                        String tosendstring = toSent.toString();
                Toast.makeText(getApplicationContext(),Double
                        .toString(SensorSer
                        vice.latcur) + " " +
                        Double.toString(SensorService.
                                lngcur),
                        Toast.LENGTH_SHORT).show();
                SensorService.channel.basicPublish("",
                        SensorService.home_id+"AES",
                        props, SensorService
                        .AESencrypt(tosendstring)
                        .getBytes("UTF-8"));
                SensorService.lngcur = 0.0;
                SensorService.latcur = 0.0;
        } catch (Exception e) {
Toast.makeText(getApplicationContext(), "Failed
        to Update Location",
    Toast.LENGTH_SHORT).show();
        }
    }
}
```

TABLE IV. SOFTWARE IMPLEMENTATION FOR THE RASPBERRY PI HOST LOCATION COMPARISON

```
elif (sent['homeid'] == homeid and sent['head'] == 'LC') :          print (sent['lat'],
sent['lng'])
    cnx = mysql.connector.connect(user='root', password='28031995',
            host='localhost', database='home'+homeid)
    cursor = cnx.cursor()
    cursor.execute("SELECT latitude, longitude FROM info")
    data = cursor.fetchone()
            # Get the user's coordinate in float form
    latf = float(sent['lat'])
    lngf = float(sent['lng'])
            # If the coordinate is (0, 0), assume the user's phone is deactivated
    if (latf == 0.0 and lngf == 0.0):
            cursor.execute("UPDATE users SET location = 'in' WHERE
            phoneid = %s", (props.correlation_id,))
    # Check if user's radius is above 100 meters
    elif (sqrt(math.pow(latf-data[0]),2) + math.pow(lngf-data[1]),2)) >
            0.001) :
            cursor.execute("UPDATE users SET location = 'out'
                    WHERE phoneid = %s", (props.correlation_id,))
            # Check if user is within radius
    cursor.execute("SELECT EXIST (SELECT location FROM users
    WHERE location = 'in')")
    # If not, assume the user is away
    if (not(cursor.fetchone()[0])):
            tosend['homeid'] = homeid
            tosend['head'] = 'SC'
            tosend['name'] = 'lock_scen'
    ch.basic_publish(exchange='', routing_key=sent['homeid']+'AES',
            properties=pika.BasicProperties(correlation_id =
            props.correlation_id),
            body=AESencrypt(json.dumps(tosend),key))
    except:
            print(sys.exc_info()[0])
    channel.basic_consume(callback, queue=queue_name,
    no_ack=True)
    channel.start_consuming()
```

TABLE V. SOFTWARE IMPLEMENTATION FOR RASPBERRY PI HOST DEVICE LOCKING

```
elif (sent['homeid'] == homeid and sent['head'] == 'SC') :
    try :
            print (sent['name'])
            cnx = mysql.connector.connect(user='root',
                    password='28031995', host='localhost',
                    database='home'+homeid)
            cursor = cnx.cursor()

            if (sent['name'] == 'lock_scen') :
                    cursor.execute("UPDATE info SET lockstatus =
                    'lock'")
                    cursor.execute("SELECT homekey FROM info")
                    ch.basic_publish(exchange='amq.topic',
                            routing_key=homeid,
                    body=AESencrypt(json.dumps(sent),
                    cursor.fetchone()[0]))
                    print('set lock')

            if (sent['name'] == 'unlock_scen') :
                    cursor.execute("UPDATE info SET lockstatus =
                    'unlock'")
                    cursor.execute("SELECT homekey FROM info")
                    ch.basic_publish(exchange='amq.topic',
                            routing_key=homeid,
                            body=AESencrypt(json.dumps(sent),
                            cursor.fetchone()[0]))
```

TABLE VI. SOFTWARE IMPLEMENTATION FOR WARNING FROM RASPBERRY PI HOST

```
# COMMAND TO DEVICES      elif (sent['homeid'] == homeid and
sent['head']              ==              'CO')              :
    try:
          #CHECK                  LOCK                  STATUS
          cnx = mysql.connector.connect(user='root',
                        password='28031995',
host='localhost',
    database='home'+homeid)
          cursor                =                cnx.cursor()
          cursor.execute("SELECT   lockstatus   FROM   info")

          #    SEND    WARNING    IF    HOUSE    IS    LOCKED
          if     (cursor.fetchone()[0]     ==     'lock')     :
                  message                 =                {}
                  message['head']           =           'WA'
                  cursor.execute("SELECT homekey FROM info")
                  ch.basic_publish(exchange='amq.topic',
                          routing_key=homeid,
                  body=AESencrypt(json.dumps(message),
                                  cursor.fetchone()[0]))
          else :
                  run_command(sent['address'],  sent['type'],
                                  sent['command'])
                  cursor.execute("SELECT homekey FROM info")
                  ch.basic_publish(exchange='amq.topic',
                          routing_key=homeid,


              BODY=AESENCRYPT(JSON.DUMPS(SENT),
                                  CURSOR.FETCHONE()[0]))
                                  CURSOR.CLOSE()
                                  CNX.CLOSE()
                                  GC.COLLECT()
```

B. *Functional Test*

The testing of the system is conducted on a door miniature (Fig. 9). To test the door, user is required to stand within determined distances with the door. In this case, the threshold is set to 10 meters from the door. According to the result as in Table VII, the door lock system that has been developed can work properly as expected. It can lock and unlock wirelessly.

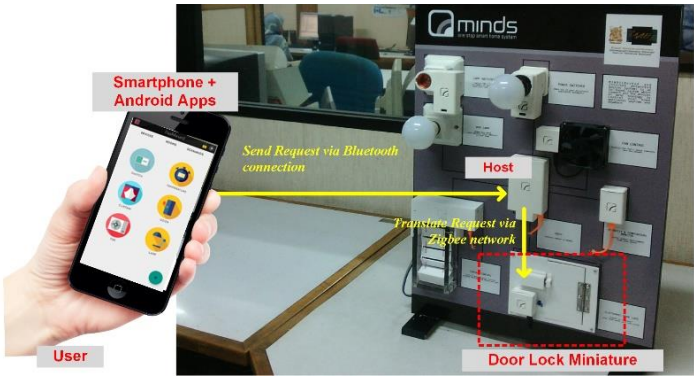


Fig. 9. A Setup for Functional Test of Door Lock.

TABLE VII. POWER MEASUREMENT DOOR LOCK DEVICE

Commands on GUI	Device Condition	Results
Lock	The door is locked	√
Unlock	The door is unlocked	√

C. Power Measurement

To measure the power consumption, simply we used digital multi-meter to know the current flow during two conditions (idle and process). The result of the test is elaborated in Table VIII. The power consumption in idle condition is 507.6 mW that obtained from 12 V_{DC} * 42.3 mA, while in process condition is 7044 mW.

TABLE VIII. POWER MEASUREMENT OF DOOR LOCK DEVICE

Input voltage	Current	
	Idle condition	*Process condition*
12 V_{DC}	42.3 mA	587.5 mA

IV. CONCLUSION

In this paper, a prototype of location-based smart door lock system is designed. The system utilizes the user's GPS coordinate that is captured from a mobile application, which is then sent to a smart home system's central host to enable or disable the door lock based on the user's proximity to the door's designated GPS coordinates. Based on the testing conducted, it can be concluded GPS coordinates can be used for controlling door lock. However, further study is required to improve the quality of the system, whether in terms of power efficiency, area tracking and indoor accuracy, and further increase the security.

V. FUTURE DIRECTION

For the next phase of the work, Android's geo-fencing implementation will be studied to examine its effectiveness on improving the lock's accuracy.

REFERENCES

[1] B. Lorenzo, et al., "A Robust Dynamic Edge Network Architecture for the Internet-of-Things", arXiv preprint arXiv:1710.04861, 2017.

[2] P.P. Ray, "A survey on Internet of Things architectures", J. of King Saud University-Computer and Information Sciences, Vol. 30(3), pp. 291-319, July 2018.

[3] T.D.P. Mendes, et al., "Smart Home Communication Technologies and Applications: Wireless Protocol Assessment for Home Area Network Resources", Energies, Vol. 8(7), pp. 7279-7311, 2015.

[4] R. Manjunatha and R. Nagaraja, "Home Security System and Door Access Control Based on Face Recognition", Int. Research J. of Engineering and Technology (IRJET), Vol. 4(3), pp. 437-442, 2017.

[5] S. Kavde, et al., "Smart Digital Door Lock System using Bluetooth Technology", Int. Conf. on Information, Communication & Embedded Systems (ICICES) 2017.

[6] N.A. Hussein and I. Al Mansoori, "Smart Door System for Home Security using Raspberry Pi 3", Proc. of the Int. Conf. on Computer and Applications, pp. 395-399, 2017.

[7] C. Vongchumyen, et al., "Door Lock System via Web Application", Proc. of the 5th Int. Electrical Engineering Congress, pp. 1-4, March 2017.

[8] Y.C. Yu, "A Practical Digital Door Lock for Smart Home", Proc. of the IEEE Int. Conf. on Consumer Electronics (ICCE), pp. 1-2, 2018.

[9] S. Jensen, "Proximity Door Locking", Master Thesis, Technical University of Denmark.

[10] T. Adiono, M.Y. Fathany, S.F. Anindya, S. Fuada, and I.G. Purwanda, "Wirelessly Control for RGB Lamp End-Device: Design and Implementation," IEEE Region 10 Conf. (TENCON), pp. 2066-2070, October 2018.

[11] T. Adiono, S.F. Anindya, S. Fuada, and M.Y. Fathany, "Curtain Control Systems Development on Mesh Wireless Network of the Smart Home," Bulletin of Electrical Engineering and Informatics (BEEI), Vol. 7, No. 4, pp. 615-625, December 2018.

[12] T. Adiono, M.Y. Fathany, S. Fuada, I.G. Purwanda, and S.F. Anindya, "A Portable Node of Humidity and Temperature Sensor for Indoor Environment Monitoring," Proc. of the 3rd Int. Conf. on Intelligent Green Building and Smart Grid (IGBSG), pp. 1-5.

[13] T. Adiono, M.Y. Fathany, S.F. Anindya, S. Fuada, and I.G. Purwanda, "Development of wireless fan speed control using smartphone for smart home prototype," Unpublished.

[14] T. Adiono, S.F. Anindya, S. Fuada, and M.Y. Fathany, "Developing of General IrDa Remote to Wirelessly Control IR-based Home Appliances," Proc. of the IEEE 7th Global Conf. on Consumer Electronics (GCCE), PP. 461-463, 2018.

[15] T. Adiono, S.F. Anindya, S. Fuada, K. Afifah, and I.G. Purwanda, "Efficient Android Software Development using MIT App Inventor 2 for Bluetooth-based Smart Home," Wireless Pers Commun, Vol. 105(1), pp. 233-256, March 2019.

[16] T. Adiono, B. Tandiawan, and S. Fuada, "Device Protocol Design for Security on Internet of Things based Smart Home," Int. J. of Online Engineering (i-JOE), Vol. 14(7), pp. 161-170, 2018. M.Y. Fathany and T. Adiono, "wireless protocol design for smart home on mesh wireless sensor network," Proc. of int. symp. On intelligent signal processing and communication systems, pp. 42-477, November 2015.

[17] T. Adiono, et al., "Design of smart home mobile application with high security and automatic features," Proc. of the 2018 3rd Int. Conf. on Intelligent Green Building and Smart Grid (IGBSG), pp. 1–4, 2018.

[18] T. Adiono, "Intelligent and secured software application for IoT based smart home," Proc. of the 2017 IEEE 6th Global Conf. on Consumer Electronics (GCCE), pp. 1-2, 2018.