

家管理システムの作成

坪田 大和 原田 僚人

1. 研究概要

最近身近になっている「IoT」の技術。
例えば、スマートフォンから遠隔でエアコンをつけておいたり、スマートフォンを使った玄関の鍵の施錠だったりが挙げられる。そんな IoT の技術を学び、IoT を知らない人でも視覚的に理解できるモデルを作成する。
Arduino UNO R4 Wi-Fi という基盤を主として機器の制御を行い、新たなプログラムの知識・技術を得ると同時にマイコンの制御について理解を深めた。

2. 研究の具体的内容

(0) 使用機器

以下の表は、実際に課題研究で使用した機器やアプリケーションの名称と用途についてまとめたものである。(表1)

表1 名称/用途

Arduino UNO R4 Wi-Fi	全体の制御
Arduino UNO Rev 3	(図1)
Seeed Studio XIAO ESP32S3 Sense	家監視カメラ
	(図2)
木材	家
サーボモータ	遠隔ロック カメラの視点移動
7セグメントLED4桁	デジタル時計
電子ブザー	アラーム機能
電球	遠隔操作の電球
人感センサ	防犯センサ
スマートフォン	操作作用
Arduino Cloud	プログラムの作成
Arduino IoT Cloud Remote	スマートフォンの 操作画面



図1 Arduino UNO R4 Wi-Fi

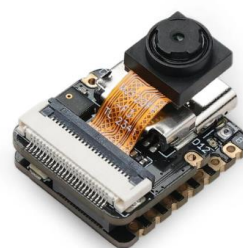


図2 Seeed Studio XIAO ESP32S3 Sense

(1) 制作手順

0：環境整備

スマートフォンから Wi-Fi で Arduino 基盤に接続できるようにするため必要があるため IP アドレスと MAC アドレスを調べ、マイコンをネットに接続できるようにした。これによりスマートフォンと Arduino の基盤とが接続できるようになった。またピンの数が不足するため、全体制御用マイコン(R4 Wi-Fi)と時計用マイコン(Rev3)に分けた。

1：プログラムの作成

Arduino 環境で開発をしていくことにした。ソフトウェアとしては Arduino Cloud を用いて開発を進めた。こちらでは、スマートフォンでの操作画面の作成に加え、実際にプログラム開発も行うことができたため操作画面とプログラムを並行して進めた。

2：家の作成

メインとなる材料には木材を使用した。安全面や外見の綺麗さに配慮して紙やすりや棒やすりを用いて整えた。また、中の様子や状態が見やすいようにL字型の設計にした。

3：機能の搭載

基盤を隠す・配線をきれいにするために家の下に土台を用意し、その中に配線した基盤やブレッドボードなどを入れた。この土台により家全体の移動もしやすいものとなった。電球の取り付けやロックの取り付けなどにはそれぞれの機器のサイズに合わせて木材を加工した。

4：追加機能

文化祭までに実装させた機能に特に問題もなかったため家監視カメラを追加機能として制作した。

(2) 作成したシステム

私たちは、文化祭までに4つのシステムを、その後カメラ機能をもつシステムを作成した。計5つのシステムを搭載することができた。

以下の表は、作成したシステムの概要についてまとめたものである。(表2)

表2 作成したシステム名とその概要

遠隔ロックシステム	スマートフォンからの入力によって玄関の施錠・解除ができる 仕組みとしては、玄関の扉の真下にくぼみを掘り、サーボモータによって突起が引かかるか抜けるかによって施錠・解除が可能である
-----------	--

アラーム機能付きデジタル時計	・7セグメントディスプレイを用いて、24時間を表すことのできる時計 ・スマートフォン側から任意の時間(時間単位であり分単位ではない)にアラームをセットができる。セットした時間が来たとき、電子音が鳴る
遠隔消・点灯可能な電球	スマートフォンからの入力によって電球の消灯・点灯を切り替えることができる
防犯センサ	人感センサを用いて、人を感知すればスマートフォンに通知(ライトの点灯により示す)が届くようになっている
家監視カメラ	ESP32 CAMを用いて、家の中を監視し映像をスマートフォン側から取得ができるカメラ機能をもつ(ビデオストリーミング)

1：遠隔ロックシステム

スマートフォン側からボタンを押すとサーボモータが動作するようにプログラムを作成した。土台の裏側にサーボモータを取り付け、扉の下側・土台の扉の位置にそれぞれ穴、くぼみをつけボタンを押すとサーボモータの突起が引かかるか戻るかにより動作するようにした。(図3)

プログラムとしては以下のようなになる。

```
if(key == HIGH){  
    myservo.write(0);  
}  
else{  
    myservo.write(90);  
}
```

*key はスマートフォンからの入力した値を受け取るための変数(今回は HIGH, LOW)

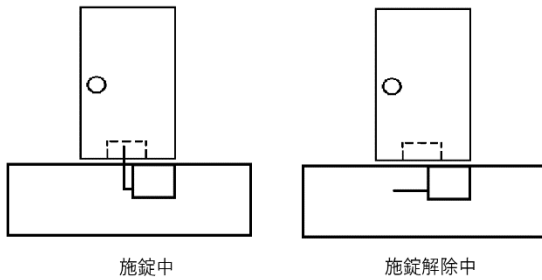


図3 サーボモータの動作

2 : アラーム機能付きデジタル時計

以下は、アラーム機能の on・off を表すものである。

```
if(timer==HIGH){
    digitalWrite(13,HIGH);
}else{
    digitalWrite(13,LOW);
}
```

*timer はスマートフォンからの入力した値を受け取るための変数（今回は HIGH, LOW）

そして以下は、実際にどのようにして時刻を表したり、アラーム機能を実現したりしたかを説明したものである。

初めに図4では、時刻計算を行っている。60秒経つと1分に、60分経つと1時間に、24時間経つともう一度0時に戻るといった計算を行っている。

```
void loop(){
    delay(1);
    times++;
    if(times==915){times2++; times=0;}
    if(times2==60){count1++; times2=0;}
    if(count1==60){count2++; count1=0;}
    if(count2==24) count2=0;

    i=count1/10; n=i*10; keta[0]=count1-n; keta[1]=i;
    i=count2/10; n=i*10; keta[2]=count2-n; keta[3]=i;

    count3=times%4;
    i=keta[count3];
    hyouzi2(i);
    hyouzi1(count3);
}
```

図4 時刻計算を表すプログラム

続いて図5では、時計制御用とは別の全体

制御用のマイコンからアラームの時間が二進数で送られているため、そのデータを十進数に戻すプログラムである。最後の行では各桁を足していることを表す。

```
if(digitalRead(14)==HIGH) s[0]=16; else s[0]=0;
if(digitalRead(15)==HIGH) s[1]=8; else s[1]=0;
if(digitalRead(16)==HIGH) s[2]=4; else s[2]=0;
if(digitalRead(17)==HIGH) s[3]=2; else s[3]=0;
if(digitalRead(18)==HIGH) s[4]=1; else s[4]=0;
```

```
num2=s[0]+s[1]+s[2]+s[3]+s[4];
```

図5 二進数から十進数に変換するプログラム

そして図6では時計制御用マイコンのプログラム全体になっている。

```
int keta[4]={0,0,0,0},count1=0,count2=0,count3=0,times=0,times2,i,b,n,num,num2,s[5];
char seg[10][7]={{1,1,1,1,1,0},{0,1,1,0,0,0},
{1,1,0,1,1,0},{1,1,1,1,0,0,1},
{0,1,1,0,0,1},{1,0,1,1,0,1,1},
{1,0,1,1,1,1,1},{1,1,1,0,0,0,0},
{1,1,1,1,1,1,1},{1,1,1,1,0,1,1}};

void setup(){
    Serial.begin(9600);
    for(i=2;i<=13;i++) pinMode(i,OUTPUT);
    for(i=14;i<=19;i++) pinMode(i,INPUT);
}

void loop(){
    delay(1);
    times++;
    if(times==915){times2++; times=0;}
    if(times2==60){count1++; times2=0;}
    if(count1==60){count2++; count1=0;}
    if(count2==24) count2=0;

    i=count1/10; n=i*10; keta[0]=count1-n; keta[1]=i;
    i=count2/10; n=i*10; keta[2]=count2-n; keta[3]=i;

    count3=times%4;
    i=keta[count3];
    hyouzi2(i);
    hyouzi1(count3);

    if(digitalRead(14)==HIGH) s[0]=16; else s[0]=0;
    if(digitalRead(15)==HIGH) s[1]=8; else s[1]=0;
    if(digitalRead(16)==HIGH) s[2]=4; else s[2]=0;
    if(digitalRead(17)==HIGH) s[3]=2; else s[3]=0;
    if(digitalRead(18)==HIGH) s[4]=1; else s[4]=0;

    num2=s[0]+s[1]+s[2]+s[3]+s[4];

    if(digitalRead(19)==HIGH){
        if(count2==num2 && count1==0){
            if(times<=500) digitalWrite(13,HIGH); else digitalWrite(13,LOW);
        }
        if(digitalRead(19)==LOW) digitalWrite(13,LOW);
    }

    void hyouzi1(int b){
        if(b==0){digitalWrite(2,LOW); digitalWrite(3,LOW); digitalWrite(4,LOW); digitalWrite(5,HIGH);}
        if(b==1){digitalWrite(2,LOW); digitalWrite(3,LOW); digitalWrite(4,HIGH); digitalWrite(5,LOW);}
        if(b==2){digitalWrite(2,LOW); digitalWrite(3,HIGH); digitalWrite(4,LOW); digitalWrite(5,LOW);}
        if(b==3){digitalWrite(2,HIGH); digitalWrite(3,LOW); digitalWrite(4,LOW); digitalWrite(5,LOW);}
    }

    void hyouzi2(int b){
        for(int i=0;i<7;i++) digitalWrite(6+i,!(seg[b][i]));
    }
}
```

図6 完成した時計制御用マイコンのプログラム

3：遠隔消・点灯可能な電球

以下がプログラムとなっている。

```
if(light == HIGH){
    digitalWrite(4,HIGH);
}else{
    digitalWrite(4,LOW);
}
```

light というスマートフォン側の変数とマイコン側のピン4がどちらも HIGH であれば点灯、どちらも LOW であれば消灯と操作画面のオブジェクトの反応によって消・点灯が切り替えることができるようになっている。

4：防犯センサ

人感センサが反応したときは、図7のように表示される。反応していないときには図8のように表示される。これにより外出時に人の有無をスマートフォンから確認できる。

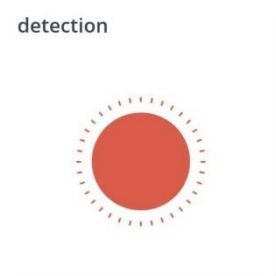


図7 スマートフォン側の表示（人検知時）



図8 スマートフォン側の表示（未検知時）

5：家監視カメラ

図9では、実際に映像が映ったときの画面を載せたものである。

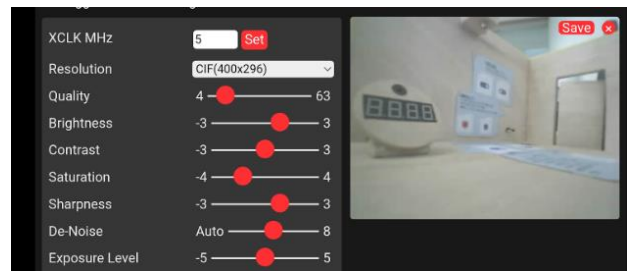


図9 実際の動作画面

図10は、家監視カメラでのプログラム全体である。

```
#include "esp_camera.h"
#include <WiFi.h>
#define CAMERA_MODEL_XIAO_ESP32S3
#include "camera_pins.h"

const char *ssid = "A303ZT";
const char *password = "bcc9c4d2c7e8";

void startCameraServer();
void setupLedFlash(int pin);

void setup() {
    Serial.begin(115200);
    Serial.setDebugOutput(true);
    Serial.println();
    camera_config_t config;
    config.ledc_channel = LEDC_CHANNEL_0;
    config.ledc_timer = LEDC_TIMER_0;
    config.pin_d0 = Y2_GPIO_NUM;
    config.pin_d1 = Y3_GPIO_NUM;
    config.pin_d2 = Y4_GPIO_NUM;
    config.pin_d3 = Y5_GPIO_NUM;
    config.pin_d4 = Y6_GPIO_NUM;
    config.pin_d5 = Y7_GPIO_NUM;
    config.pin_d6 = Y8_GPIO_NUM;
    config.pin_d7 = Y9_GPIO_NUM;
    config.pin_xclk = XCLK_GPIO_NUM;
    config.pin_pclk = PCLK_GPIO_NUM;
    config.pin_vsync = VSYNC_GPIO_NUM;
    config.pin_href = HREF_GPIO_NUM;
    config.pin_sccb_sda = SIOD_GPIO_NUM;
    config.pin_sccb_scl = SIOC_GPIO_NUM;
    config.pin_pwdn = PWDN_GPIO_NUM;
    config.pin_reset = RESET_GPIO_NUM;
    config.xclk_freq_hz = 20000000;
    config.frame_size = FRAMESIZE_UXGA;
    config.pixel_format = PIXFORMAT_JPEG;
    config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
    config.fb_location = CAMERA_FB_IN_PSRAM;
    config.jpeg_quality = 12;
    config.fb_count = 1;
    if (config.pixel_format == PIXFORMAT_JPEG) {
        if (psramFound()) {
            config.jpeg_quality = 10;
            config.fb_count = 2;
            config.grab_mode = CAMERA_GRAB_LATEST;
        } else {
            config.frame_size = FRAMESIZE_SVGA;
            config.fb_location = CAMERA_FB_IN_DRAM;
        }
    } else {

```

図10 家監視カメラのメインプログラム

```

    config.frame_size = FRAMESIZE_240X240;
    #if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
    #endif
    }
    #if defined(CAMERA_MODEL_ESP_EYE)
    pinMode(13, INPUT_PULLUP);
    pinMode(14, INPUT_PULLUP);
    #endif
    esp_err_t err = esp_camera_init(&config);
    if (err != ESP_OK) {
        Serial.printf("Camera init failed with error 0x%x", err);
        return;
    }
    sensor_t *s = esp_camera_sensor_get();
    if (s->id.PID == OV3660_PID) {
        s->set_vflip(s, 1);
        s->set_brightness(s, 1);
        s->set_saturation(s, -2);
    }
    if (config.pixel_format == PIXFORMAT_JPEG) {
        s->set_framesize(s, FRAMESIZE_QVGA);
    }
    #if defined(CAMERA_MODEL_M5STACK_WIDE) ||
    defined(CAMERA_MODEL_M5STACK_ESP32CAM)
    s->set_vflip(s, 1);
    s->set_hmirror(s, 1);
    #endif
    #if defined(CAMERA_MODEL_ESP32S3_EYE)
    s->set_vflip(s, 1);
    #endif
    #if defined(LED_GPIO_NUM)
    setupLedFlash(LED_GPIO_NUM);
    #endif
    WiFi.begin(ssid, password);
    WiFi.setSleep(false);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    startCameraServer();
    Serial.print("Camera Ready! Use 'http://'");
    Serial.print(WiFi.localIP());
    Serial.println(" to connect");
}

void loop() {
    delay(10000);
}

```

図 10 家監視カメラのメインプログラム

6 : プログラムの全容

以下は、今回のプログラムの全体をまとめたものである。スマートフォンと各マイコンとを接続するためのものである。(図 11)

```

#include "thingProperties.h"
#include <Servo.h>

Servo myservo;
int count=0, save, s;

void setup(){
    Serial.begin(9600);
    delay(1500);
    initProperties();
    ArduinoCloud.begin(ArduinoIoTPreferredConnection);
    setDebugMessageLevel(2);
    ArduinoCloud.printDebugInfo();
    myservo.attach(2);
    pinMode(3,INPUT); pinMode(4,OUTPUT); pinMode(5,OUTPUT);
    pinMode(6,OUTPUT); pinMode(7,INPUT);
    for(int i=8;i<=13;i++) pinMode(i,OUTPUT);
}

void loop(){
    ArduinoCloud.update();
    delay(1);
    save=times;

    if(save>=16){save-=16; digitalWrite(8,HIGH);} else digitalWrite(8,LOW);
    if(save>=8){save-=8; digitalWrite(9,HIGH); s=1;}
    else {digitalWrite(9,LOW); s=0;}
    if(save>=4){save-=4; digitalWrite(10,HIGH);} else digitalWrite(10,LOW);
    if(save>=2){save-=2; digitalWrite(11,HIGH);} else digitalWrite(11,LOW);
    if(save>=1){save-=1; digitalWrite(12,HIGH);} else digitalWrite(12,LOW);
    if(timer==HIGH) digitalWrite(13,HIGH); else digitalWrite(13,LOW);
    if(light==HIGH) digitalWrite(4,HIGH); else digitalWrite(4,LOW);
    if(digitalRead(7)==LOW) kenti=true; else kenti=false;
    if(key==HIGH) myservo.write(0); else myservo.write(90);
    if(digitalRead(3)==HIGH) digitalWrite(6,HIGH);
    else digitalWrite(6,LOW);}

```

図 11 完成した全体制御用マイコンのプログラム

スマートフォン側のオブジェクト（ボタンなど）が返す値（HIGH か LOW など）によってマイコン側の指定したピンと反応しシステムが稼働するようになっている。

（2）家の途中経過から完成まで

初めに大きさを決め、木材を加工しおおよその外枠を決めた。(図 12)

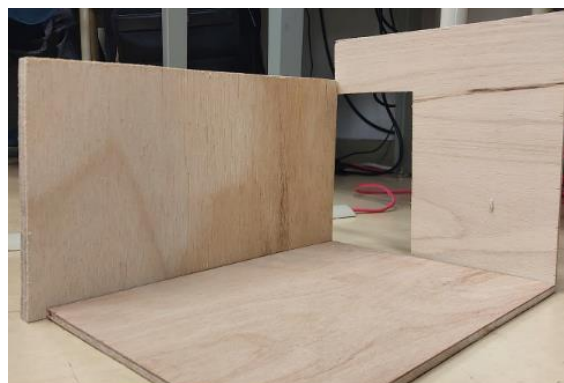


図 12 家の外枠（試作段階）

この段階では薄い木材を使用していたため、厚い木材を使用して強度のある外枠を作成した。そこに各機能を順に搭載した。

まず初めに、アラーム機能付きデジタル時計を搭載した。(図 13)



図 13 アラーム機能付きデジタル時計

次に、防犯センサを搭載した。(図 14)



図 14 防犯センサ

そして、遠隔消・点灯可能な電球、遠隔ロックシステムを搭載した。

(図 15・16)



図 15 遠隔で消・点灯可能な電球



図 16 遠隔ロックシステム

最後に、家監視カメラを搭載した。
*図 9 にて紹介しているため写真は省略している

以下の図は、全体の完成図を表したものである。(図 17)



図 17 完成図（正面）

実際に配線した基盤などは土台の中に隠し、
整っている状態を表した。(図 18)

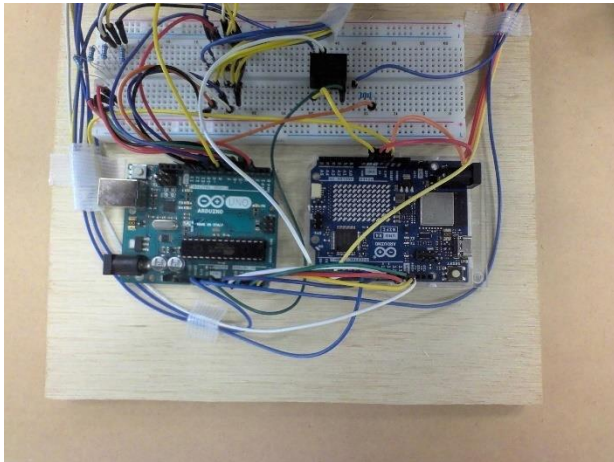


図 18 配線した基盤

以下は、スマートフォン側の操作画面である。(図 19)

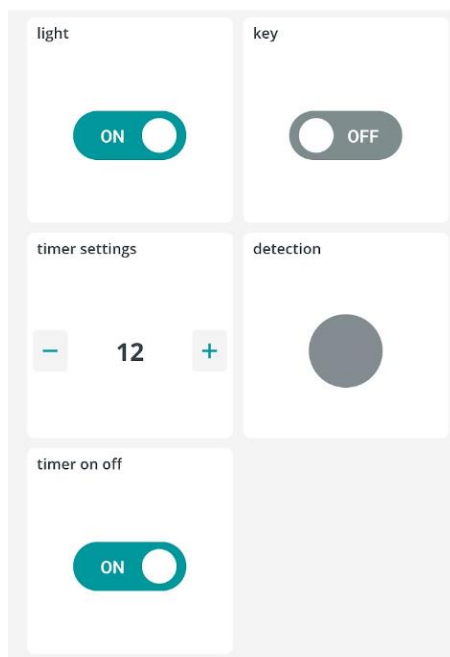


図 19 スマートフォンでの操作画面

各ボタンなどの説明については以下に示す。

light : on のときに電球が点灯し、off のときに電球が消灯する

key : on のときにロックがかかり、off のときにロックが解除される

timer settings : アラームの時間設定

detection: 人感センサに反応があれば点灯
timer on off : タイマーをつけるかどうかを
on、off によって切り替える。

(3) 作業表・役割分担表

以下の表は、月別の作業内容である。(表 3)

表 3 作業表

3・4月	予定表の作成 研究概要の確定 必要品の購入
5月	試作 簡易プログラム作成
6月	企画発表 (資料作成) プログラム作成 (デジタル時計) 家の作成
7・8月	プログラム作成
9月	家の作成 プログラム作成 (電球・防犯センサ)
10月	家に機能を搭載 プログラム作成 (電球・防犯センサ)
11月	プログラム作成 (遠隔ロック システム) 文化祭 (ポスター作成)
12・1月	追加機能の作成 (家監視カメラ) 報告書作成 研究発表 (資料作成)

そして以下の表は役割分担表である。(表 4)

Arduino Cloud の基本的な使い方

https://cediablog.com/arduino-cloud-switch/#index_id5

表 4 役割分担表

坪田大和	プログラムの作成 発表用資料の作成
原田僚人	家の作成 日報・報告書・発表 用資料の作成

3. 研究のまとめ

本研究では、IoT (Internet of Things) の技術を活用し、スマートフォンから設備を操作できるスマートホームの仕組みについて研究・制作を行った。具体的には、スマートフォンからの操作によって家庭内の電気の消灯・点灯を行うシステム、および遠隔で施錠・解錠が可能なロックシステムをプログラムによって構築した。制作を通して、私たちはインターネットを介して機器同士が接続されることで、離れた場所からでもリアルタイムに制御が可能になることを確認できた。また、IoT 技術が生活の利便性や安全性を高める可能性を持っていることを、実際の動作を通して理解することができた。一方で、通信エラーやセキュリティ対策の重要性など、実用化に向けた課題も明らかになった。本研究を通じて、IoT は日常生活をより便利にするだけでなく、今後の社会において重要な技術であると考えられる。今後は、セキュリティ機能の強化や操作の自動化などを取り入れることで、より実用的で安全なシステムへ発展させていきたい。

参考文献

ESP32CAM のビデオストリーミング

https://wiki.seeedstudio.com/ja/xiao_esp32s3_camera_usage/#project-ii-video-streaming