#include "esp\_camera.h"

#include "FS.h"

#include "SPI.h"

#include "SD.h"

#include "EEPROM.h"

#include "driver/rtc\_io.h"

#include "ESP32\_MailClient.h"

// Select camera model

//#define CAMERA\_MODEL\_WROVER\_KIT

//#define CAMERA\_MODEL\_ESP\_EYE

//#define CAMERA\_MODEL\_M5STACK\_PSRAM

//#define CAMERA\_MODEL\_M5STACK\_WIDE

#define CAMERA\_MODEL\_AI\_THINKER

#include "camera\_pins.h"

#define ID\_ADDRESS 0x00

#define COUNT\_ADDRESS 0x01

#define ID\_BYTE 0xAA

#define EEPROM\_SIZE 0x0F

uint16\_t nextImageNumber = 0;

#define WIFI\_SSID "vignanwifi"

#define WIFI\_PASSWORD "123456789"

#define emailSenderAccount "hmiinternship123@gmail.com" //To use send Email for Gmail to port 465 (SSL), less secure app option should be enabled. https://myaccount.google.com/lesssecureapps?pli=1

#define emailSenderPassword "dinesh123@"

#define emailRecipient "hmiinternship123@gmail.com"

//The Email Sending data object contains config and data to send

SMTPData smtpData;

//Callback function to get the Email sending status

void sendCallback(SendStatus info);

void setup()

{

Serial.begin(115200);

Serial.println();

Serial.println("Booting...");

//connect to WiFi network

Serial.print("Connecting to AP");

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED)

{

Serial.print(".");

delay(200);

}

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

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\_sscb\_sda = SIOD\_GPIO\_NUM;

config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;

config.pin\_pwdn = PWDN\_GPIO\_NUM;

config.pin\_reset = RESET\_GPIO\_NUM;

config.xclk\_freq\_hz = 20000000;

config.pixel\_format = PIXFORMAT\_JPEG;

//init with high specs to pre-allocate larger buffers

if(psramFound())

{

config.frame\_size = FRAMESIZE\_UXGA;

config.jpeg\_quality = 10;

config.fb\_count = 2;

} else

{

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

#if defined(CAMERA\_MODEL\_ESP\_EYE)

pinMode(13, INPUT\_PULLUP);

pinMode(14, INPUT\_PULLUP);

#endif

//initialize camera

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK)

{

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

//set the camera parameters

sensor\_t \* s = esp\_camera\_sensor\_get();

s->set\_contrast(s, 2); //min=-2, max=2

s->set\_brightness(s, 2); //min=-2, max=2

s->set\_saturation(s, 2); //min=-2, max=2

delay(100); //wait a little for settings to take effect

//mount SD card

Serial.println("Mounting SD Card...");

MailClient.sdBegin(14,2,15,13);

if(!SD.begin())

{

Serial.println("Card Mount Failed");

return;

}

//initialize EEPROM & get file number

if (!EEPROM.begin(EEPROM\_SIZE))

{

Serial.println("Failed to initialise EEPROM");

Serial.println("Exiting now");

while(1); //wait here as something is not right

}

if(EEPROM.read(ID\_ADDRESS) != ID\_BYTE) //there will not be a valid picture number

{

Serial.println("Initializing ID byte & restarting picture count");

nextImageNumber = 0;

EEPROM.write(ID\_ADDRESS, ID\_BYTE);

EEPROM.commit();

}

else //obtain next picture number

{

EEPROM.get(COUNT\_ADDRESS, nextImageNumber);

nextImageNumber += 1;

Serial.print("Next image number:");

Serial.println(nextImageNumber);

}

//take new image

camera\_fb\_t \* fb = NULL;

//obtain camera frame buffer

fb = esp\_camera\_fb\_get();

if (!fb)

{

Serial.println("Camera capture failed");

Serial.println("Exiting now");

while(1); //wait here as something is not right

}

//save to SD card

//generate file path

String path = "/IMG" + String(nextImageNumber) + ".jpg";

fs::FS &fs = SD;

//create new file

File file = fs.open(path.c\_str(), FILE\_WRITE);

if(!file)

{

Serial.println("Failed to create file");

Serial.println("Exiting now");

while(1); //wait here as something is not right

}

else

{

file.write(fb->buf, fb->len);

EEPROM.put(COUNT\_ADDRESS, nextImageNumber);

EEPROM.commit();

}

file.close();

//return camera frame buffer

esp\_camera\_fb\_return(fb);

Serial.printf("Image saved: %s\n", path.c\_str());

//send email

Serial.println("Sending email...");

//Set the Email host, port, account and password

smtpData.setLogin("smtp.gmail.com", 587, emailSenderAccount, emailSenderPassword);

//Set the sender name and Email

smtpData.setSender("SMART CROP PROTECTION", emailSenderAccount);

//Set Email priority or importance High, Normal, Low or 1 to 5 (1 is highest)

smtpData.setPriority("Normal");

//Set the subject

smtpData.setSubject("ANIMALS FOUND IN YOUR FARM");

//Set the message - normal text or html format

smtpData.setMessage("<div style=\"color:#003366;font-size:20px;\">Image captured and attached.</div>", true);

//Add recipients, can add more than one recipient

smtpData.addRecipient(emailRecipient);

//smtpData.addRecipient(emailRecipient2);

//Add attach files from SD card

smtpData.addAttachFile(path);

//Set the storage types to read the attach files (SD is default)

smtpData.setFileStorageType(MailClientStorageType::SD);

smtpData.setSendCallback(sendCallback);

//Start sending Email, can be set callback function to track the status

if (!MailClient.sendMail(smtpData))

Serial.println("Error sending Email, " + MailClient.smtpErrorReason());

//Clear all data from Email object to free memory

smtpData.empty();

pinMode(4, OUTPUT); //GPIO for LED flash

digitalWrite(4, LOW); //turn OFF flash LED

rtc\_gpio\_hold\_en(GPIO\_NUM\_4); //make sure flash is held LOW in sleep

Serial.println("Entering deep sleep mode");

Serial.flush();

esp\_sleep\_enable\_ext0\_wakeup(GPIO\_NUM\_13, 0); //wake up when pin 13 goes LOW

delay(10000); //wait for 10 seconds to let PIR sensor settle

esp\_deep\_sleep\_start();

}

void loop()

{

}

//Callback function to get the Email sending status

void sendCallback(SendStatus msg)

{

//Print the current status

Serial.println(msg.info());

//Do something when complete

if (msg.success())

{

Serial.println("----------------");

}

}