

«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

ИНСТИТУТ КОМПЬЮТЕРНЫХ НАУК И ТЕХНОЛОГИЙ  
Высшая школа программной инженерии

Курсовая работа  
по дисциплине «Микропроцессорные системы»

Выполнили  
студенты гр. 3530904/80101

Пылаев Я. С.  
Смирнов П. А.  
Трофимов Ф. О.

Руководитель

Круглов С. К.

Санкт-Петербург  
2020

# Содержание

<b>1 Задание</b>	<b>2</b>
<b>2 Общее описание проектируемой системы</b>	<b>2</b>
2.1 Сценарий использования . . . . .	2
2.2 Требования . . . . .	2
<b>3 Аппаратная часть</b>	<b>3</b>
3.1 Микрокомпьютер Raspberry Pi 3 Model B . . . . .	3
3.2 Подключение периферийных устройств . . . . .	3
3.3 Raspberry Pi Camera Module Rev 1.3 . . . . .	4
3.3.1 Описание камеры . . . . .	4
3.3.2 Принципиальная схема . . . . .	5
3.3.3 Технические характеристики . . . . .	5
3.3.4 Программные возможности . . . . .	6
3.4 Макетная плата MB102 . . . . .	6
3.4.1 Технические характеристики . . . . .	6
3.4.2 Принципиальная схема . . . . .	7
<b>4 Интерфейсы</b>	<b>7</b>
4.1 GPIO . . . . .	7
4.1.1 Учет падения напряжения . . . . .	8
4.1.2 PULL-UP / PULL-DOWN резисторы . . . . .	8
4.2 CSI . . . . .	9
4.2.1 ПО для работы с камерой . . . . .	11
<b>5 Удаленное подключение к Raspberry Pi</b>	<b>11</b>
5.1 Настройка сервера . . . . .	12
5.2 Настройка клиента . . . . .	13
<b>6 Программная реализация: исходный код программы на GitHub</b>	<b>14</b>
6.1 Описание схемы реализации . . . . .	15
6.1.1 Временная диаграмма активности компонентов системы . . . . .	16
6.1.2 Функционал бота . . . . .	16
<b>7 Демонстрация работы системы</b>	<b>16</b>
<b>8 Вывод</b>	<b>18</b>

## 1 Задание

Реализовать бытовую систему или устройство на основе процессора семейства ARM. Особенно важно сдеать исследование стуртуры и возможностей аппаратуры.

## 2 Общее описание проектируемой системы

**My-HomeCam-Security** - система контроля входа в умном доме.

### 2.1 Сценарий использования

Когда гость подходит к входной двери и нажимает на ней кнопку звонка (1), срабатывает распознавание лица (2) и фотография человека вместе с информацией о нем отправляется прямиком на мобильное устройство хозяина дома (3). Далее хозяин решает званный или нет гость стоит у него за дверью, нажимая соответствующую кнопку в приложении (4), и дверь в зависимости от этого решения открывается либо остается закрытой (5). Также владелец в любой момент может включить прямую трансляцию с камеры системы, тем самым наблюдая за входом в реальном времени. Общая схема работы системы представлена на *Figure 1*.

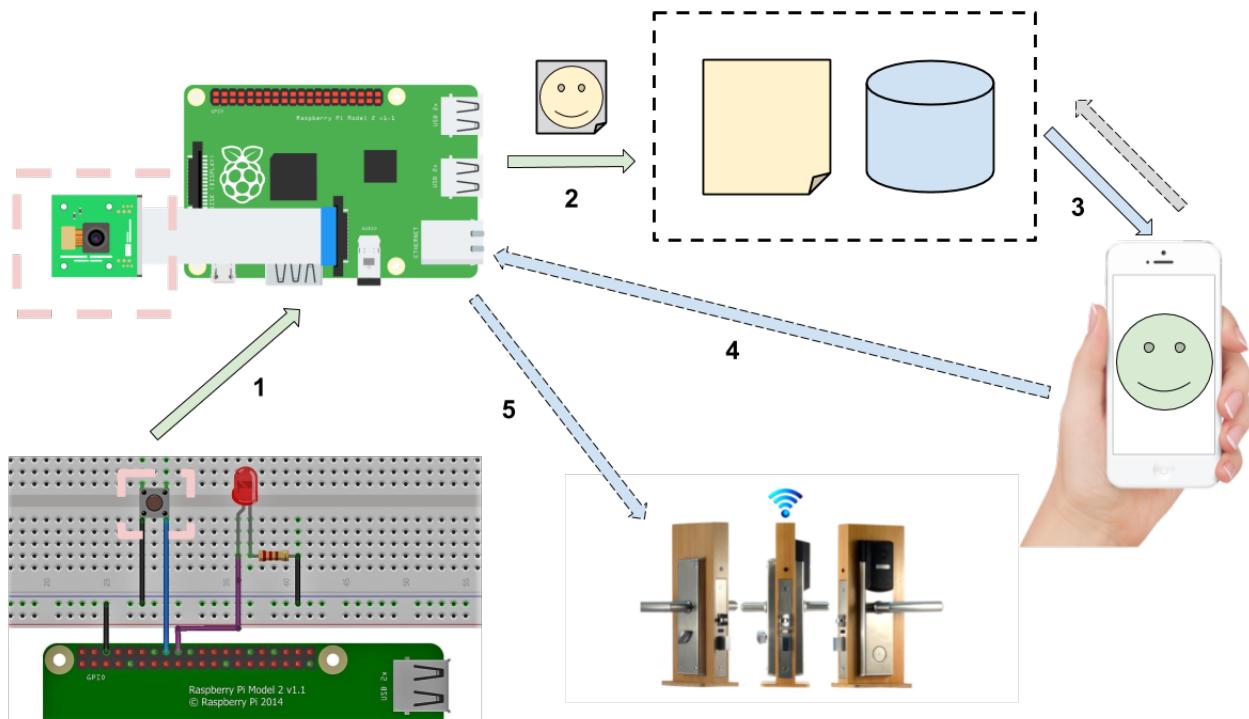


Figure 1: Общая схема работы

### 2.2 Требования

Для описанного сценария была выбрана следующая реализация:

- Основыиным аппаратным модулем является одноплатыный компьютер Raspberry Pi.
- Кнопка, световой индикатор и камера - непосредственно подключаются к "малинке".

- В качестве приложения для управления системой выбран формат бота Telegram.
- Язык программной реализации - Python.

### 3 Аппаратная часть

#### 3.1 Микрокомпьютер Raspberry Pi 3 Model B

На устройстве установлен 64-х битный четырехъядерный процессор **ARM Cortex-A53** с тактовой частотой 1,2 ГГц на ядро в составе однокристальной платформы Broadcom BCM2837. Данный чип обеспечивает прирост производительности на 50–60%. Подробные характеристики представлены на рисунке ниже.

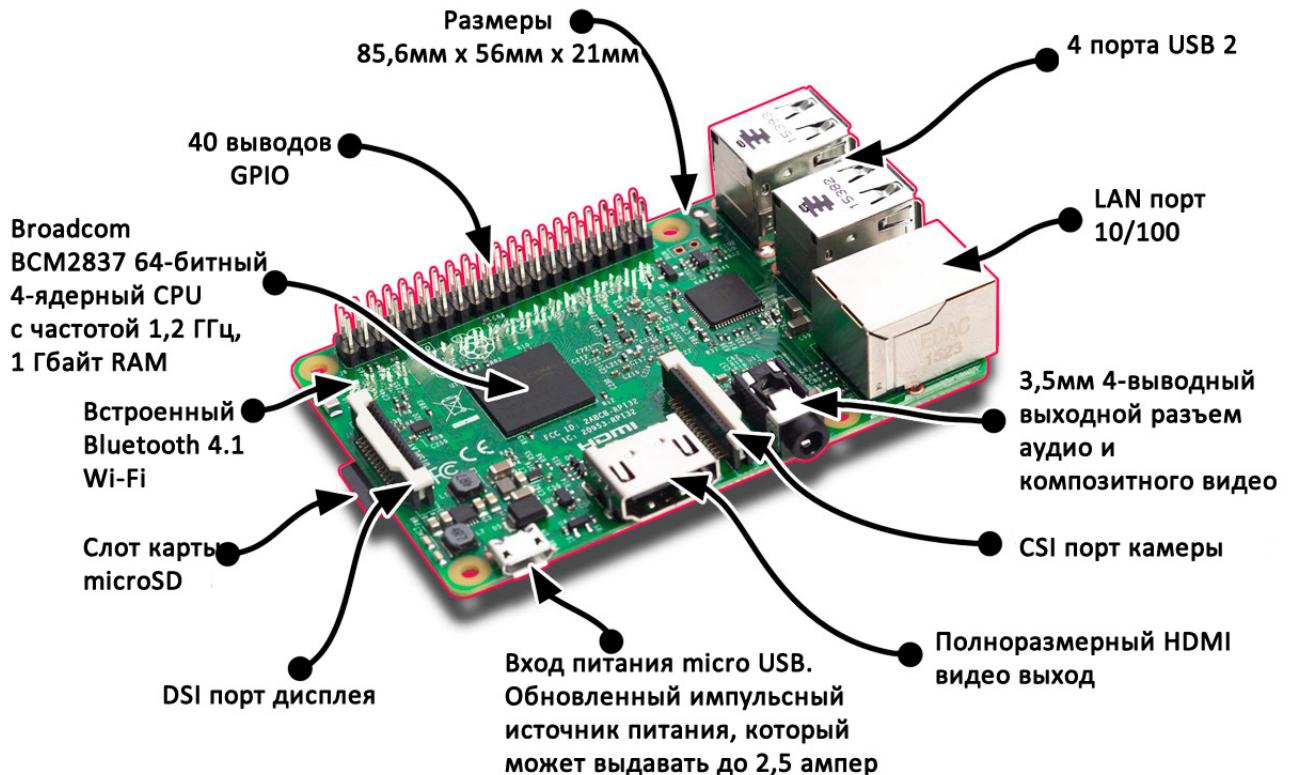


Figure 2: Расположение компонентов на Raspberry Pi 3 Model B

#### 3.2 Подключение периферийных устройств

Для подключения монитора или телевизора используется композитный видеовыход или разъём HDMI. Разрешение варьируется от 640×350 (EGA) до 1920×1200 (WUXGA) для HDMI. Композитный выход работает в форматах PAL и NTSC. Raspberry Pi 3 Model B предоставляет 4 USB-порта, объединённых внутренним хабом. К ним, помимо прочего, можно подключить клавиатуру и мышь.

Для экономии ресурсов центрального процессора, Raspberry Pi предлагает подключения штатных модулей через 15-пиновые слоты:

- CSI-2 — для подключения камеры по интерфейсу MIPI;

- DSI — для подключения штатного дисплея.

В качестве низкоуровневых интерфейсов доступны:

- 40 портов ввода-вывода общего назначения;
- UART (Serial);
- I<sup>2</sup>C/TWI;
- SPI с селектором между двумя устройствами;
- пины питания: 3,3 В, 5 В и земля.

Для коммуникации на Raspberry Pi 3 Model B доступны интерфейсы:

- Ethernet на 10/100 Мбит с выходом на стандартное гнездо 8P8C;
- Wi-Fi 802.11n и Bluetooth 4.1, обеспечиваемые микросхемой Broadcom BCM43438.

### 3.3 Raspberry Pi Camera Module Rev 1.3

#### 3.3.1 Описание камеры



Figure 3: Raspberry Pi Camera Module Rev 1.3

Камера Raspberry Pi подключается непосредственно к разъему CSI на Raspberry Pi. Она способен обеспечить качественное изображение с разрешением 5 МП или запись HD-видео 1080р со скоростью 30 кадров в секунду. Модуль камеры Raspberry Pi оснащен 5-мегапиксельным датчиком Omnivision 5647 в модуле фиксированной фокусировки. Модуль подключается к Raspberry Pi с помощью 15-контактного ленточного кабеля к специальному 15-контактному последовательному интерфейсу камеры MIPI (CSI), который был разработан специально для взаимодействия с камерами. Шина CSI способна обеспечивать чрезвычайно высокую скорость передачи данных и передает пиксельные данные напрямую процессору. Что касается неподвижных изображений, камера способна воспроизводить статические изображения 2592 x 1944 пикселей, а также поддерживает видео 1080р при 30 кадрах в секунду, 720р при 60 кадрах в секунду и 640x480р при 60/90 кадрах.

### 3.3.2 Принципиальная схема

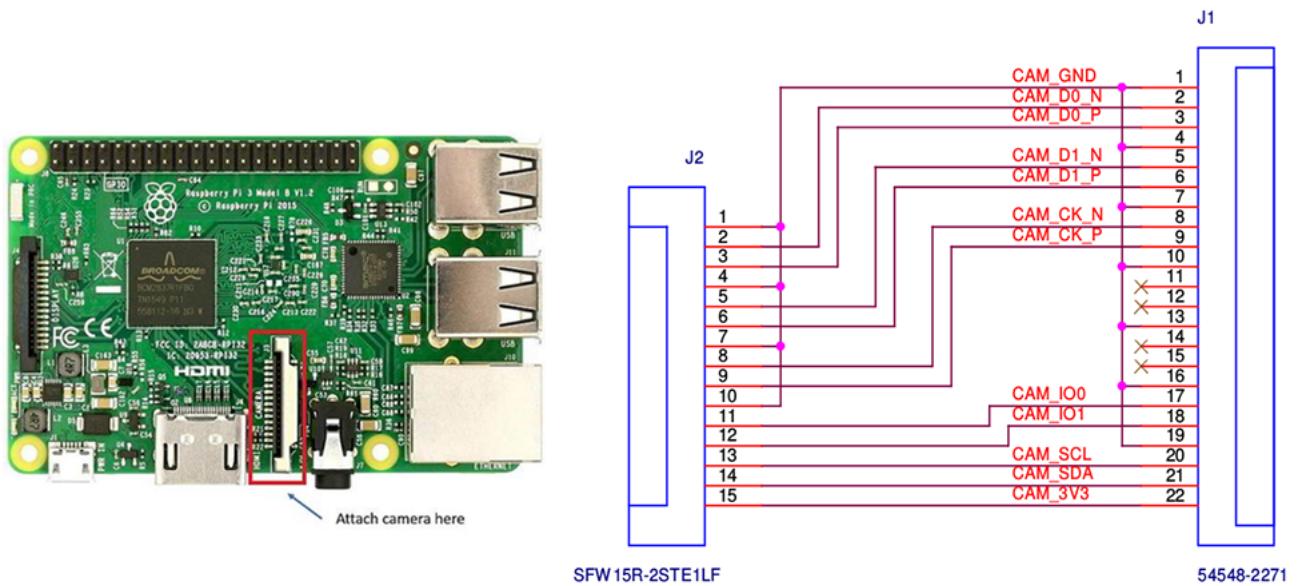


Figure 4: Подключение камеры в CSI порт

### 3.3.3 Технические характеристики

- Размер:  $25 \times 24 \times 9$  mm
- Разрешение: 5 мегапикселей
- Видеорежимы: 1080p30, 720p60, 640×480p60/90
- Сенсор: OmniVision OV5647
- Разрешение сенсора:  $2592 \times 1944$  пикселей
- Отношение сигнал/шум: 36 dB
- Фиксированный фокус: От 1 метра
- Фокусное расстояние: 3.60 mm
- Горизонтальное поле зрения: 53.5 градусов
- Вертикальное поле зрения: 41.4 градусов
- Рабочая температура: от -20 до +70 °C
- Цвет: 24-Bit True color
- Структура пикселов: Байеровская

### 3.3.4 Программные возможности

- Формат изображения: JPEG, GIF, BMP, PNG, YUV420
- Формат видео: raw h.264
- Эффекты: negative, solarise, posterize, whiteboard, blackboard, sketch, denoise, emboss, oil-paint, hatch, gpen, pastel, watercolour, film, blur, saturation
- Режимы экспозиции: auto, night, nightpreview, backlight, spotlight, sports, snow, beach, very-long, fixedfps, antishake, fireworks
- Режимы баланса белого: off, auto, sun, cloud, shade, tungsten, fluorescent, incandescent, flash, horizon

## 3.4 Макетная плата MB102

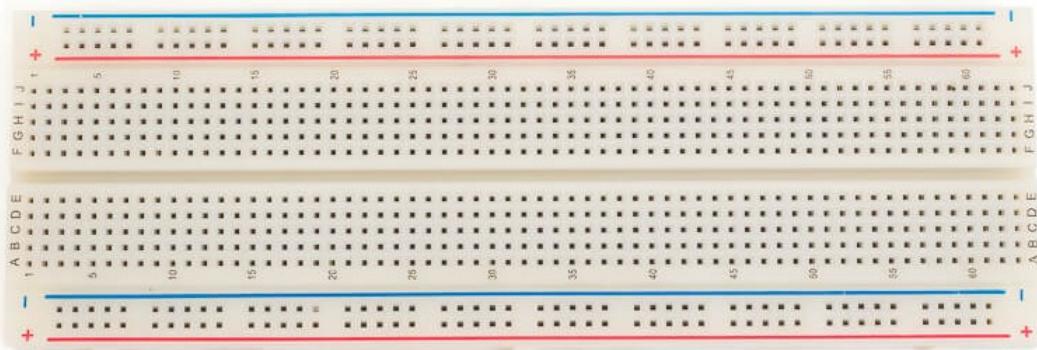


Figure 5: Макетная плата MB102

Макетная плата, имеющая четыре пары рель для подключения питания и заземления позволяет создать функционирующий макет микросхемы. В платформе имеется 830 отверстий для простой установки контактов между различными элементами прототипируемого устройства. Центральное поле платформы имеет пять рядов контактов, в каждом из которых 126 отверстий. Еще два ряда по 100 контактов расположены по бокам макетной платы.

### 3.4.1 Технические характеристики

- Размеры платформы: 165 x 55 x 8,5 mm
- Вес: 54 г
- Максимальное сопротивление контактов: 100 мОм
- Сопротивление изолятора: 1000 мОм
- Диапазон рабочих температур: от -20 до 80 °C
- Максимально допустимая кратковременная температура: +150°C
- Количество контактных отверстий: 830

### 3.4.2 Принципиальная схема

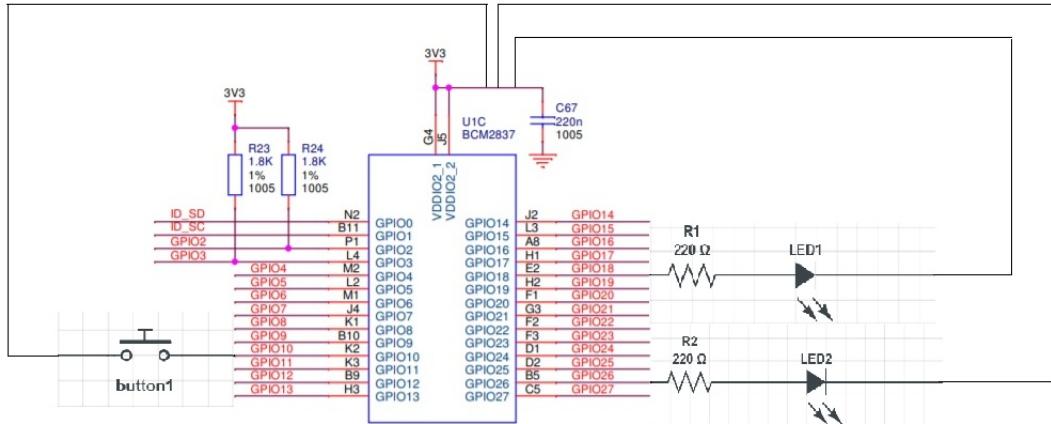


Figure 6: Подключение кнопки и светодиодов

## 4 Интерфейсы

### 4.1 GPIO

В нашем проекте для работы кнопки и светодиодов используется интерфейс GPIO (General Purpose Inputs/Outputs) – выводы общего назначения. GPIO — это группа контактов, которыми можно управлять программным образом. Причем управлять можно и простыми процессами, например, включение/выключение светодиода, и весьма сложными — обмен данными с периферийными устройствами по специализированным протоколам.

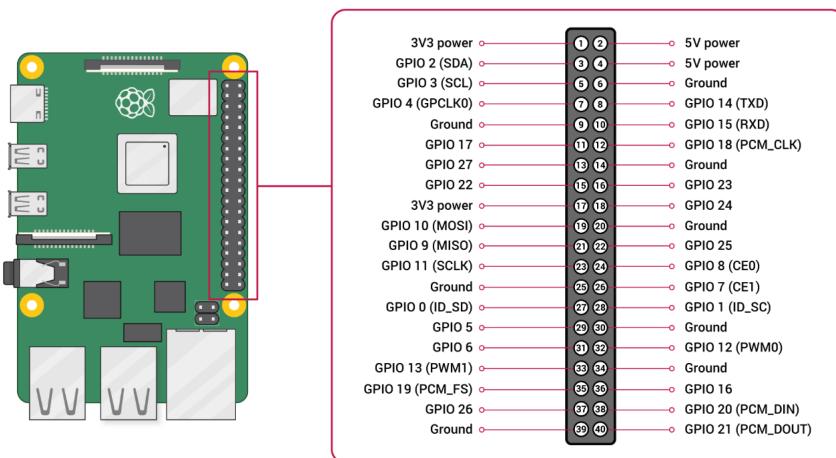


Figure 7: Распиновка GPIO выходов

В терминах цифровой электроники управлять — значит менять на выводе уровень напряжения. Другими словами, все что мы можем сделать с помощью программы — это соединить желаемый вывод либо с контактом питания ( $+3.3\text{ V}$ ), либо с землей (Gnd). Изобразим это на принципиальной схеме:

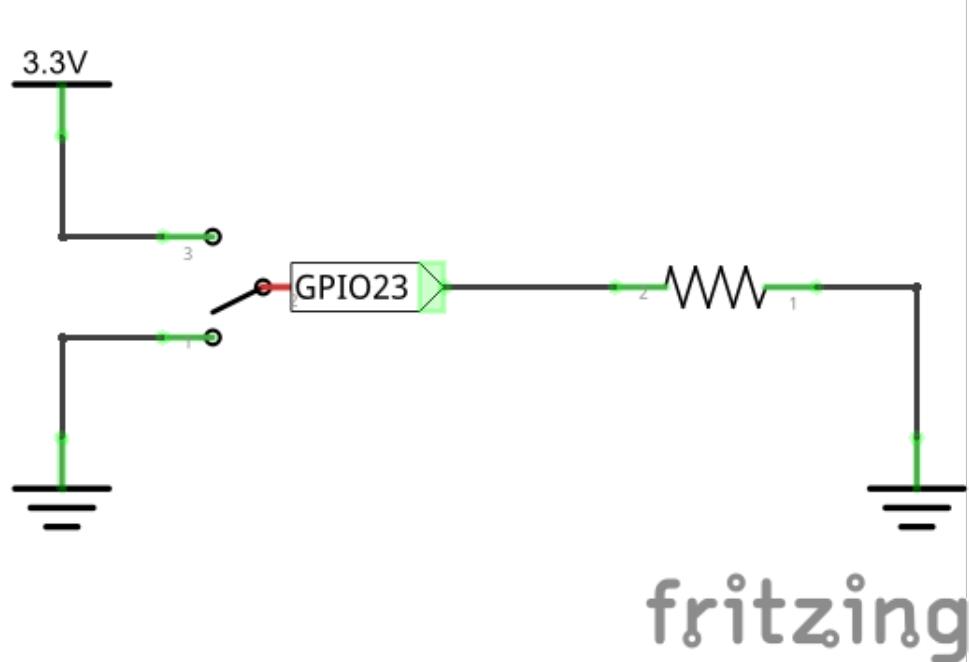


Figure 8: Управление GPIO выходом

На схеме имеется резистор, соединенный справа с землей — это наша нагрузка. Вместо резистора может быть светодиод, реле, зуммер, и т.п. Вывод GPIO23 и переключатель прямо за ним символизируют внутреннее устройство каждого вывода общего назначения.

#### 4.1.1 Учет падения напряжения

- Рабочее напряжение светодиода - 2 – 2.2 В,
- Потребляемый ток светодиода - 20 мА,
- Выходной ток Raspberry Pi - 3.3 В.

$$3.3V - 2V = 1.3V, \quad (1)$$

$$R = \frac{U}{I} = \frac{1.3V}{0.02A} = 65\Omega \quad (2)$$

Настолько нужно понизить напряжение и, получив необходимое сопротивление резистора 65 Ом, взяли ближайший доступный резистор на 220 Ом.

#### 4.1.2 PULL-UP / PULL-DOWN резисторы

Для установки начального состояния GPIO выхода при подключении кнопки используется внутренний подтягивающий резистор Raspberry Pi. Имеются два режима состояния, которые можно задавать программно:

- Подтягивает пин к напряжению 3,3V

```
GPIO.setup(1, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

- Подтягивает пин к земле

```
GPIO.setup(1, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

В нашем проекте пин изначально подтянут к напряжению, и, пока кнопка не нажата, GPIO вход считывает значение TRUE, но при опускании кнопки напряжение на входе понижается и значение меняется на FALSE.

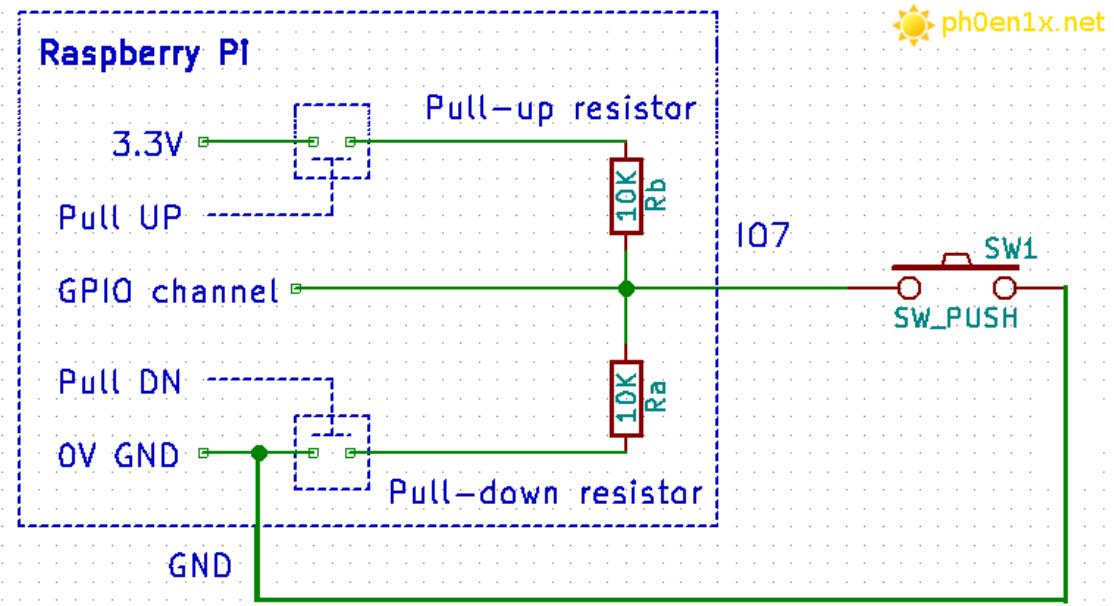


Figure 9: Подключение кнопки с использованием подтягивающего резистора

## 4.2 CSI

Для начала ответим на вопрос, почему мы решили подключить камеру в CSI порт, а не через USB 3.0. Для нашей системы требуется высокая скорость анализа изображений, так как мы и так имеем некоторый замедлитель в виде распознавания лица, а Camera Serial Interface пин как раз обеспечивает поступление изображение с камеры сразу на графический процессор, без промежуточной обработки, как в случае с USB 3.0. Грубо говоря CSI и создан для подобных задач.

Формирование изображения на матрице камеры происходит при помощи фильтра Байера. Это двумерный массив цветных фильтров, которыми накрыты фотодиоды фотоматрицы. Фильтр состоит из 25% красных элементов, 25% синих и 50% зеленых элементов.

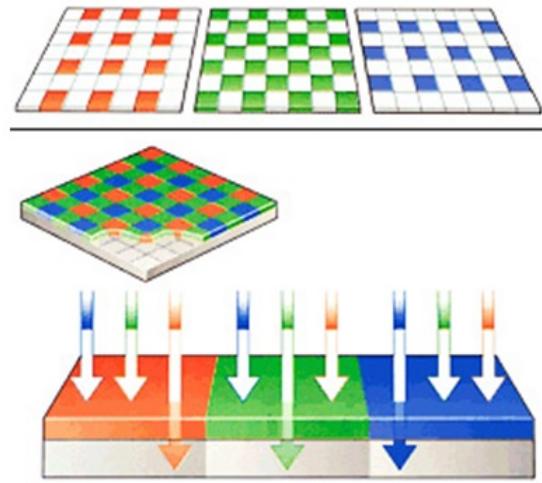


Figure 10: Фильтр Байера

Далее на рисунке представлена общая схема взаимодействия камеры и Raspberry Pi.

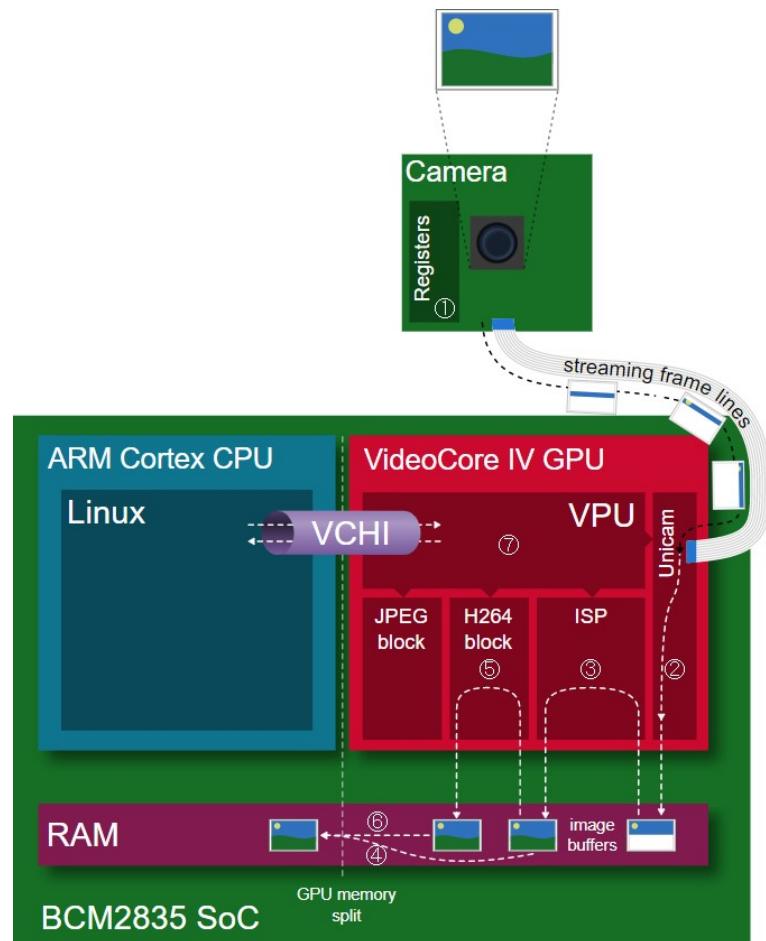


Figure 11: Передача изображения

1. Модуль камеры подключается к плате посредством пятнадцатипинового плоского шлейфа

длиной 15 см через последовательны интерфейс камеры (CSI), который имеет достаточную скорость передачи видеоданных в форматах до 1080р при 30 к/сек или 720р при 60 к/сек.

2. А через интерфейс I<sup>2</sup>C можно послать команды управления камере.
3. На плате также есть CSI-2 приемник, который также называют Unicam, через него и просходит принятие и отправка информации между "малинкой" и камерой.
4. ISP (Image Signal Processor) конвертирует массив принятый байтов в изображение, которое уже способен воспринять человек. Этот компонент также взаимодействует с некоторыми алгоритмами обработки изображения и сбора метаданных.

На следующем рисунке более подробно демонстрируется связь интерфейсов платы и камеры.

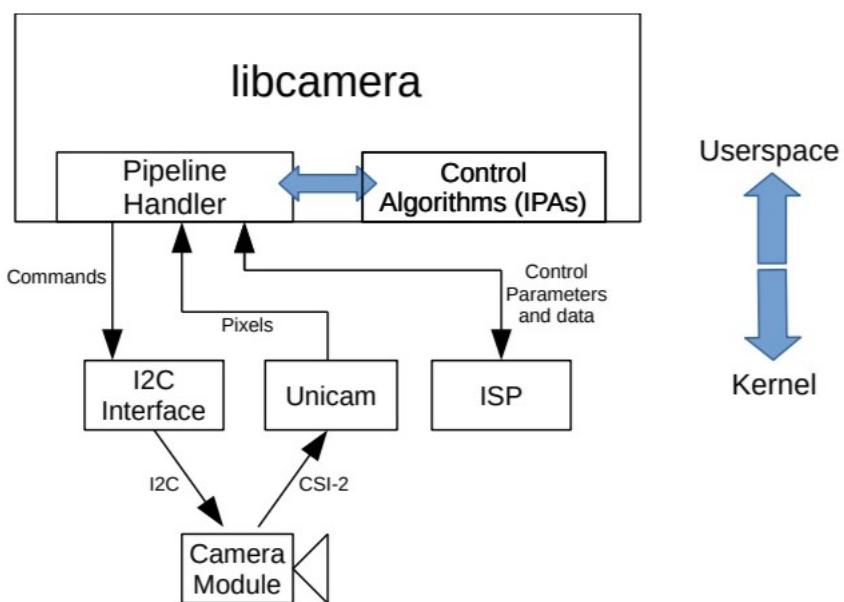


Figure 12: UNICAM

#### 4.2.1 ПО для работы с камерой

- **ffmpeg** - для кодеков,
- **raspivid** - команда для записи видео,
- **raspistill** - команда для захвата изображения (фото),
- **PiCamera** - библиотека для Python.

## 5 Удаленное подключение к Raspberry Pi

Для комфортной командной работы над проектом было решено настроить удаленный способ подключения к Raspberry Pi. Существует два распространенных способа удаленного управления:

- Управление платой по протоколу SSH.
- Подключение к RPi с помощью VNC сервера.

Так как нам требовалось возможность работать в полноценном графическом режиме, был выбран второй способ.

## 5.1 Настройка сервера

Был установлен VNC сервер при помощи следующей команды:

```
sudo apt-get install realvnc-vnc-server realvnc-vnc-viewer
```

Затем в настройках RPi был включен VNC сервер:

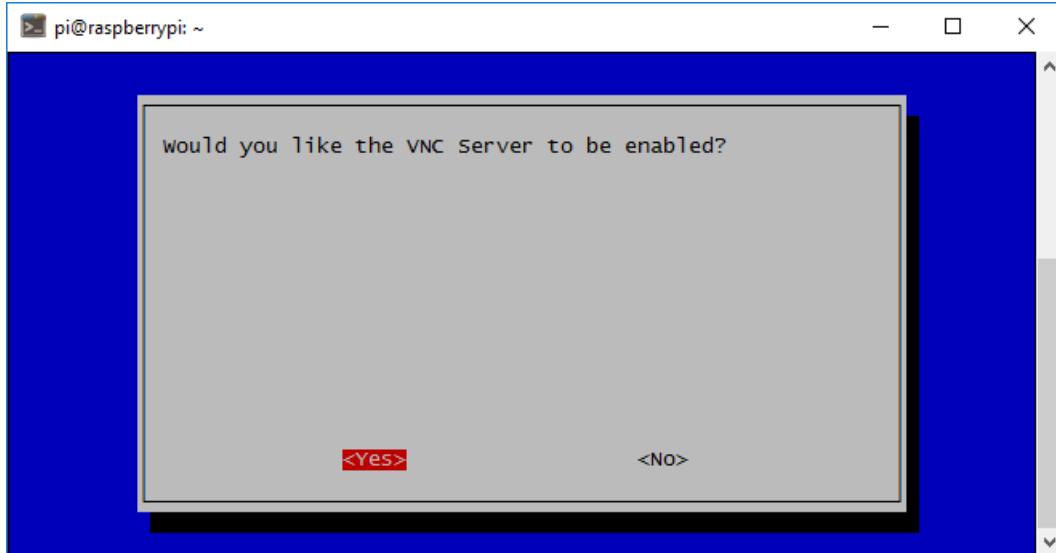


Figure 13: Включение VNC

Запустили сервер при помощи команды `vncserver`. В итоге мы увидели сообщение об удачном запуске сервера с IP-адресом и номером порта:

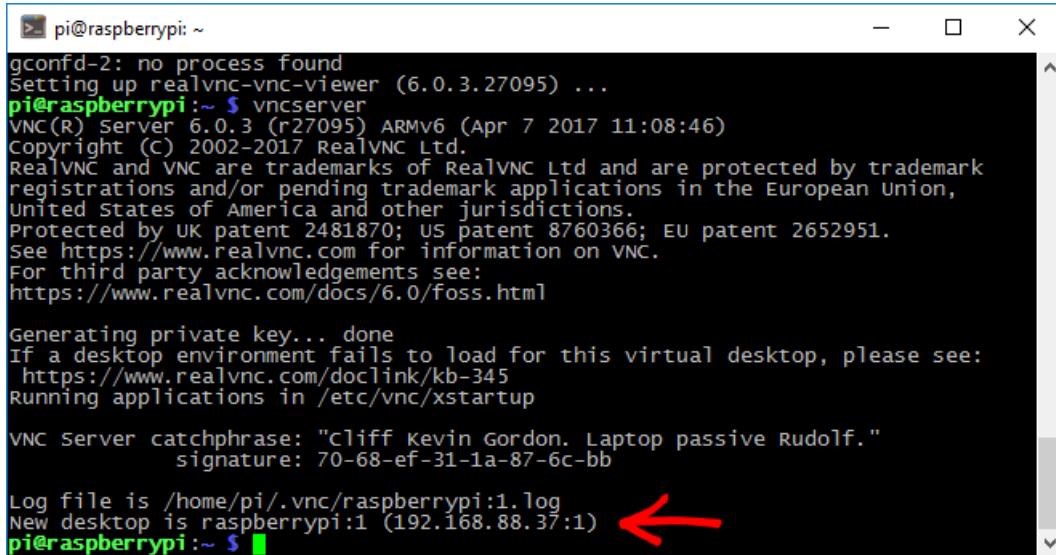


Figure 14: Номер порта

Затем для удобства было решено прописать запуск VNC-сервера в автозагрузку Raspbian, чтобы не приходилось запускать его вручную после каждой перезагрузки. Для этого мы перешли в папку с конфигурациями пользователя, создали директорию `autostart` и в ней файл автозагрузки `realvnc.desktop` и прописали следующее:

```
[Desktop Entry]
Type=Application
Name=RealVNCServer
Exec=vncserver :1
StartupNotify=false
```

Теперь при каждой загрузке графического интерфейса этот файл будет выполнять команду запуска VNC сервера.

## 5.2 Настройка клиента

На компьютеры участников проекта был скачан VNC-клиент VNC Viewer. Через File -> New connection создали подключение к Raspberry Pi, прописав его IP-адрес и порт, на котором прописан VNC-сервер.

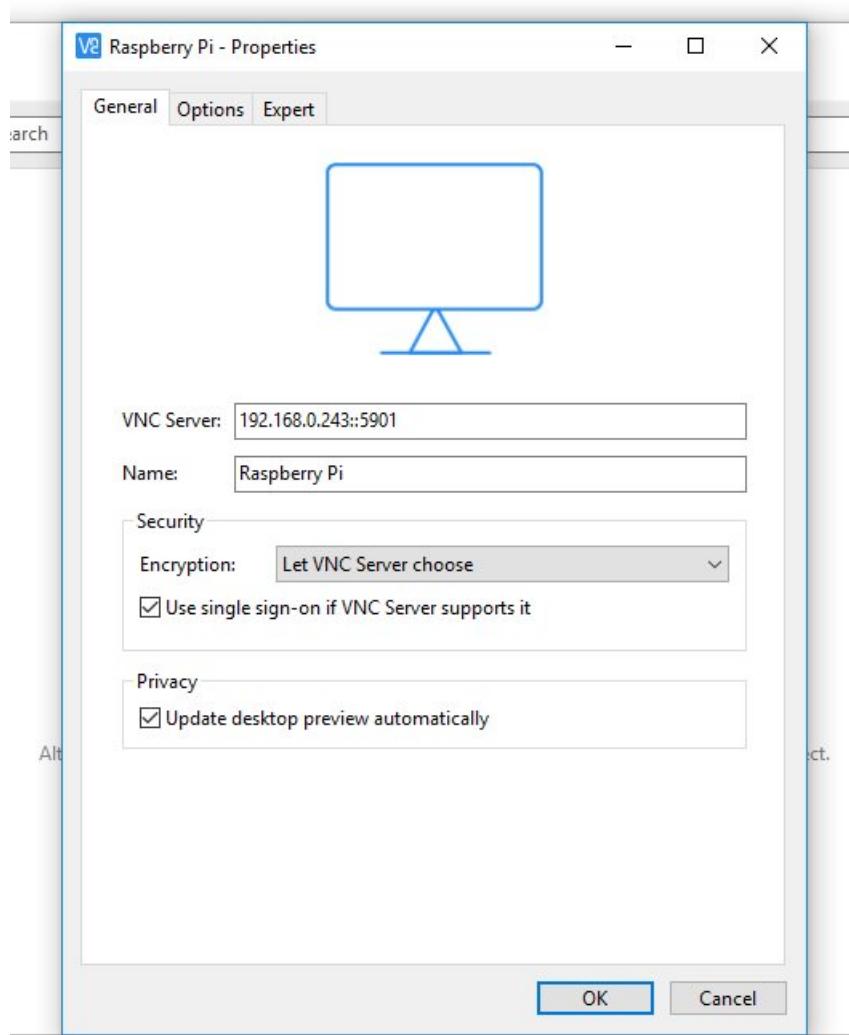


Figure 15: VNC Viewer

После ввода имени пользователя и пароля мы получили полный доступ к графическому интерфейсу RPi:

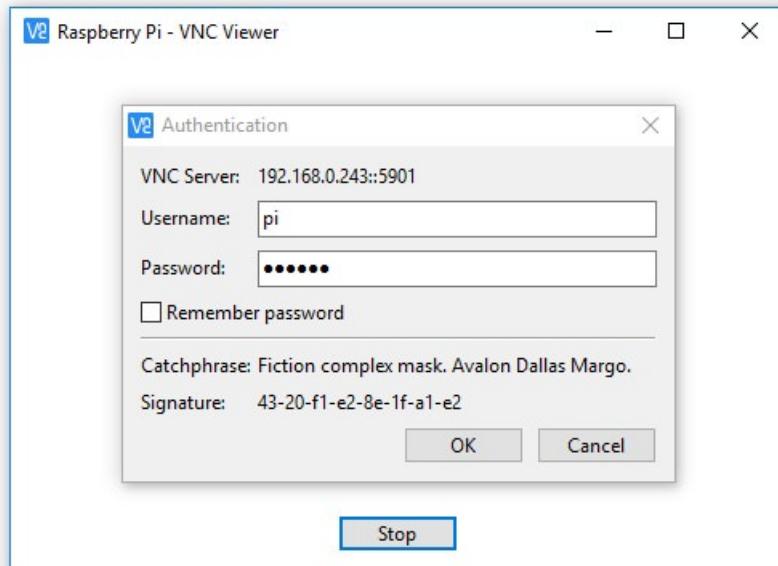


Figure 16: VNC Viewer авторизация

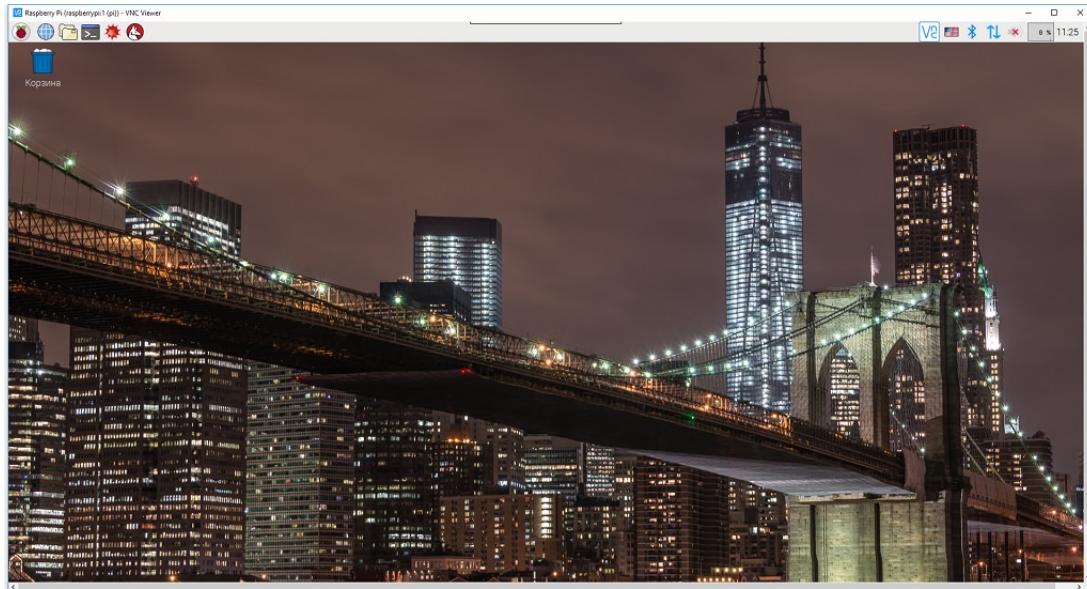


Figure 17: VNC Viewer успешное подключение

## 6 Программная реализация: исходный код программы на GitHub

Как было сказано ранее, языком программирования был выбран Python, а для решения задачи распознавания лиц была скачена библиотека OpenCV и OpenCV-contrib.

Рспознавание лиц мы спроектировали на основе *алгоритма Виолы-Джонса*, в основе которого лежат примитивы Хаары. Метод использует представление изображения в интегральном виде, которое позволяет вычислять быстро, необходимые объекты, с помощью признаков Хаара. Алгоритм применяет бустинг для выбора наиболее подходящих признаков для искомого объекта на данной части заданного изображения. Метод использует классификатор, на вход которого поступают все признаки, затем выдаётся результат «верно» либо «ложь», то есть, принадлежит ли выделенный объект искомому классу или нет. Для быстрого отбрасывания окон, где найден объект, используются каскады признаков. Алгоритм распознает черты лица под небольшим углом, примерно до 30 градусов.

## 6.1 Описание схемы реализации

Если обобщить, то наша программа состоит в последовательном вызове пяти процедур:

1. Сбор данных (добавление нового объекта в базу данных).
2. Обучение нейросети на основе обновленных данных.
3. Распознавание (предсказание в процентах).
4. Регистрация нажатия кнопки.
5. Включение светодиода.

А в целом схема системы выглядит следующим образом (план-чертеж):

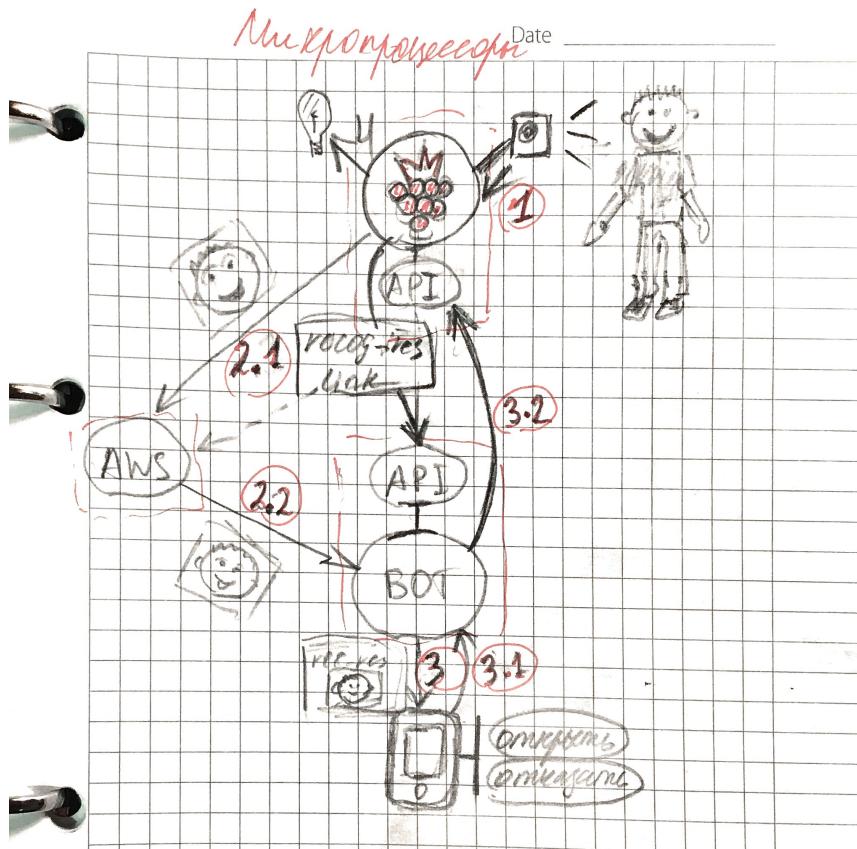


Figure 18: Схема программной реализации

Где

1. Нажатие кнопки - работает 3-я и 4-я программы.
2. Передача изображения и информации об объекте на нем хозяину.
3. Ответ пользователя - работает 5-я программа.

Регистрация нового гостя инициируется из бота, в этом случае срабатывают 1-я и 2-я программы по-порядку.

### 6.1.1 Временная диаграмма активности компонентов системы

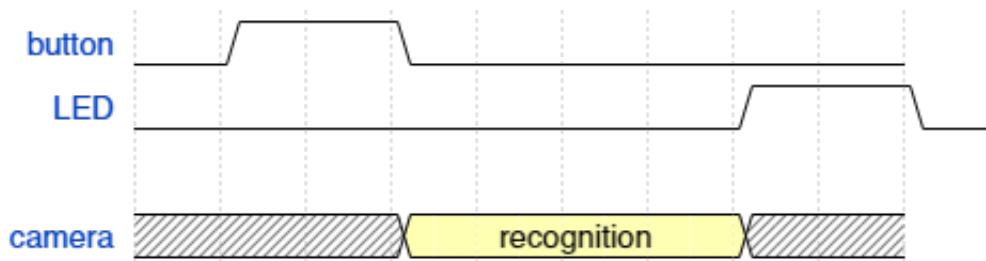


Figure 19: Активность компонентов

### 6.1.2 Функционал бота

- **Предоставление доступа.** При нажатии кнопки бот должен оповестить пользователя об этом и сообщить информацию вида: изображение нажавшего кнопку и результат распознавания. К сообщению прикреплены кнопки “Открыть” и “Не впускать”. Ответ отправляется в малину, которая включает соответствующий световой сигнал.
- **Регистрация человека.** Регистрируемый должен находиться перед камерой, чтобы было видно лицо. Далее владелец нажимает кнопку “Новый друг”, программа делает 300 снимков, нейросеть переобучается, владелец вводит имя друга, которое вместе с id пользователя для нейронки заносятся в базу данных.
- **Включить трансляцию.** В боте есть кнопка на клавиатуре (и команда) ”Online”. При нажатии предоставляется ссылке, перейдя по которой открывается трансляция в прямом эфире с камеры. Дополнительно можно сделать пароль для пользователя, чтобы только он мог смотреть. После необходимо завершить трансляцию. Или же она прервется другими операциями.

В реализации бота использовался следующий технологический стек:

- Python + Django
- Ngrok
- Celery + Redis
- TelegramBot API

## 7 Демонстрация работы системы

Все видео, начиная с этапов тестирования отдельных компонентов



Figure 20: Наша плата и камера

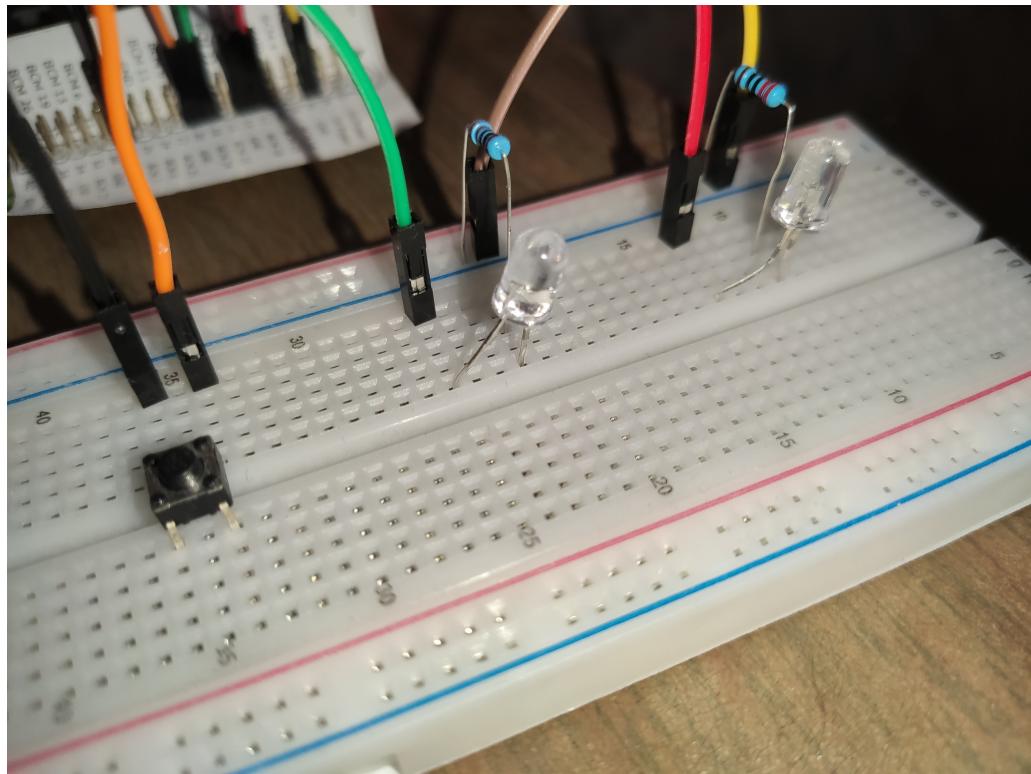


Figure 21: Наши кнопки и светодиоды

## **8 Вывод**

В ходе работы по реализации системы контроля входа в умном доме мы исследовали возможности микрокомпьютера Raspberry Pi 3 Model B, на котором установлен процессор ARM Cortex-A53, подключали к плате различные периферийные устройства, а именно камеру, кнопку и два светодиода, и следили за их правильным функционированием, контролируя процессы через различные интерфейсов (GPIO, CSI, I<sup>2</sup>C).