СОДЕРЖАНИЕ

[Введение 5](#_Toc39833483)

[1 Анализ исходных данных 7](#_Toc39833484)

[2 Теоретические сведения и принципы функционирования отдельных узлов   
 прибора 8](#_Toc39833485)

[2.1 Понятие и принцип действия болометрической матрицы 8](#_Toc39833486)

[2.2 Математическое описание физических принципов работы болометра 10](#_Toc39833487)

[2.3](#_Toc39833488) Понятие изображения…………………………………………………….11

[2.4](#_Toc39833490) Принципы представления и передачи изображений [**Ошибка! Закладка не определена.**](#_Toc39833491)2

[2.5](#_Toc39833492) Обзор принципов действия, структурных решений и микропроцессорной базы современных тепловизоров [**Ошибка! Закладка не определена.**](#_Toc39833493)6

[2.6 Структура микроконтроллерного ядра *ARM* *Cortex* *M*-4 18](#_Toc39833494)8

[2.7 Регистровая модель интерфейса *GPIO* *ARM* *Cortex*-*M*4 18](#_Toc39833495)

[2.8 Принцип функционирования вложенных векторов прерывания](#_Toc39833496)

[и детектора внешних событий микроконтроллера семейства](#_Toc39833497)

[*ARM* *Cortex*-*M*4 19](#_Toc39833498)9

[2.9](#_Toc39833499) Структура и логика функционирования цифрового болометра *LEPTON* *FLIR* [**Ошибка! Закладка не определена.**](#_Toc39833500)1

[2.10](#_Toc39833501) Физический и канальный уровни интерфейсов *FSMC, I2C* и *SPI*……22

[2.11 Принципы функционирования блока *DMA* прямого доступа  
 к](#_Toc39833502) [памяти 24](#_Toc39833503)4

[2.12](#_Toc39833504) Структура и логика функционирования дисплейного модуля *HY32D* на базе видеопроцессора *ILI9341* [**Ошибка! Закладка не определена.**](#_Toc39833505)5

[3 Разработка структурной электрической схемы ведомого мобильного тепловизора 26](#_Toc39833506)6

[3.1 Обоснование базовых блоков структурной схемы](#_Toc39833507) мобильного тепловизора  [**Ошибка! Закладка не определена.**](#_Toc39833508)6

[3.2 Обоснование связей структурной схемы мобильного](#_Toc39833509) тепловизора  [**Ошибка! Закладка не определена.**](#_Toc39833510)7

[4 Разработка принципиальной](#_Toc39833511) [электрической схемы](#_Toc39833512) мобильного тепловизора [**Ошибка! Закладка не определена.**](#_Toc39833513)9

[4.1 Обоснование выбора САПР для разработки принципиальной   
 электрической схемы 29](#_Toc39833514)9

[4.2 Описание используемых библиотечных элементов и процесса  
 их создания 29](#_Toc39833515)9

[4.3 Обоснование выбора базовых компонентов принципиальной схемы мобильный тепловизор 31](#_Toc39833516)2

[4.4 Обоснование связей принципиальной электрической схемы мобильный тепловизор 31](#_Toc39833517)3

[5 Разработка алгоритма функционирования ведомого мобильный тепловизор 32](#_Toc39833518)4

[5.1 Разработка диаграммы состояний мобильный тепловизор 32](#_Toc39833519)4

[5.2 Разработка схемы алгоритма ведомого мобильный тепловизор 32](#_Toc39833520)4

[6 Разработка конструкции проектируемого прибора 33](#_Toc39833521)5

[6.1 Выбор и обоснование элементной базы 33](#_Toc39833522)5

[6.2. Выбор и обоснование конструктивных элементов и](#_Toc39833523)

[установочных изделий. 34](#_Toc39833524)6

[7 Расчёт конструктивно-технологических](#_Toc39833525) [параметров проектируемого   
 прибора 35](#_Toc39833526)8

[7.1 Проектирование печатного модуля 35](#_Toc39833527)8

[7.2 Выбор и обоснование материалов конструкции и защитных](#_Toc39833528) покрытий,

[маркировки деталей и сборочных единиц 35](#_Toc39833529)8

[8 Применение средств автоматизированного проектирования при разработке  
 прибора](#_Toc39833530) 40

[Заключение](#_Toc39833531) 41

[Список используемых источников 39](#_Toc39833532)2

[Приложение А (обязательное) Перечень элементов к схеме электрической   
 принципиальной 41](#_Toc39833534)3

[Приложение Б (обязательное) Спецификация к сборочному чертежу печатной платы 43](#_Toc39833535)

[Приложение В (обязательное) Ведомость курсового проекта 46](#_Toc39833536)

[Приложение Г (обязательное) Визуализированная трёхмерная](#_Toc39833537)

[модель печатной платы 48](#_Toc39833537)

[Приложение Д (обязательное) Текст программы 50](#_Toc39833538)

[Приложение Е (обязательное) Проверка в системе «Антиплагиат» 7](#_Toc39833530)0

ВВЕДЕНИЕ

Тепловизор – измерительный прибор, который позволяет видеть тепловое (инфракрасное) излучение окружающих объектов в любое время суток, измерять температуру в любой точке на поверхности с точностью 0,1°С и выше. Основное предназначение тепловизора - бесконтактное измерение температуры объектов живой и неживой природы, поиск неисправностей оборудования и электрики, недочётов строительства. Тепловизионные камеры создают чёткие тепловые изображения, основываясь на разнице температур. А сложные алгоритмы простых с виду камер считывают с этих изображений температурные значения. Самые горячие места окрашиваются в красный, жёлтый и оранжевый цвета, холодные в синий и чёрный. Популярность тепловизоры обрели благодаря возможности применения во всех отраслях жизнедеятельности человека. Самые популярные области применения это строительство, охота, медицина и промышленность. Всё чаще тепловизоры используются и в быту для обследования квартир и частных домов, позволяют находить места утечек тепла и неполадки в электрике.

Человеческий глаз видит очень маленькую часть электромагнитного спектра. Наши «детекторы» несовершенны, мы воспринимаем только видимый свет, инфракрасное излучение находится за пределами возможностей наших глаз. Видимый свет занимает диапазон длин волн электромагнитного излучении от 0,38 до 0,76 мкм, причем середина этого диапазона приходится на длину волны 0,55 мкм, которая соответствует максимуму солнечного излучения. Поскольку весь диапазон электромагнитного излучения простирается от ангстрем до сотен километров и фактически не ограничен ни «слева», ни «справа», человеческая цивилизация на протяжении своей технологической истории стремится освоить те диапазоны излучения, где глаз человека бессилен.

ИК-излучение находится в диапазоне между видимым светом и СВЧ- диапазоном электромагнитного спектра. Инфракрасное (ИК) излучение занимает диапазон длин волн от 0,76 до 1000 мкм. Основным источником инфракрасного излучения является тепло или тепловое излучение. Любой предмет с температурой выше абсолютного нуля (-273,15 °C или 0 градусов Кельвина) испускает излучение в ИК-области. Даже объекты, которые нам кажутся очень холодными, такие как кубики льда, испускают ИК-лучи. Иными словами, если бы глаз человека видел в ИК диапазоне, то мы могли бы оценивать температуру объектов, не прикасаясь к ним.  Тепло солнечных лучей, костер или радиатор отопления - все это ИК-излучение. Хотя глаза его не видят, наша подкожная нервная система ощущает это излучение как тепло. Чем теплее объект, тем больше ИК-излучение он испускает. Инфракрасное излучение, исходящее от объекта, фокусируется объективом тепловизора на инфракрасном детекторе. Этот детектор передает сигнал в электронный блок для обработки изображения. Электронный блок преобразует сигналы, поступающие от датчика, в тепловизионное изображение, которое отображается в видоискателе, на стандартном мониторе или ЖК-дисплее. А за счёт преобразования инфракрасного изображения в радиометрическое, считываются температурные значения с тепловизионного изображения.

1 Анализ исходных данных

Основанием для разработки устройства стало задание на курсовое проектирование. В ходе проекта необходимо создать мобильный тепловизор которое может использоваться в ремонтно-технических операций. Так же может использоваться в медицине.

Обязательные параметры:

* потребляемый ток не более 100 мА;
* возможность зарядки от внешнего источника питания напряжением 5В посредством разъёма *microUSB*.

Изделие должно сохранять свои параметры в пределах норм, установленных техническими заданиями, стандартами или техническими условиями в течение сроков службы и сроков сохраняемости, указанных в техническом задании после или в процессе воздействия климатических факторов, значения которых установлены ГОСТ 15150-69 УХЛ3.1.

Показатели надежности должны соответствовать ГОСТ 27.003-90 и заданным значениям при нормальных климатических условиях (температура окружающего воздуха) 20 °С, относительная влажность 98%, атмосферное давление от 84 до 107 кПа (450–750 мм рт.ст.) с напряжением сети 220 В с частотой переменного тока 50 Гц.

Конструктивно, мобильный тепловизорвыполнено в виде модуля, содержащего электрорадиоэлементы.

Габаритные размеры прибора – не более 100 х 80 х 40 мм.

Масса прибора не более 0,3 кг.

2 Теоретические сведения и принципы

**ФУНКЦИОНИРОВАНИЯ ОТДЕЛЬНЫХ УЗЛОВ**

**ПРИБОРА**

Тепловизор — устройство для наблюдения за распределением [температуры](https://ru.wikipedia.org/wiki/Температура) исследуемой поверхности. Распределение температуры отображается на дисплее как цветная картинка, где разным температурам соответствуют разные [цвета](https://ru.wikipedia.org/wiki/Цвет). Изучение тепловых изображений называется [термографией](https://ru.wikipedia.org/wiki/Термография).

2.1  [Понятие и принцип действия болометрической матрицы](#_Toc39833486)

Болометр — сенсор, изменяющий сопротивление в зависимости от температуры.

Улавливая эти изменения, электронная схема формирует выходной сигнал, представляющий собой диаграмму распределения температур по сцене. Для этого на матрицу периодически подаётся ток смещения и измеряется величина полученного напряжения. Наиболее распространённым материалом для изготовления болометрических детекторов в настоящее время является поликристаллическая окись ванадия. Благодаря прогрессу технологий производства полупроводниковых компонентов, производители тепловизоров на микроболометрах сумели эффективно решить проблему разброса параметров отдельных пикселов при миниатюризации приборов (это основной источник собственных шумов матриц). Современные неохлаждаемые камеры способны уловить разницу температур в 0,07 К. Одна из технических тонкостей в проектировании подобных матриц видеокамеры внутреннего купольного типа — расчёт размера ножек ячейки: они должны быть достаточно малы, чтобы обеспечить заданные теплоизоляционные свойства, и достаточно велики для того, чтобы успевать обеспечить отвод тепла в течение интервала сканирования пикселов (в противном случае на следующее измерение сопротивления будет влиять результат предыдущего).

Помимо отсутствия требований к охлаждению, микроболометры имеют ещё одно существенное преимущество перед фотоэлектрическими сенсорами. Диаграмма их спектральной чувствительности имеет практически плоскую форму во всём диапазоне воспринимаемых ими ИК-лучей. Современные микроболометрические матрицы обладают временем реакции в несколько сотых долей секунды, что позволяет им обеспечивать формирование стандартного телевизионного сигнала в реальном масштабе времени со стандартной частотой кадров. Это частично компенсирует их отставание от охлаждаемых детекторов по чувствительности.

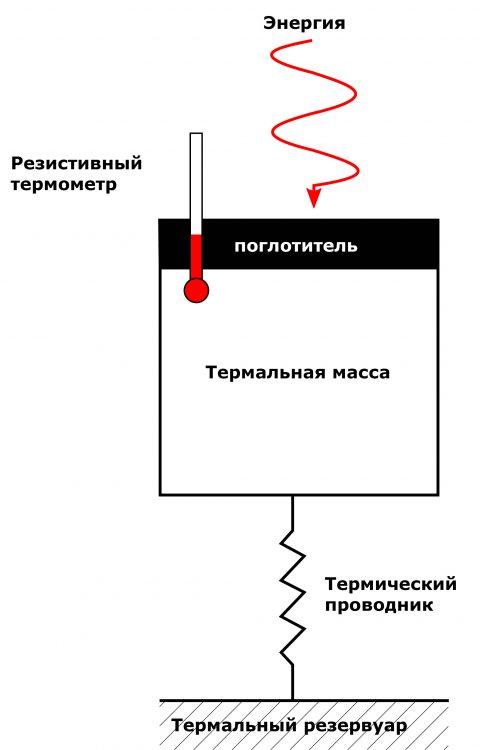
[](http://www.artly.ru/wp-content/uploads/2012/01/bolomer.jpg)

Рисунок 2.1 – Схема работы болометрической матрицы

Несмотря на отсутствие необходимости в охлаждении, многие разновидности микроболометров нуждаются в температурной стабилизации, реализуемой при помощи термовентилятора. Как правило, сенсор и вентилятор заключены в герметичный корпус, термически изолированный от остальных компонентов камеры. Это позволяет гарантировать, что на сенсор будут попадать лишь ИК-лучи, испускаемые объектами в сцене. Ряд производителей пользуются схемами электронной температурной компенсации, позволяющими исключить применение термовентиляторов.Сравнительно недавно возник и совершенно новый вид детекторов ИК-излучения на базе микро-кантилеверной технологии. По чувствительности они в десять раз превосходят микроболометры, приближаясь к теоретическим пределам чувствительности материалов к инфракрасным лучам.

2.2 Математическое описание физических принципов работы болометра

Принцип действия болометра основан на изменении [электрического сопротивления](https://ru.wikipedia.org/wiki/Электрическое_сопротивление) термочувствительного элемента вследствие [нагревания](https://ru.wikipedia.org/wiki/Нагрев) под воздействием поглощаемого потока [электромагнитной](https://ru.wikipedia.org/wiki/Электромагнитное_излучение) [энергии](https://ru.wikipedia.org/wiki/Энергия)[[2]](https://ru.wikipedia.org/wiki/Болометр" \l "cite_note-bolsun-2).

Основной компонент болометра — очень тонкая пластинка (например, из [платины](https://ru.wikipedia.org/wiki/Платина) или другого проводящего материала), зачернённая для лучшего [поглощения излучения](https://ru.wikipedia.org/wiki/Поглощение_излучения). Из-за своей малой толщины пластинка под действием излучения быстро нагревается и её [сопротивление](https://ru.wikipedia.org/wiki/Электрическое_сопротивление) повышается. Для измерения малых отклонений сопротивления пластинки её включают в [мостовую схему](https://ru.wikipedia.org/wiki/Измерительный_мост), которую балансируют при отсутствии засветки. [Металлические](https://ru.wikipedia.org/wiki/Металл) болометры часто подсоединяют через [трансформаторный](https://ru.wikipedia.org/wiki/Трансформатор) вход, так как у них очень малое собственное сопротивление.

Первый [полупроводниковый](https://ru.wikipedia.org/wiki/Полупроводниковые_приборы) болометр был создан компанией [Bell](https://ru.wikipedia.org/wiki/Bell_Labs) в годы [Второй мировой войны](https://ru.wikipedia.org/wiki/Вторая_мировая_война). Отличался простотой, надёжностью и высокой чувствительностью. Был использован в ИК-спектроскопии и теплопеленгации.

Первые [терморезистивные](https://ru.wikipedia.org/wiki/Терморезистор) болометры успешно работали на [искусственных спутниках Земли](https://ru.wikipedia.org/wiki/Искусственный_спутник_Земли), но позже были вытеснены [пироэлектрическими](https://ru.wikipedia.org/wiki/Пироэлектрический_эффект) приёмниками.

В качестве материалов для металлических болометров используют платину, [никель](https://ru.wikipedia.org/wiki/Никель), [золото](https://ru.wikipedia.org/wiki/Золото), для полупроводниковых — сплавы окислов [никеля](https://ru.wikipedia.org/wiki/Никель), [кобальта](https://ru.wikipedia.org/wiki/Кобальт), [марганца](https://ru.wikipedia.org/wiki/Марганец).

Полупроводниковый болометр состоит из двух плёночных (толщиной до 10 [мкм](https://ru.wikipedia.org/wiki/Мкм)) [термисторов](https://ru.wikipedia.org/wiki/Термистор). Один из [термисторов](https://ru.wikipedia.org/wiki/Термистор), непосредственно подвергающийся облучению, является активным. Второй — компенсационный. Он [экранирован](https://ru.wikipedia.org/w/index.php?title=Экран_(устройство)&action=edit&redlink=1) от внешнего излучения и предназначен для компенсации изменений [температуры](https://ru.wikipedia.org/wiki/Температура) окружающей среды. Оба термистора помещаются в общий герметичный корпус.

[Чувствительность](https://ru.wikipedia.org/wiki/Чувствительность_(техника)) болометра улучшается с понижением температуры чувствительного элемента. В астрономии обычно используются болометры, охлаждаемые до температуры [жидкого гелия](https://ru.wikipedia.org/wiki/Жидкий_гелий).

Основные параметры болометров:

* сопротивление активного термистора при номинальной температуре;
* рабочее напряжение;
* чувствительность при определённой частоте [модуляции](https://ru.wikipedia.org/wiki/Модуляция) [светового потока](https://ru.wikipedia.org/wiki/Световой_поток);
* порог чувствительности;
* [постоянная времени](https://ru.wikipedia.org/wiki/Постоянная_времени);
* уровень собственных шумов — у металлических преобладает тепловой [шум](https://ru.wikipedia.org/wiki/Шум), у полупроводниковых — токовый.

2.3 Понятие изображения

Изображение — двумерное изображение, представленное в цифровом виде. В зависимости от способа описания, изображение может быть [растровым](https://ru.wikipedia.org/wiki/Растровая_графика) или [векторным](https://ru.wikipedia.org/wiki/Векторная_графика).

Растровое изображение — изображение, представляющее собой сетку (мозаику) [пикселей](https://ru.wikipedia.org/wiki/Пиксель) — цветных точек (обычно прямоугольных) на [мониторе](https://ru.wikipedia.org/wiki/Монитор_(устройство)), бумаге и других отображающих устройствах.

Важными характеристиками изображения являются:

* Размер изображения в пикселях — может выражаться в виде количества пикселей по ширине и по высоте (800 × 600 px, 1024 × 768 px, 1600 × 1200 px и т. д.) или же в виде общего количества пикселей (так, изображение размером 1600 × 1200 px состоит из 1 920 000 точек, то есть примерно из двух мегапикселей);
* Количество используемых цветов или [глубина цвета](https://ru.wikipedia.org/wiki/Глубина_цвета) (эти характеристики имеют следующую зависимость: {\displaystyle N=2^{k}}где {\displaystyle N} — количество цветов, {\displaystyle k} — глубина цвета);
* [Цветовое пространство](https://ru.wikipedia.org/wiki/Цветовое_пространство) ([цветовая модель](https://ru.wikipedia.org/wiki/Цветовая_модель))  —  [RGB](https://ru.wikipedia.org/wiki/RGB), [CMYK](https://ru.wikipedia.org/wiki/CMYK), [XYZ](https://ru.wikipedia.org/wiki/Цветовая_модель" \l "Цветовое_пространство_CIE_XYZ), [YCbCr](https://ru.wikipedia.org/wiki/YCbCr)  и др.;
* [Разрешение изображения](https://ru.wikipedia.org/wiki/Разрешение_(компьютерная_графика)) — величина, определяющая количество точек (элементов растрового изображения) на единицу площади (или единицу длины).

Векторная графика — способ представления объектов и изображений (формат описания) в [компьютерной графике](https://ru.wikipedia.org/wiki/Компьютерная_графика), основанный на математическом описании элементарных геометрических объектов, обычно называемых примитивами, таких как: точки, линии, [сплайны](https://ru.wikipedia.org/wiki/Сплайн), [кривые Безье](https://ru.wikipedia.org/wiki/Кривая_Безье), круги и окружности, [многоугольники](https://ru.wikipedia.org/wiki/Многоугольник).

Объекты векторной графики являются графическими изображениями математических объектов.

Термин «векторная графика» используется для пояснения различий от [растровой графики](https://ru.wikipedia.org/wiki/Растровая_графика), в которой изображение представлено в виде графической матрицы.

При выводе на матричные устройства отображения (мониторы) векторная графика предварительно преобразуется в растровую графику, преобразование производится программно или аппаратно средствами современных видеокарт.

2.4 Принципы представления и передачи изображений

Графическую информацию можно представлять в двух формах: аналоговой или дискретной. Живописное полотно, цвет которого изменяется непрерывно - это пример аналогового представления, а изображение, напечатанное при помощи струйного принтера и состоящее из отдельных точек разного цвета - это дискретное представление. Путем разбиения графического изображения (дискретизации) происходит преобразование графической информации из аналоговой формы в дискретную. При этом производится кодирование - присвоение каждому элементу конкретного значения в форме кода. При кодировании изображения происходит его пространственная дискретизация. Ее можно сравнить с построением изображения из большого количества маленьких цветных фрагментов (метод мозаики). Все изображение разбивается на отдельные точки, каждому элементу ставится в соответствие код его цвета. При этом качество кодирования будет зависеть от следующих параметров: размера точки и количества используемых цветов. Чем меньше размер точки, а, значит, изображение составляется из большего количества точек, тем выше качество кодирования. Чем большее количество цветов используется (т. е. точка изображения может принимать больше возможных состояний), тем больше информации несет каждая точка, а, значит, увеличивается качество кодирования. Создание и хранение графических объектов возможно в нескольких видах - в виде векторного, фрактального или растрового изображения. Отдельным предметом считается 3D (трехмерная) графика, в которой сочетаются векторный и растровый способы формирования изображений. Она изучает методы и приемы построения объемных моделей объектов в виртуальном пространстве. Для каждого вида используется свой способ кодирования графической информации.

Растровое изображение.

   При помощи увеличительного стекла можно увидеть, что черно-белое графическое изображение, например из газеты, состоит из мельчайших точек, составляющих определенный узор - растр. Во Франции в 19 веке возникло новое направление в живописи - пуантилизм. Его техника заключалась в том, что на холст рисунок наносился кистью в виде разноцветных точек. Также этот метод издавна применяется в полиграфии для кодирования графической информации. Точность передачи рисунка зависит от количества точек и их размера. После разбиения рисунка на точки, начиная с левого угла, двигаясь по строкам слева направо, можно кодировать цвет каждой точки. Далее одну такую точку будем называть пикселем (происхождение этого слова связано с английской аббревиатурой "picture element" - элемент рисунка). Объем растрового изображения определяется умножением количества пикселей (на информационный объем одной точки, который зависит от количества возможных цветов. Качество изображения определяется разрешающей способностью монитора. Чем она выше, то есть больше количество строк растра и точек в строке, тем выше качество изображения. В современных ПК в основном используют следующие разрешающие способности экрана: 640 на 480, 800 на 600, 1024 на 768 и 1280 на 1024 точки. Так как яркость каждой точки и ее линейные координаты можно выразить с помощью целых чисел, то можно сказать, что этот метод кодирования позволяет использовать двоичный код для того чтобы обрабатывать графические данные.

   Если говорить о черно-белых иллюстрациях, то, если не использовать полутона, то пиксель будет принимать одно из двух состояний: светится (белый) и не светится (черный). А так как информация о цвете пикселя называется кодом пикселя, то для его кодирования достаточно одного бита памяти: 0 - черный, 1 - белый. Если же рассматриваются иллюстрации в виде комбинации точек с 256 градациями серого цвета (а именно такие в настоящее время общеприняты), то достаточно восьмиразрядного двоичного числа для того чтобы закодировать яркость любой точки. В компьютерной графике чрезвычайно важен цвет. Он выступает как средство усиления зрительного впечатления и повышения информационной насыщенности изображения. Как формируется ощущение цвета человеческим мозгом? Это происходит в результате анализа светового потока, попадающего на сетчатку глаза от отражающих или излучающих объектов. Принято считать, что цветовые рецепторы человека, которые еще называют колбочками, подразделяются на три группы, причем каждая может воспринимать всего один цвет - красный, или зеленый, или синий.

    Цветовые модели.

   Если говорить о кодировании цветных графических изображений, то нужно рассмотреть принцип декомпозиции произвольного цвета на основные составляющие. Применяют несколько систем кодирования: HSB, RGB и CMYK. Первая цветовая модель проста и интуитивно понятна, т. е. удобна для человека, вторая наиболее удобна для компьютера, а последняя модель CMYK-для типографий. Использование этих цветовых моделей связано с тем, что световой поток может формироваться излучениями, представляющими собой комбинацию " чистых" спектральных цветов : красного, зеленого, синего или их производных. Различают аддитивное цветовоспроизведение (характерно для излучающих объектов) и субтрактивное цветовоспроизведение (характерно для отражающих объектов). В качестве примера объекта первого типа можно привести электронно-лучевую трубку монитора, второго типа - полиграфический отпечаток.

   1) Модель HSB характеризуется тремя компонентами: оттенок цвета(Hue), насыщенность цвета (Saturation) и яркость цвета (Brightness). Можно получить большое количество произвольных цветов, регулируя эти компоненты. Эту цветовую модель лучше применять в тех графических редакторах, в которых изображения создают сами, а не обрабатывают уже готовые. Затем созданное свое произведение можно преобразовать в цветовую модель RGB, если ее планируется использовать в качестве экранной иллюстрации, или CMYK, если в качестве печатной, Значение цвета выбирается как вектор, выходящий из центра окружности. Направление вектора задается в угловых градусах и определяет цветовой оттенок. Насыщенность цвета определяется длиной вектора, а яркость цвета задается на отдельной оси, нулевая точка которой имеет черный цвет. Точка в центре соответствует белому (нейтральному) цвету, а точки по периметру - чистым цветам.

   2) Принцип метода RGB заключается в следующем: известно, что любой цвет можно представить в виде комбинации трех цветов: красного (Red, R), зеленого (Green, G), синего (Blue, B). Другие цвета и их оттенки получаются за счет наличия или отсутствия этих составляющих.По первым буквам основных цветов система и получила свое название - RGB. Данная цветовая модель является аддитивной, то есть любой цвет можно получить сочетание основных цветов в различных пропорциях. При наложении одного компонента основного цвета на другой яркость суммарного излучения увеличивается.

Если совместить все три компоненты, то получим ахроматический серый цвет, при увеличении яркости которого происходит приближение к белому цвету. При 256 градациях тона (каждая точка кодируется 3 байтами) минимальные значения RGB (0,0,0) соответствуют черному цвету, а белому - максимальные с координатами (255, 255, 255). Чем больше значение байта цветовой составляющей, тем этот цвет ярче. Например, темно-синий кодируется тремя байтами ( 0, 0, 128), а ярко-синий (0, 0, 255).

   3) Принцип метода CMYK. Эта цветовая модель используется при подготовке публикаций к печати. Каждому из основных цветов ставится в соответствие дополнительный цвет (дополняющий основной до белого). Получают дополнительный цвет за счет суммирования пары остальных основных цветов. Значит, дополнительными цветами для красного является голубой (Cyan,C) = зеленый + синий = белый - красный, для зеленого - пурпурный (Magenta, M) = красный + синий = белый - зеленый, для синего - желтый (Yellow, Y) = красный + зеленый = белый - синий. Причем принцип декомпозиции произвольного цвета на составляющие можно применять как для основных, так и для дополнительных, то есть любой цвет можно представить или в виде суммы красной, зеленой, синей составляющей или же в виде суммы голубой, пурупурной, желтой составляющей. В основном такой метод принят в полиграфии. Но там еще используют черный цвет (BlacК, так как буква В уже занята синим цветом, то обозначают буквой K). Это связано с тем, что наложение друг на друга дополнительных цветов не дает чистого черного цвета.

Различают несколько режимов представления цветной графики:  
 а) полноцветный (True Color);  
 б) High Color;  
          в) индексный.

При полноцветном режиме для кодирования яркости каждой из составляющих используют по 256 значений (восемь двоичных разрядов), то есть на кодирование цвета одного пикселя (в системе RGB) надо затратить 8\*3=24 разряда. Это позволяет однозначно определять 16,5 млн цветов. Это довольно близко к чувствительности человеческого глаза. При кодировании с помощью системы CMYK для представления цветной графики надо иметь 8\*4=32 двоичных разряда.

Режим High Color - это кодирование при помощи 16-разрядных двоичных чисел, то есть уменьшается количестко двоичных разрядов при кодировании каждой точки. Но при этом значительно уменьшается диапазон кодируемых цветов.

При индексном кодировании цвета можно передать всго лишь 256 цветовых оттенков. Каждый цвет кодируется при помощи восьми бит данных. Но так как 256 значений не передают весь диапазон цветов, доступный человеческому глазу, то подразумевается, что к графическим данным прилагается палитра (справочная таблица), без которой воспроизведение будет неадекватным: море может получиться красным, а листья - синими. Сам код точки растра в данном случае означает не сам по себе цвет, а только его номер (индекс) в палитре. Отсюда и название режима - индексный.

Соответствие между количеством отображаемых цветов (К) и количеством бит для их кодировки (а) находиться по формуле: К = 2а.

Двоичный код изображения, выводимого на экран, хранится в видеопамяти. Видеопамять - это электронное энергозависимое запоминающее устройство. Размер видеопамяти зависит от разрешающей способности дисплея и количества цветов. Но ее минимальный объем определяется так, чтобы поместился один кадр (одна страница) изображения, т.е. как результат произведения разрешающей способности на размер кода пикселя.

Vmin = M \* N \* a.

Шестнадцатицветная палитра позволяет увеличить количество используемых цветов. Здесь будет использоваться 4-разрядная кодировка пикселя: 3 бита основных цветов + 1 бит интенсивности. Последний управляет яркостью трех базовых цветов одновременно (интенсивностью трех электронных пучков).

При раздельном управлении интенсивностью основных цветов количество получаемых цветов увеличивается. Так для получения палитры при глубине цвета в 24 бита на каждый цвет выделяется по 8 бит, то есть возможны 256 уровней интенсивности (К = 28).

2.5 Обзор принципов действия, структурных решений и микропроцессорной базы современных тепловизоров

Любой объект излучает электромагнитные волны в очень широком диапазоне частот, в том числе и волны в инфракрасном спектре, так называемое «тепловое излучение». При этом интенсивность теплового излучения напрямую зависит от температуры объекта, и лишь в очень малой степени зависит от условий освещенности в видимом диапазоне. Таким образом, при помощи тепловизионного прибора о любом наблюдаемом объекте может быть собрана и визуализирована дополнительная информация, недоступная человеческому глазу и приборам, Тепловизор – устройство, позволяющее визуализировать картину теплового излучения наблюдаемого объекта. Это открывает ряд уникальных возможностей для разных сфер деятельности: точных измерений, контроля технологических процессов, и конечно – обеспечения безопасности.

Принцип действия современных тепловизоров основан на способности некоторых материалов фиксировать излучение в инфракрасном диапазоне. Посредством оптического прибора, в состав которого входят линзы, изготовленные с применением редких материалов, прозрачных для инфракрасного излучения (таких как германий), тепловое излучение объектов проецируется на матрицу датчиков, чувствительных к инфракрасному излучению. Далее сложные микросхемы считывают информацию с этих датчиков, и генерируют видеосигнал, где разной температуре наблюдаемого объекта соответствует разный цвет изображения. Шкала соответствия цвета точки на изображении к абсолютной температуре наблюдаемого объекта может быть выведена поверх кадра. Также возможно указание температур наиболее горячей и наиболее холодной точки на изображении. В зависимости от модели тепловизоры различаются по величине шага измеряемой температуры. Современные технологии позволяют различать температуру объектов с точностью до 0,05-0,1 К.

Многие тепловизионные приборы также оснащены устройствами памяти для записи полученного видеоизображения картины теплового излучения, производительными микропроцессорами, позволяющими осуществлять в режиме реального времени минимальную аналитику полученного в результате сканирования изображения инфракрасного излучения. Довольно часто используется конфигурация совместного использования тепловизора и видеокамеры, что позволяет в общем случае получить изображение объекта в «расширенном» диапазоне объединенных инфракрасного и видимого спектров, а в неблагоприятных условиях (например - отсутствие освещения объекта) наблюдать объект хотя бы в одном из диапазонов. ИК или видимый диапазон могут как накладываться друг на друга, так и транслироваться отдельно. Специальное программное обеспечение позволяет настроить работу тепловизионного комплекса, максимально эффективно скоординировав работу всех входящих в него устройств.

Точность изображения и другие характеристики тепловизора обычно определяются сферой его использования. В научных лабораториях используются более сложные конструкции, имеющие за счет узкой специализации наименьший шаг измеряемой температуры. Для обеспечения безопасности на различных объектах используются модели, фиксирующие тепловое излучение с чуть меньшей точностью, однако работающие на более широком диапазоне частот и с более чем достаточной для эффективного выполнения своих функций точностью. В любом случае, принцип действия тепловизора – измерение и визуализация теплового излучения – востребован во всех сферах жизни современного общества.

2.6 Структура микроконтроллерного ядра *ARM* *Cortex*-*M4*

Вариант микроконтроллерного ядра *Cortex*-*M*4, по сравнению с *Cortex*-*M*3, не характеризуется ростом общих показателей. Фактически *M*4 тот же самый *M*3, но дополнительно оснащенный *DSP*-инструкциями. Наличие последних существенно ускоряет обработку потоковых данных, что в свою очередь делает *M*4 весьма привлекательным для использования в системах управления и обработки информации [9].

Возможности *DSP*, входящего в состав *M*4, позволяют параллельно выполнять четыре операции сложения/вычитания для восьмиразрядных чисел или две операции сложения/вычитания с шестнадцати разрядными операндами. Также реализовано умножение за один цикл, при этом для шестнадцати разрядных чисел возможно параллельное исполнение двух операций.

В отличие от профиля А, *Cortex*-*M* развивается не столь бурно. Можно только предположить, что развитие пойдет по пути «больших» систем и в скором времени привычными станут двух-, трех или четырехядерные контроллеры.

2.7 Регистровая модель интерфейса *GPIO* *ARM* *Cortex*-*M*4

С помощью портов ввода/вывода общего назначения *GPIO* (*General*-*Purpose* *I*/*O* *port*) можно осуществить обмен цифровыми сигналами со внешними по отношению к микроконтроллеру устройствами на низком уровне. Виды входа:

– *floating* («плавающий», т.е. без подтягивающих резисторов, с высоким входным сопротивлением);

– *pull*-*up* (с внутренним, подтягивающим к высокому уровню резистором);

– *pull*-*down* (с подтягивающим к низкому уровню резистором);

– *analog* (аналоговый вход).

Выход может быть: *push*-*pull* (двухтактный), *open*-*drain* (с открытым стоком), *alternate* *function* *push*-*pull* (двухтактный для альтернативной функции), *alternate* *function* *open*-*drain* (с открытым стоком для альтернативной функции).

В связи с особенностью интерфейса во время и после сброса все порты конфигурируются как *floating* *input*.

2.8 Принцип функционирования вложенных векторов  
 прерывания и детектора внешних событий микроконтроллера  
 семейства *ARM* *Cortex*-*M*4

*Event* называют аппаратное или программное событие, на которое могут реагировать ядро или периферийные блоки. В качестве ответной реакции система может начать прерывание [10].

Прерыванием работы программы (*Interrupt*) называют переход управления в специализированный участок – обработчик прерывания. *Event* и *Interrupt* взаимосвязаны таким образом: каждый *Interrupt* вызывается *Event*, но не каждый *Event* вызывает *Interrupt*.

Контроллер вложенных векторизованных прерываний (КВВП) является стандартным блоком ядра *Cortex*. Это означает, что у любого *Cortex*-микроконтроллера будет присутствовать одна и та же структура прерываний, независимо от его производителя. Таким образом, прикладной код и операционные системы можно легко портировать с одного МК на любой другой и, при этом, программисту не потребуется изучение нового набора регистров. При разработке КВВП также учитывалось, что задержка реагирования на прерывание должна быть очень малой.

Управление и обработка прерываниями производится контроллером приоритетных векторных прерываний *NVIC*. При возникновении некоторого события контроллер прерываний автоматически прерывает выполнение основной программы и вызывает соответствующую функцию обработки прерываний. После выхода из функции обработчика прерываний программа продолжает выполнение с того места, где произошло прерывание.

**Вход в прерывание и выход из него**. При инициации прерывания *NVIC* переключает ядро в режим обработки прерывания. После перехода в режим обработки прерывания регистры ядра помещаются в стек. Непосредственно во время записи значения регистров в стек осуществляется выборка начального адреса функции обработки прерывания.

В стек перемещается регистр статуса программы (*Program* *Status* *Register* (*PSR*)), счетчик программы (*Program* *Counter* (*PC*)) и регистр связи (*Link* *Register* (*LR*)). Описание регистров ядра приведено в *Cortex*-*M*4 *Generic* *User* *Guide*. Благодаря этому запоминается состояние, в котором находилось ядро перед переходом в режим обработки прерываний.

Также сохраняются регистры *R*0 – *R*3 и *R*12. Эти регистры используются в инструкциях для передачи параметров, поэтому, помещение в стек делает возможным их использование в функции обработки прерывания, а *R*12 часто выступает в роли рабочего регистра программы.

По завершении обработки прерывания все действия выполняются в обратном порядке: извлекается содержимое стека и, параллельно с этим, осуществляется выборка адреса возврата.

С момента инициации прерывания до выполнения первой команды обработчика прерываний проходит 12 тактов, такое же время необходимо для возобновления основной программы после завершения обработки прерывания.

1. Приостановка низкоприоритетного прерывания – в этой ситуации обработка низкоприоритетного прерывания прекращается. Следующие 12 циклов выполняется сохранение в стек нового набора данных и запускается обработка высокоприоритетного прерывания. После его обработки содержимое стека автоматически извлекается и возобновляется обработка низкоприоритетного прерывания.
2. Непрерывная обработка прерываний – эта ситуация может возникнуть в двух случаях: если два прерывания имеют одинаковый приоритет и возникают одновременно, если низкоприоритетное прерывание возникает во время обработки высокоприоритетного.

В этом случае, промежуточные операции над стеком не производятся. Происходит только загрузка адреса обработчика низкоприоритетного прерывания и переход к его выполнению. Отказ от операций над стеком экономит 6 тактов. Переход к следующему прерыванию происходит не за 12 тактов, а всего за 6.

1. Запаздывание высокоприоритетного прерывания – ситуация возникает, если высокоприоритетное прерывание происходит во время перехода к обработке низкоприоритетного (за те самые 12 тактов). В этом случае переход к высокоприоритетному прерыванию будет происходить не менее 6 тактов с момента его возникновения (время необходимое для загрузки адреса обработчика прерывания и перехода к нему). Возврат в низкоприоритетное уже описан выше.

Все возможные прерывания, поддерживаемые *NVIC*, записываются в таблицу векторов прерываний.

2.9 Структура и логика функционирования цифрового болометра *LEPTON* *FLIR*

Lepton Flir имеет LWIR-сенсор, который умеет регистрировать инфракрасные волны в диапазоне 8–14 мкм. Именно на этот диапазон приходится максимум излучения тел с температурой от –50 до +50 градусов Цельсия. (Но это не означает, что более горячие тела, например, чайник, не будут видны в этом тепловизоре).  
 Датчик миниатюрный, обеспечивает тепловое разрешение 80 х 60 пикселей с частотой обновления в 9 Гц. Интерфейс получения данных — Video over SPI, а управления — I2C. Болометр представляет собой приемник лучистой энергии, действие которого основано на изменении электропроводности чувствительного элемента при нагревании его вследствие поглощения излучения.

Более широкое распространение по сравнению со сверхпроводящими получили полупроводниковые охлаждаемые и неохлаждаемые болометры. Для их изготовления используются материалы, температурный коэффициент сопротивления которых отрицателен и на порядок выше, чем у металлов. Для большинства полупроводников bT = -3×103 /T 2 . Чувствительные элементы изготовляются на основе окисей никеля, кобальта, марганца, а также из пленок германия и кремния. В смесь порошкообразных окислов добавляют органическое связывающее вещество, после чего наносят пасту на стеклянную подложку или прессуют в виде таблеток или столбиков. В том и другом случае приводят термическую обработку при высокой температуре. Поверхность чувствительного слоя чернят для увеличения коэффициента поглощения, а сам слой закрепляют на цоколе из стекла или кварца и помещают в герметический корпус. Сопротивление полупроводниковых болометров обычно составляет от 1 до 10 МОм, а инерционность от 250 мкс до 40 мс. Пороговая чувствительность болометров обычно лежит в интервале 10-9…10-11 Вт. Для повышения чувствительности болометров применяют иммерсионные линзы. В иммерсионном полупроводниковом болометре чувствительный элемент расположен в фокальной области сферической иммерсионной линзы с высоким показателем преломления (германий). Иммерсионная линза позволяет в 3…4 раза улучшить пороговую чувствительность за счет уменьшения приемной площадки болометра при сохранении заданного поля зрения, или получить ту же чувствительность, что и у обычного болометра, используя в оптической системе приемный объектив вдвое меньшего диаметра.

Охлаждение полупроводниковых болометров повышает их обнаружительную способность. Применение интегрирующей полости позволяет добиться почти полного поглощения принимаемого излучения. Тепловой контакт чувствительного элемента с хладагентом обычно осуществляется при монтаже последнего на какой-нибудь детали конструкции, имеющей хороший тепловой контакт с гелиевой ванной. Теплопроводность при этом определяет как постоянную времени приемника, так и компоненту шума, связанную с температурными флуктуациями. Наиболее существенные источники шумов охлаждаемых болометров связаны с фоновым излучением, попадающим в апертуру приемника, и шумами предусилителя.

2.10 Физический и канальный уровни интерфейсов *FSMC*, *I2C* и *SPI*

*SPI* – это синхронная четырёхпроводная шина, предназначенная для обеспечения высокоскоростного сопряжения микроконтроллеров и периферии. Она представляет собой соединение двух синхронных сдвиговых регистров, которые является центральным элементом любого *SPI* устройства. Для соединения используется конфигурация, ведущий/ведомый. Только ведущий может генерировать импульсы синхронизации. В схеме всегда только один ведущий (в отличие от той же шины *I*2*C*, где возможен вариант с более чем одним ведущим), количество ведомых может быть различно [12].

Главным составным блоком интерфейса *SPI* является обычный сдвиговый регистр, сигналы синхронизации и ввода/вывода битового потока которого и образуют интерфейсные сигналы. Два проводника используются для передачи данных, один для подачи тактовых импульсов и по одному сигналу выбора ведомого для каждого из ведомых. Описание используемых сигналов:

*MOSI* – *Master* *Output*, *Slave* *Input* (выход ведущего, вход ведомого). Данный сигнал предназначен для последовательной передачи данных от ведущего к ведомому. Также может называться *SDO*, *DO* и т.п.

*MISO* – *Master* *Input*, *Slave* *Output* (вход ведущего, выход ведомого). Данный сигнал предназначен для последовательной передачи данных от ведомого к ведущему. Может называться *SDI*, *DI* и т.п.

*SCK* – *Serial* *Clock* (сигнал синхронизации). Используется для синхронизации при передаче данных. Также может иметь название *SCLK*, *CLK* и др.

*CS* – *Chip* *Select* (выбор микросхемы). С помощью данного сигнала происходит активация ведомого устройства. Обычно он является инверсным, то есть низкий уровень считается активным.

Конкретные имена портов интерфейса *SPI* могут различаться в зависимости от производителя аппаратных средств, при этом возможны следующие варианты:

* *MOSI*: *SOMI*, *SDO* (на устройстве), *DO*, *DON*, *SO*, *MRSR*;
* *MISO*: *SIMO*, *SDI* (на устройстве), *DI*, *DIN*, *SI*, *MTST*;
* *SCLK*: *SCK*, *CLK*;
* *SS*: *nCS*, *CS*, *CSB*, *CSN*, *nSS*, *STE*, *SYNC*.

В отличие от *SPI, I2C* использует только два провода для всего процесса, возможно, поэтому он также известен как протокол двухпроводного интерфейса *(TWI)*. Эти два провода представляют собой *SDA* (последовательные данные) и *SCL* (последовательное тактирование). Протокол *I2C* может поддерживать несколько подчиненных устройств, но в отличие от *SPI*, который поддерживает только одно ведущее устройство, *I2C* может также поддерживать несколько мастер-устройств. Каждое устройство отправляет / принимает данные, используя только один провод, который является *SDA*. *SCL* поддерживает синхронизацию между устройствами через общую систему тактирования, которая реализуется активным ведущим устройством.

*I2C*

Каждое подчиненное устройство имеет свой собственный уникальный адрес (от 7 до 10 бит), который ведущий использует для их идентификации. Всякий раз, когда ведущий хочет отправить данные, он сначала генерирует запрос, который имеет конкретный адрес этого подчиненного устройства. Каждое подчиненное устройство сопоставляет этот адрес со своим и тот, чей адрес соответствует, отвечает ведущему. Каждое сообщение начинается с условия запуска и заканчивается условием остановки. Одно сообщение может содержать несколько байтов данных, каждый из которых имеет бит подтверждения *(ACK)* или отрицательный бит подтверждения *(NACK)* между ними.

Преимущества *I2C* следующие. Во-первых, несколько ведущих и несколько ведомых могут соединяться друг с другом. Во-вторых, для этого протокола требуется только два провода. К недостаткам протокола *I2C* можно отнести то, что он медленнее по сравнению с *SPI*, потому что в рамках этого протокола выполняется много операций с кадрами данных.

FSMC - это параллельный интерфейс, позволяющий подключать к МК дополнительную память в область памяти самого МК. Эта шина напоминает интерфейс 8080.

**SetupTime**- определяет сколько тактов HSE прошло между тем, как CE стало 0 и тем как WE так же стало 0.  
 **WaitSetupTime**- Время, которое WE находится в 0.  
 **HoldSetupTime**- Время, которое продолжают идти адресные данные после установки WE в 1. (В расчётах данного параметра можно опираться на время установки СЕ)  
 **HiZSetupTime**- Время в тактах, которое проходит между тем как CE стало 0 и началом передачи команды.

2.11 Принципы функционирования блока *DMA* прямого доступа   
 к памяти

*DMA*, или *Direct* *Memory* *Access* – способ быстродействующего подключения внешнего устройства, при котором обращение к памяти происходит, минуя процессор. Такой обмен происходит под управлением контроллера прямого доступа к памяти (КПДП).

Контроллер *DMA* может получать доступ к системной шине независимо от ЦП и имеет несколько регистров. Регистры контроллера *DMA* доступны ЦП для чтения и записи и используются для задания:

– номера порта, который должен быть использован для передачи данных;

– вида операции (чтение или запись);

– единицы переноса (побайтно или пословно);

– размера данных, которые следует перенести, в байтах;

*EasyDMA* – это простой в использовании модуль прямого доступа к памяти, который некоторые периферийные устройства используют для получения прямого доступа к ОЗУ данных. Модуль изображен на рисунке 2.6.

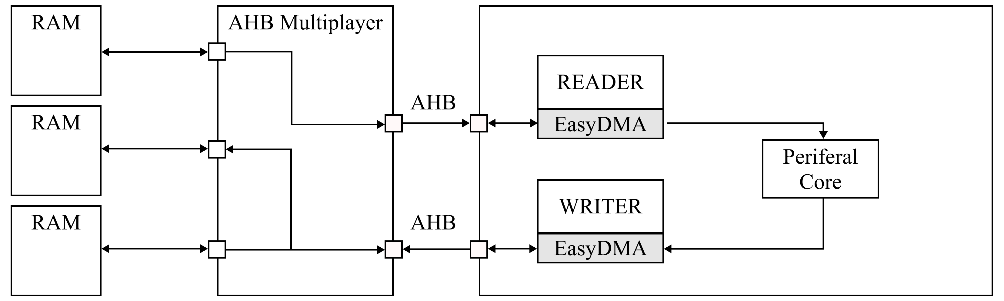


Рисунок 2.6 – *EasyDMA*

*EasyDMA*, как и все *DMA*, позволяет разгрузить ЦП микроконтроллера.

2.12 Структура и логика функционирования дисплейного модуля *HY32D* на базе видеопроцессора *ILI9341*

*LI9341* - это однокристальный драйвер *SOC* с 262 144 цветами для жидкокристаллического дисплея *TFT* с разрешением *240RGBx320* точки, содержащие 720-канальный драйвер источника, 320-канальный драйвер шлюза, 172 800 байт *GRAM* для графики отображать данные размером *240RGBx320* точек и цепи питания.

*ILI9341* поддерживает параллельный 8- / 9- / 16- / 18-битный интерфейс *MCU* шины данных, 6- / 16- / 18-битный интерфейс *RGB* шины данных и 3- / 4-строчный последовательный периферийный интерфейс (*SPI*). Область движущегося изображения может быть указана во внутренней *GRAM* окном адресная функция. Указанная область окна может быть обновлена ​​выборочно, так что движущееся изображение может быть отображается одновременно независимо от области неподвижного изображения.

*ILI9341* может работать при напряжении интерфейса ввода-вывода 1,65 В ~ 3,3 В и встроенной цепи повторителя напряжения для генерирования уровня напряжения для управления ЖК-дисплеем. *ILI9341* поддерживает полноцветный, 8-цветный режим отображения и режим ожидания для точного управлением питанием с помощью программного обеспечения и эти функции делают *ILI9341* идеальным драйвером ЖК-дисплея для средних и малых портативных устройств, такие как цифровые сотовые телефоны, смартфоны, *MP3* и *PMP*, где длительное время автономной работы это серьезная проблема.

3 Разработка структурной электрической

**СХЕМЫ МОБИЛЬНОГО ТЕПЛОВИЗОРА**

3.1 Обоснование базовых блоков структурной схемы мобильного тепловизора

Структурная схема разрабатывается на начальных стадиях проектирования и предшествует разработке схем других типов. Структурная схема определяет основные функциональные части изделия, их назначение и взаимосвязи между ними. Схема отображает принцип действия изделия в самом общем виде [14].

Действительное расположение составных частей на структурной схеме не учитывают и способ связи не раскрывают. Построение схемы должно давать наглядное представление о составе изделия и последовательности взаимодействия функциональных частей в изделии.

Проектируя структурную схему данного курсового проекта было решено выделить следующие функциональные части:

* порт последовательный универсальный;
* микроконтроллер;
* интерфейс последовательный;
* регистр контрольный;
* процессор центральный;
* стабилизатор напряжения;
* генератор частоты;
* контроль питания.

Порт последовательный универсальный служит для подключения к ведущему устройству и питания устройства.

Микроконтроллер (МК) сочетает на одном кристалле функции микропроцессора и периферийных устройств, содержит ОЗУ и/или ПЗУ.

Интерфейс последовательный служит для сообщения между периферийными системами МК и самим микропроцессором.

Регистр контрольный образует ОЗУ и служит для хранения служебных данных внутренних операций контроллера.

Процессор центральный выполняет непосредственные алгоритмы и вычисления, предусмотренные прошивкой МК.

Стабилизатор напряжения защищает электрорадиоэлементы от нестабильного электроснабжения, сглаживает перепады и скачки отклонения питающего напряжения в сети электропитания (стабилизация напряжения питания).

Генератор частоты служит для получения управляющего сигнала особой формы и размера, необходимой для кодировки информации и последующей работы с данными.

Контроль питания проверяет, нет ли обрывов в подключении, а также правильность и корректность самого подключения. Питание начинается только после разрешающего сигнала от контроллера.

3.2 Обоснование связей структурной схемы мобильного тепловизора

Структурная схема определяет основные функциональные части изделия, их назначение и взаимосвязи между ними. Схема отображает принцип действия изделия в самом общем виде.

Действительное расположение составных частей на структурной схеме не учитывают и способ связи не раскрывают. Построение схемы должно давать наглядное представление о

* составе изделия;
* последовательности взаимодействия функциональных частей  
   в изделии.

Направление хода процесса, происходящего в изделии, обозначают стрелками, соединяющими функциональные части. На схемах простых изделии функциональные части располагают в виде цепочки в соответствии с ходом рабочего процесса в направлении слева направо.

При подключении устройства через порт последовательный универсальный к компьютеру начинается обмен данных. Через универсальный последовательный порт осуществляются прием и передача информации, представленной в последовательном коде (младшими битами вперед). Наличие буферного регистра приемника позволяет совмещать операцию чтения ранее принятого байта с приемом очередного [15].

Через последовательный интерфейс происходит обмен микроконтроллера *DD*1 с оптическим датчиком на микроконтроллере *DD*2 и осуществляется дальнейшая обработка данных.

Контрольный регистр хранит в себе информацию с оптического датчика и при необходимости передаёт её на последовательный интерфейс.

Центральный процессор питается напряжением от стабилизатора напряжения, тактируется от заданной тактовой частоты генератора.

Тактовая частота МК – это количество тактов за секунду, выполняемых контроллером. Чем она выше, тем большее количество операций он может выполнить [16].

Так как в микроконтроллер *DD*2 встроен оптический датчик, то центральный процессор обрабатывает данные с него с заданной частотой и посылает данные на контрольный регистр.

Блок контроля питания. При работе с данным блоком есть возможность при необходимости произвести остановку устройства или перезагрузить работу микроконтроллера.

После того, как были проанализированы основные понятия и после детального изучения электрической принципиальной схемы была составлена структурная схема мобильного тепловизора.

4 РАЗРАБОТКА ПРИНЦИПИАЛЬНОЙ

ЭЛЕКТРИЧЕСКОЙ СХЕМЫ МОБИЛЬНОГО ТЕПЛОВИЗОРА

4.1 Обоснование выбора САПР для разработки принципиальной  
 электрической схемы

Чертежи электрических схем являются технической документацией, которая используется для изображения и описания проектов электрических систем. В чертежах электрических схем для компонентов и соединений любых типов используются собственные специальные обозначения, и каждая деталь играет важную роль [17].

Для разработки принципиальной электрической схемы разрабатываемого мобильного тепловизора была выбрана одна из самых популярных графических систем автоматизированного проектирования – *AutoCAD*. Несмотря на большое количество команд, *AutoCAD* обладает удобным для пользователя интерфейсом.

*AutoCAD* представляет собой систему, позволяющую автоматизировать чертежно-графические работы. В графическом пакете *AutoCAD* есть все, что необходимо конструктору для создания чертежа [18].

Инструментам ручного черчения в автоматизированной среде соответствуют графические примитивы (точка, отрезок, окружность и др.), команды их редактирования (стирание, перенос, копирование и т. п.), команды установки свойств примитива (задание толщины, типа и цвета графических объектов). Для выбора листа нужного формата и масштаба чертежа в системе есть соответствующие команды настройки чертежа. Для нанесения размера конструктору необходимо лишь задать место его расположения на чертеже. Размерная и выносная линии, а также стрелки и надписи выполняются автоматически, а в последних версиях *AutoCAD* есть режим полной автоматизации простановки размеров.

4.2 Описание используемых библиотечных элементов и процесса  
 их создания

На принципиальной схеме изображают все электрические элементы или устройства, необходимые для осуществления и контроля в изделии установленных электрических процессов, все электрические взаимосвязи между ними, а также электрические элементы (соединители, зажимы и т.д.), которыми заканчиваются входные и выходные цепи.

Согласно ГОСТ 2.702-75 порядковые номера элементам (устройствам) следует присваивать, начиная с единицы, в пределах группы элементов (устройств), которым на схеме присвоено одинаковое буквенное позиционное обозначение.

Все элементы были созданы в *AutoCAD* в соответствии с ГОСТ 2.701-84.

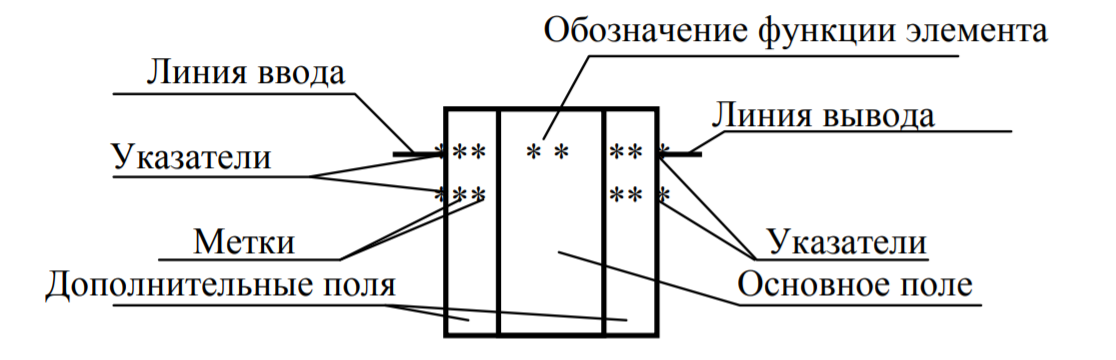


Рисунок 3.5 – Условно-графическое обозначение элемента цифровой техники (в общем виде)

УГО может состоять только из основного поля или из основного поля и одного дополнительного, которое располагают справа или слева от основного, а также из основного поля и двух дополнительных. Допускается дополнительные поля разделять на зоны, которые отделяют горизонтальной чертой.

Основное и дополнительные поля могут быть не отделены линией. При этом расстояние между буквенными, цифровыми или буквенно-цифровыми обозначениями, помещенными в основное и дополнительные поля, определяется однозначностью понимания каждого обозначения, а для обозначений, помещенных на одной строке, должно быть не менее двух букв (цифр, знаков), которыми выполнены эти обозначения.

Размеры УГО определяют по высоте:

* число линий выводов,
* число интервалов, число строк информации в основном и дополнительных полях, размером шрифта;

По ширине: наличием дополнительных полей и числом знаков, помещаемых в одной строке внутри УГО (с учетом пробелов), размером шрифта.

4.3 Обоснование выбора базовых компонентов принципиальной  
 схемы мобильного тепловизора

Все электронные компоненты, которые применяются в каких-либо устройствах, чаще всего изготавливаются в заводских условиях на основе определенных стандартов и технических условий, а также обладают законченной формой и определенным видом [18].

К таким компонентам относятся:

* резисторы;
* диоды;
* конденсаторы;
* источники тока;
* транзисторы;
* трансформаторы и другие.

Базовые компоненты используются для обеспечения стабильной работы принципиальной схемы. Благодаря им устраняются перепады напряжения, сетевые помехи, шумы и влияние факторов внешней среды.

4.4 Обоснование связей принципиальной электрической схемы  
 мобильного тепловизора

Для соединения компонентов принципиальной электрической схемы, разрабатываемого устройства, были соединительные провода – проводники. Взаимосвязи между компонентами на схеме изображают в виде УГО, установленных в стандартах ЕСКД.

На электрической принципиальной схеме показываются все электрические связи между входящими в нее элементами электрооборудования производственного механизма. Принципиальная схема проектируется с использованием существующих типовых узлов и схем автоматического управления электропроводами.

1. РАЗРАБОТКА АЛГОРИТМА ФУНКЦИОНИРОВАНИЯ

**МОБИЛЬНОГО ТЕПЛОВИЗОРА**

Исходя из информации, рассмотренной в пункте 2.13, устройство должно быть запрограммировано на цифровую обработку изображения, полученного с оптического сенсора, дальнейшее её преобразование и сравнение с имеющейся в буфере информацией..

Таким образом, разработаем диаграмму состояний и схему алгоритма.

* 1. Разработка диаграммы состояний устройства

Основываясь на задумке, можно предположить, что устройство будет постоянно передавайть данные камерына дисплей.

В состоянии обнаружения изменений данных, полученных с оптического сенсора, устройство будет захватывать картинку,

* 1. Разработка схемы алгоритма устройства

Прежде всего будет происходить инициализация всех компонентов. Если инициализация прошла успешно, устройство начнет корректную работу.

Встроенный оптический датчик включиться, и если обнаружит, то передаст информацию микроконтроллеру, который в свою очередь обработает полученный сигнал и обеспечит передачу данных на подключенные устройства, чтобы вывести необходимую информацию на экран ведущего устройства, будь то ПК или смартфон.

Диаграмма состояний и блок-схема алгоритма находятся среди графического материала в формате А3.

1. Разработка конструкции проектируемого

**ПРИБОРА**

* 1. Выбор и обоснование элементной базы

Критерием выбора электрорадиоэлементов (ЭРЭ) в любом радиоэлектронном устройстве является соответствие технологических и эксплуатационных характеристик ЭРЭ заданным условиям работы и условиям эксплуатации. Основными параметрами при выборе ЭРЭ являются технические и эксплуатационные параметры. Технические:

* номинальное значение параметров ЭРЭ согласно принципиальной электрической схеме;
* допустимые отклонения величин ЭРЭ от их номинальных значений;
* допустимые рабочие напряжения ЭРЭ;
* допустимые рассеиваемые мощности ЭРЭ;
* диапазон рабочих частот ЭРЭ;
* коэффициент электрической нагрузки ЭРЭ.

Эксплуатационные параметры:

* диапазон рабочих температур;
* относительная влажность воздуха;
* давление окружающей среды; вибрационные нагрузки и другие (специальные) показатели.

Дополнительными критериями при выборе ЭРЭ являются:

* унификация ЭРЭ; масса и габариты ЭРЭ;
* наименьшая стоимость;
* надежность.

Выбор элементной базы по вышеназванным критериям позволяет обеспечить надежную работу изделия. Применение принципов стандартизации и унификации при выборе ЭРЭ, а также при конструировании изделия в целом позволяет получить следующие преимущества:

* значительно сократить сроки и стоимость проектирования;
* сократить на предприятии–изготовителе номенклатуру применяемых деталей и сборочных единиц;
* увеличить применяемость и масштаб производства;
* исключить разработку специальной оснастки и специального оборудования для каждого нового варианта РЭА, т. е. упростить подготовку производства;
* улучшить производственную и эксплуатационную технологичность;
* снизить себестоимость выпускаемого изделия [20].

Учитывая вышесказанное, перейдем к выбору элементной базы конструкции проектируемого прибора. Сравнительный анализ по использованию элементной базы в данной системе согласно предложенной схемы электрической принципиальной показал соответствие эксплуатационных и технических характеристик ЭРЭ заданным условиям эксплуатации. Этими элементами являются:

6.2. Выбор и обоснование конструктивных элементов  
 и установочных изделий

Для изготовления печатных плат в РЭА наиболее широкое распространение получили такие материалы, как гетенакс, стеклотекстолит. При выборе материала печатной платы необходимо иметь в виду следующее: материал печатной платы должен иметь высокие электроизоляционные показатели в заданных условиях эксплуатации РЭА (т. е. иметь большую электрическую прочность, малые диэлектрические потери, обладать химической стойкостью к действию химических растворов, используемых при изготовлении печатных плат, допускать штамповку, выдерживать кратковременные воздействия температуры до 240С в процессе пайки на плате ЭРЭ, иметь высокую влагостойкость, быть дешевым). При выборе материала печатной платы необходимо руководствоваться документами: ГОСТ 23751-86 и ГОСТ 23752-86 [21].

Учитывая вышеизложенные требования, в разрабатываемом устройстве применены платы из стеклотекстолита марки СТЭФ 35/35мкм 1,5 мм с покрытием сплавом «Розе».

При монтаже элементов было принято решение об использовании одностороннего смешанного метода монтажа. Этот метод был выбран из-за сочетания преимуществ поверхностного и штыревого монтажа.

7 Расчёт конструктивно-технологических параметров проектируемого прибора

7.1 Проектирование печатного модуля

Для проектирования разрабатываемого устройства выбрана односторонняя печатная плата. Данный тип печатной платы выбран из-за того, что все элементы могут разместиться на одной стороне.

Класс точности изготовленного печатного модуля – 3-й, шаг координатной сетки – 0,5 мм. Изготовление печатных плат определенного класса точности (ГОСТ 23751-86) обеспечивают, применяя различные техническое оснащение и вспомогательные материалы. Печатные платы 1 и 2 классов точности наиболее просты в исполнении, надежны в эксплуатации и имеют минимальную стоимость; 3 класса – требуют использования высококачественных материалов, более точного инструмента и оборудования; 4 и 5 классов – специальных материалов, прецизионного оборудования, особых условий для изготовления.

7.2 Выбор и обоснование материалов конструкции и защитных  
 покрытий, маркировки деталей и сборочных единиц

При выборе материалов необходимо учитывать, что они должны обладать следующими свойствами:

* иметь малую стоимость;
* легко обрабатываться;
* быть легкими;
* обладать достаточной прочностью и жесткостью;
* внешний вид материалов лицевой и задней панели должен отвечать требованиям технической эстетики;
* сохранять свои физико–химические свойства в процессе эксплуатации.

Применение унифицированных материалов в конструкции, ограничение номенклатуры применяемых деталей позволяет уменьшить себестоимость разрабатываемого изделия, улучшить производственную и эксплуатационную технологичность. Сохранение физико-химических свойств материалов в процессе их эксплуатации достигается выбором для них необходимых покрытий. При выборе покрытий необходимо руководствоваться рекомендациями и требованиями ГОСТ 9303-84 и ОСТ 4ГО.014.000.

Для изготовления печатных плат применяют различные материалы. Материал печатной платы должен обладать высокой механической прочностью, хорошими электроизоляционными свойствами, иметь высокую нагревостойкость, а также иметь высокую степень адгезии печатных проводников.

Исходя из требования механической прочности, как упоминалось ранее в подпункте 6.2, в качестве материала печатной платы выбран фольгированный стеклотекстолит марки «СТЭФ» 35/35 1,5 мм.

В качестве паяльной пасты выбрана *Hard* *Gold*. Это безотмывочная паста, состоящая из *RMA*-флюса и припойного порошка, частицы которого имеют строго сферическую форму. Применяемый флюс высокотехнологичен, не требует отмывки. Не содержит галогенов. Паста обеспечивает пайку элементов, изготовленных по бессвинцовым технологиям [22].

Для пайки выводных элементов выбран припой ПОС-61. Он относится к типу, который создается на основе специального материала (свинец 39-50% + олово 50-61 %).

8 Применение средств автоматизированного проектирования при разработке прибора

В данном курсовом проекте были использованы следующие программные средства: в *Altium* *Designer* для проектирования принципиальной схемы, печатной платы и её визуализации, необходимая документация и структурная схема была разработана в САПР *AutoCAD*. Кроме того, в онлайн-утилите *draw*.*io* были разработаны диаграмма состояний и модель алгоритма.

С помощью встроенных в *Altium* *Designer* библиотечных элементов были разработаны компоненты на принципиальной схеме, а также посадочные места для каждого из них. После сопоставления библиотечных элементов, компонентов на схеме и посадочных мест, средствами *Altium* *Designer* можно смоделировать печатную плату устройства.

В *AutoCAD* были спроектированы необходимые чертежи, документация, такая как перечни, спецификация, сборочный чертёж.

С помощью интернет-сервиса *draw*.*io* можно удобно спроектировать диаграмму состояний и блок-схему алгоритма.

Заключение

В ходе курсового проекта разработано мобильный тепловизор и выполнены все поставленные задачи:

– проанализирована предметная область;

– спроектирована печатная плата;

– разработана конструкция устройства;

– разработана сопровождающая документация и необходимые чертежи;

– спроектирована программную часть устройства и разработан программный код для её реализации.

В ходе данного курсового проекта были использованы следующие программы и интернет-ресурсы: *AutoCAD*, *Altium* *Designer*, *draw*.*io*.

Разработанное в рамках проекта устройство может использоваться в качестве манипулятора «мышь» и устройства взаимодействия «человек-компьютер» не только с ПЭВМ, но и с мобильными системами.

Размер печатной платы устройства составляет 50х100мм. Она соответствует всем поставленным требованиям.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Мибильный тепловизор [Электронный ресурс]. – Режим доступа: *https*://*cyberleninka*.*ru*/*article*/*n*/*metod*-*izmeneniya*-*povedeniya*-*hid*-*ustroystv*-*pod*-*upravleniem*-*os*-*linux*.
2. Словарь бизнес-терминов [Электронный ресурс]. – Режим доступа: *https*://*dic*.*academic*.*ru*/*dic*.*nsf*/*business*/14833.
3. Тепловизо*р* [Электронный ресурс]. – Режим доступа : *http*://*microterm*.*ru*/*f*/*hid\_rus*.*pdf*.
4. Разбираем и собираем обратно стек тепловизо*р* [Электронный ресурс]. – Режим доступа: *https*://*habr*.*com*/*ru*/*post*/236401/.
5. Дескрипторы [Электронный ресурс]. – Режим доступа: *http*://*www*.*gaw*.*ru*/*html*.*cgi*/*txt*/*interface*/*usb*/*instr*/5.*htm*.
6. Lepton flir [Электронный ресурс]. – Режим доступа: *http*://*microsin*.*net*/*programming*/*arm*-*working*-*with*-*usb*/*usb*-*in*-*a*-*nutshell*-*part*2/.
7. Популярно о болометрах. Часть 1 [Электронный ресурс]. – Режим доступа: *https*://*www*.*rlocman*.*ru*/*review*/*article*.*html*?*di*=152278.
8. Народная электроника [Электронный ресурс] : Популярно о *USB*. – Режим доступа : *http*://*rus*-*linux*.*net*/*MyLDP*/*BOOKS*/*pop\_about\_USB*.*pdf*.
9. Микроконтроллеры *Cortex*-*M*0/*M*3/*M*4 [Электронный ресурс]. – Режим доступа: *https*://*mcucpu*.*ru*/*index*.*php*/*ucontrollers*/*mcu*/113-*mikrokontrollery*-*cortex*-*m*0*m*3*m*4.
10. Начинаем изучать *Cortex*-*M* на примере *STM*32 [Электронный ресурс]. – Режим доступа: *https*://*habr*.*com*/*en*/*post*/218825/.
11. *STM*32: контроллер *USB* *OTG* *full*-*speed* [Электронный ресурс]. – Режим доступа: *http*://*microsin*.*net*/*programming*/*arm*-*working*-*with*-*usb*/*stm*32-*usb*-*otg*-*full*-*speed*.*html*.
12. Работа с *SD* картой памяти по *SDIO* *STM*32 *F*4 *Discovery* [Электронный ресурс]. – Режим доступа: *http*://*blablacode*.*ru*/*mikrokontrollery*/452.
13. Устройство и принцип работы оптической мыши [Электронный ресурс]. – Режим доступа: *http*://*the*-*mostly*.*ru*/*misc*/*kak\_rabotayet\_opticheskaya\_mysh*.*html*.
14. Школа для электрика [Электронный ресурс]. – Режим доступа: *http*://*electricalschool*.*info*/*main*/*electroshemy*/*chto*-*takoe*-*strukturnaja*-*skhema*/.
15. Микроконтроллер: определение, задачи, разновидности, применение [Электронный ресурс]. – Режим доступа: *https*://*future*2*day*.*ru*/*mikrokontroller*/.
16. Создание чертежей электрических схем [Электронный ресурс]. – Режим доступа: *https*://*www*.*autodesk*.*ru*/*solutions*/*electrical*-*drawing*.
17. Назначение системы *AutoCAD* [Электронный ресурс]. – Режим доступа: *https*://*studopedia*.*su*/11*\_*101110*\_naznachenie*-*sistemi*-*AutoCAD*.*html*.
18. Основные элементы электроники [Электронный ресурс]. – Режим доступа: *http*://*selectelement*.*ru*/.
19. Бердичевский, Б. Е. Вопросы обеспечения надежности РЭА при разработке / Б. Е. Бердичевский. – М. : Сов. радио, 1977. – 384 с.
20. Справочник конструктора-приборостроителя. Проектирование. Основные нормы / В. Л. Соломахо [и др.] – М.: Высш. школа, 1988. – 272 с.
21. Материалы для пайки [Электронный ресурс]. – Режим доступа: *http*://*www*.*spider*-*gsm*.*ru*/*produkciya*/*materialy*-*dlya*-*pajki*/

**ПРИЛОЖЕНИЕ *A***

(обязательное)

ПЕРЕЧЕНЬ ЭЛЕМЕНТОВ К СХЕМЕ ЭЛЕКТРИЧЕСКОЙ ПРИНЦИПИАЛЬНОЙ

**ПРИЛОЖЕНИЕ Б**

(обязательное)

СПЕЦИФИКАЦИЯ К СБОРОЧНОМУ ЧЕРТЕЖУ ПЕЧАТНОЙ ПЛАТЫ

**ПРИЛОЖЕНИЕ В**

(обязательное)

ВЕДОМОСТЬ КУРСОВОГО ПРОЕКТА

**ПРИЛОЖЕНИЕ Г**

(обязательное)

ВИЗУАЛИЗИРОВАННАЯ ТРЁХМЕРНАЯ

МОДЕЛЬ ПЕЧАТНОЙ ПЛАТЫ

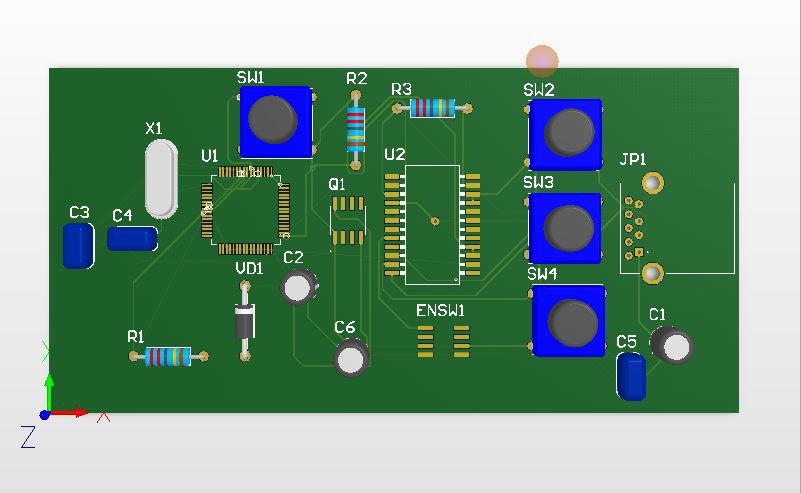


Рисунок Д.1 – Лицевая сторона печатной платы

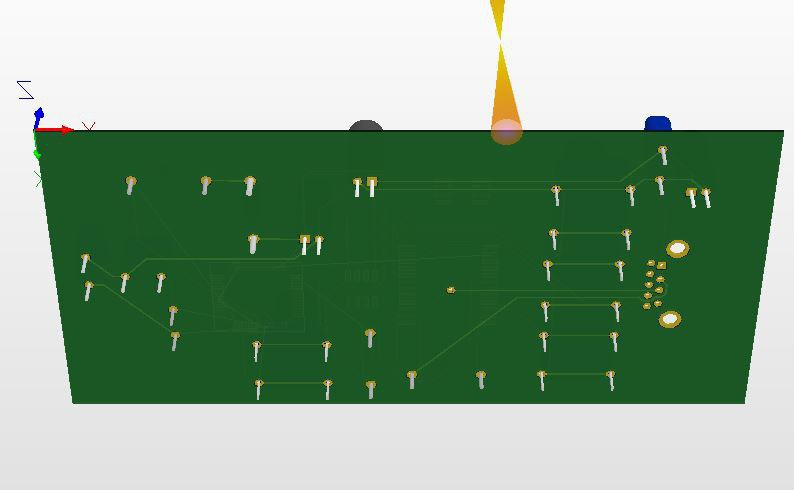


Рисунок Д.2 – Тыльная сторона печатной платы

**ПРИЛОЖЕНИЕ Д**

(обязательное)

ТЕКСТ ПРОГРАММЫ

Листинг файла *main*.*c*

/\*\*

  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

  \* *File* *Name*          : *main*.*c*

  \* *Description*        : *Main* *program* *body*

  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

  \*/

/\* *Includes* ------------------------------------------------------------------\*/

#*include* "*main*.*h*"

#*include* "*stm*32*f*10*xx\_hal*.*h*"

#*include* "*rtc*.*h*"

#*include* "*usart*.*h*"

#*include* "*usb\_device*.*h*"

#*include* "*gpio*.*h*"

/\* *Private* *variables* ---------------------------------------------------------\*/

/\* *USER* *CODE* *BEGIN* *PV* \*/

/\* *Private* *variables* ---------------------------------------------------------\*/

/\* *USER* *CODE* *END* *PV* \*/

/\* *Private* *function* *prototypes* -----------------------------------------------\*/

*void* *SystemClock\_Config*(*void*);

*void* *Error\_Handler*(*void*);

/\* *USER* *CODE* *BEGIN* *PFP* \*/

/\* *Private* *function* *prototypes* -----------------------------------------------\*/

/\* *USER* *CODE* *END* *PFP* \*/

*int* *main*(*void*)

{

*RTC\_TimeTypeDef* *sTime*;

*RTC\_DateTypeDef* *sDate*;

  // *WS*2812

*uint*8*\_t* *green*[] = {0*x*1*F*, 0*x*00, 0*x*00};

*uint*8*\_t* *red*[]   = {0*x*00, 0*x*1*F*, 0*x*00};

*uint*8*\_t* *blue*[]  = {0*x*00, 0*x*00, 0*x*1*F*};

*uint*8*\_t* *black*[] = {0*x*00, 0*x*00, 0*x*00};

  //*uint*8*\_t* *colors*[3][3] = { {0*x*1*F*, 0*x*00, 0*x*00}, {0*x*00, 0*x*1*F*, 0*x*00}, {0*x*00, 0*x*00, 0*x*1*F*} };

*uint*8*\_t* \**colors*[] = { *green*, *red*, *blue* };

*uint*8*\_t* *colorsIndex* = 0;

  // *HID* *Mouse*

*struct* *mouseHID\_t* {

*uint*8*\_t* *buttons*;

*int*8*\_t* *x*;

*int*8*\_t* *y*;

*int*8*\_t* *wheel*;

  };

*struct* *mouseHID\_t* *mouseHID*;

*mouseHID*.*buttons* = 0;

*mouseHID*.*x* = 10;

*mouseHID*.*y* = 0;

*mouseHID*.*wheel* = 0;

  /\* *USER* *CODE* *END* 1 \*/

  /\* *MCU* *Configuration*----------------------------------------------------------\*/

  /\* *Reset* *of* *all* *peripherals*, *Initializes* *the* *Flash* *interface* *and* *the* *Systick*. \*/

*HAL\_Init*();

  /\* *Configure* *the* *system* *clock* \*/

*SystemClock\_Config*();

  /\* *Initialize* *all* *configured* *peripherals* \*/

*MX\_GPIO\_Init*();

*MX\_RTC\_Init*();

*MX\_USART*1*\_UART\_Init*();

*MX\_USB\_DEVICE\_Init*();

  // *RTC* *settings*

*sTime*.*DayLightSaving* = *RTC\_DAYLIGHTSAVING\_NONE*;

*sTime*.*StoreOperation* = *RTC\_STOREOPERATION\_RESET*;

*if* (*HAL\_RTC\_SetTime*(&*hrtc*, &*sTime*, *RTC\_FORMAT\_BIN*) != *HAL\_OK*) {

*Error\_Handler*();

  }

*if* (*HAL\_RTC\_SetDate*(&*hrtc*, &*sDate*, *RTC\_FORMAT\_BIN*) != *HAL\_OK*) {

*Error\_Handler*();

  }

  /\* *Infinite* *loop* \*/

*while* (1)

  {

    // *RTC*

*HAL\_RTC\_GetTime*(&*hrtc*, &*sTime*, *RTC\_FORMAT\_BIN*);

*HAL\_RTC\_GetDate*(&*hrtc*, &*sDate*, *RTC\_FORMAT\_BIN*);

*HAL\_GPIO\_TogglePin*(*MOSFET\_GPIO\_Port*, *MOSFET\_Pin*);

*ws*2812*\_sendarray*(*colors*[*colorsIndex*], 3);

*colorsIndex* = (*colorsIndex*+1) % 3;

*HAL\_Delay*(1000);

    // *Send* *HID* *report*

*mouseHID*.*x* = 10;

*USBD\_HID\_SendReport*(&*hUsbDeviceFS*, &*mouseHID*, *sizeof*(*struct* *mouseHID\_t*));

  }

}

/\*\* *System* *Clock* *Configuration*

\*/

*void* *SystemClock\_Config*(*void*)

{

*RCC\_OscInitTypeDef* *RCC\_OscInitStruct*;

*RCC\_ClkInitTypeDef* *RCC\_ClkInitStruct*;

*RCC\_PeriphCLKInitTypeDef* *PeriphClkInit*;

    /\*\**Configure* *the* *main* *internal* *regulator* *output* *voltage*

    \*/

*\_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG*(*PWR\_REGULATOR\_VOLTAGE\_SCALE*1);

    /\*\**Initializes* *the* *CPU*, *AHB* *and* *APB* *busses* *clocks*

    \*/

*RCC\_OscInitStruct*.*OscillatorType* = *RCC\_OSCILLATORTYPE\_HSI*|*RCC\_OSCILLATORTYPE\_LSE*

                              |*RCC\_OSCILLATORTYPE\_HSI*48;

*RCC\_OscInitStruct*.*LSEState* = *RCC\_LSE\_ON*;

*RCC\_OscInitStruct*.*HSIState* = *RCC\_HSI\_ON*;

*RCC\_OscInitStruct*.*HSICalibrationValue* = 16;

*RCC\_OscInitStruct*.*HSI*48*State* = *RCC\_HSI*48*\_ON*;

*RCC\_OscInitStruct*.*PLL*.*PLLState* = *RCC\_PLL\_NONE*;

*if* (*HAL\_RCC\_OscConfig*(&*RCC\_OscInitStruct*) != *HAL\_OK*)

  {

*Error\_Handler*();

  }

    /\*\**Initializes* *the* *CPU*, *AHB* *and* *APB* *busses* *clocks*

    \*/

*RCC\_ClkInitStruct*.*ClockType* = *RCC\_CLOCKTYPE\_HCLK*|*RCC\_CLOCKTYPE\_SYSCLK*

                              |*RCC\_CLOCKTYPE\_PCLK*1|*RCC\_CLOCKTYPE\_PCLK*2;

*RCC\_ClkInitStruct*.*SYSCLKSource* = *RCC\_SYSCLKSOURCE\_HSI*;

*RCC\_ClkInitStruct*.*AHBCLKDivider* = *RCC\_SYSCLK\_DIV*1;

*RCC\_ClkInitStruct*.*APB*1*CLKDivider* = *RCC\_HCLK\_DIV*1;

*RCC\_ClkInitStruct*.*APB*2*CLKDivider* = *RCC\_HCLK\_DIV*1;

*if* (*HAL\_RCC\_ClockConfig*(&*RCC\_ClkInitStruct*, *FLASH\_LATENCY\_*0) != *HAL\_OK*)

  {

*Error\_Handler*();

  }

*PeriphClkInit*.*PeriphClockSelection* = *RCC\_PERIPHCLK\_USART*1|*RCC\_PERIPHCLK\_RTC*

                              |*RCC\_PERIPHCLK\_USB*;

*PeriphClkInit*.*Usart*1*ClockSelection* = *RCC\_USART*1*CLKSOURCE\_PCLK*2;

*PeriphClkInit*.*RTCClockSelection* = *RCC\_RTCCLKSOURCE\_LSE*;

*PeriphClkInit*.*UsbClockSelection* = *RCC\_USBCLKSOURCE\_HSI*48;

*if* (*HAL\_RCCEx\_PeriphCLKConfig*(&*PeriphClkInit*) != *HAL\_OK*)

  {

*Error\_Handler*();

  }

    /\*\**Configure* *the* *Systick* *interrupt* *time*

    \*/

*HAL\_SYSTICK\_Config*(*HAL\_RCC\_GetHCLKFreq*()/1000);

    /\*\**Configure* *the* *Systick*

    \*/

*HAL\_SYSTICK\_CLKSourceConfig*(*SYSTICK\_CLKSOURCE\_HCLK*);

  /\* *SysTick\_IRQn* *interrupt* *configuration* \*/

*HAL\_NVIC\_SetPriority*(*SysTick\_IRQn*, 0, 0);

}

/\*\*

  \* @*brief*  *This* *function* *is* *executed* *in* *case* *of* *error* *occurrence*.

  \* @*param*  *None*

  \* @*retval* *None*

  \*/

*void* *Error\_Handler*(*void*) {}

#*ifdef* *USE\_FULL\_ASSERT*

/\*\*

   \* @*brief* *Reports* *the* *name* *of* *the* *source* *file* *and* *the* *source* *line* *number*

   \* *where* *the* *assert\_param* *error* *has* *occurred*.

   \* @*param* *file*: *pointer* *to* *the* *source* *file* *name*

   \* @*param* *line*: *assert\_param* *error* *line* *source* *number*

   \* @*retval* *None*

   \*/

*void* *assert\_failed*(*uint*8*\_t*\* *file*, *uint*32*\_t* *line*)

{

*printf*("*Wrong* *parameters* *value*: *file* %*s* *on* *line* %*d\r\n*", *file*, *line*);

}

#*endif*

Листинг кода файла *usb\_device*.*c*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

  \* @*file*           : *USB\_DEVICE*

  \* @*version*        : *v*1.0*\_Cube*

  \* @*brief*          : *This* *file* *implements* *the* *USB* *Device*

  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\* *Includes* ------------------------------------------------------------------\*/

#*include* "*usb\_device*.*h*"

#*include* "*usbd\_core*.*h*"

#*include* "*usbd\_desc*.*h*"

#*include* "*usbd\_hid*.*h*"

/\* *USB* *Device* *Core* *handle* *declaration* \*/

*USBD\_HandleTypeDef* *hUsbDeviceFS*;

/\* *init* *function* \*/

*void* *MX\_USB\_DEVICE\_Init*(*void*)

{

  /\* *Init* *Device* *Library*,*Add* *Supported* *Class* *and* *Start* *the* *library*\*/

*USBD\_Init*(&*hUsbDeviceFS*, &*FS\_Desc*, *DEVICE\_FS*);

*USBD\_RegisterClass*(&*hUsbDeviceFS*, &*USBD\_HID*);

*USBD\_Start*(&*hUsbDeviceFS*);

}

Листинг кода файла *usbd\_desc*.*c*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

  \* @*file*           : *usbd\_desc*.*c*

  \* @*version*        : *v*1.0*\_Cube*

  \* @*brief*          : *This* *file* *implements* *the* *USB* *Device* *descriptors*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* *Includes* ------------------------------------------------------------------\*/

#*include* "*usbd\_core*.*h*"

#*include* "*usbd\_desc*.*h*"

#*include* "*usbd\_conf*.*h*"

/\*\* @*addtogroup* *STM*32*\_USB\_OTG\_DEVICE\_LIBRARY*

  \* @{

  \*/

/\*\* @*defgroup* *USBD\_DESC*

  \* @*brief* *USBD* *descriptors* *module*

  \* @{

  \*/

/\*\* @*defgroup* *USBD\_DESC\_Private\_TypesDefinitions*

  \* @{

  \*/

/\*\*

  \* @}

  \*/

/\*\* @*defgroup* *USBD\_DESC\_Private\_Defines*

  \* @{

  \*/

#*define* *USBD\_VID*     1155

#*define* *USBD\_LANGID\_STRING*     1033

#*define* *USBD\_MANUFACTURER\_STRING*     "*ioPush*.*net*"

#*define* *USBD\_PID\_FS*     22315

#*define* *USBD\_PRODUCT\_STRING\_FS*     "*OpticMouse*"

#*define* *USBD\_SERIALNUMBER\_STRING\_FS*     "00000000001*A*"

#*define* *USBD\_CONFIGURATION\_STRING\_FS*     "*HID* *Config*"

#*define* *USBD\_INTERFACE\_STRING\_FS*     "*HID* *Interface*"

/\*\* @*defgroup* *USBD\_DESC\_Private\_Macros*

  \*/

/\*\* @*defgroup* *USBD\_DESC\_Private\_Variables*

  \* @{

  \*/

*uint*8*\_t* \*     *USBD\_FS\_DeviceDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_LangIDStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_ManufacturerStrDescriptor* ( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_ProductStrDescriptor* ( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_SerialStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_ConfigStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

*uint*8*\_t* \*     *USBD\_FS\_InterfaceStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*);

#*ifdef* *USB\_SUPPORT\_USER\_STRING\_DESC*

*uint*8*\_t* \*     *USBD\_FS\_USRStringDesc* (*USBD\_SpeedTypeDef* *speed*, *uint*8*\_t* *idx* , *uint*16*\_t* \**length*);

#*endif* /\* *USB\_SUPPORT\_USER\_STRING\_DESC* \*/

*USBD\_DescriptorsTypeDef* *FS\_Desc* =

{

*USBD\_FS\_DeviceDescriptor*,

*USBD\_FS\_LangIDStrDescriptor*,

*USBD\_FS\_ManufacturerStrDescriptor*,

*USBD\_FS\_ProductStrDescriptor*,

*USBD\_FS\_SerialStrDescriptor*,

*USBD\_FS\_ConfigStrDescriptor*,

*USBD\_FS\_InterfaceStrDescriptor*,

};

#*if* *defined* ( *\_\_ICCARM\_\_* ) /\*!< *IAR* *Compiler* \*/

  #*pragma* *data\_alignment*=4

#*endif*

/\* *USB* *Standard* *Device* *Descriptor* \*/

*\_\_ALIGN\_BEGIN* *uint*8*\_t* *USBD\_FS\_DeviceDesc*[*USB\_LEN\_DEV\_DESC*] *\_\_ALIGN\_END* =

  {

    0*x*12,                       /\**bLength* \*/

*USB\_DESC\_TYPE\_DEVICE*,       /\**bDescriptorType*\*/

    0*x*00,                       /\* *bcdUSB* \*/

    0*x*02,

    0*x*00,                       /\**bDeviceClass*\*/

    0*x*00,                       /\**bDeviceSubClass*\*/

    0*x*00,                       /\**bDeviceProtocol*\*/

*USB\_MAX\_EP*0*\_SIZE*,          /\**bMaxPacketSize*\*/

*LOBYTE*(*USBD\_VID*),           /\**idVendor*\*/

*HIBYTE*(*USBD\_VID*),           /\**idVendor*\*/

*LOBYTE*(*USBD\_PID\_FS*),           /\**idVendor*\*/

*HIBYTE*(*USBD\_PID\_FS*),           /\**idVendor*\*/

    0*x*00,                       /\**bcdDevice* *rel*. 2.00\*/

    0*x*02,

*USBD\_IDX\_MFC\_STR*,           /\**Index* *of* *manufacturer*  *string*\*/

*USBD\_IDX\_PRODUCT\_STR*,       /\**Index* *of* *product* *string*\*/

*USBD\_IDX\_SERIAL\_STR*,        /\**Index* *of* *serial* *number* *string*\*/

*USBD\_MAX\_NUM\_CONFIGURATION*  /\**bNumConfigurations*\*/

  } ;

/\* *USB\_DeviceDescriptor* \*/

#*if* *defined* ( *\_\_ICCARM\_\_* ) /\*!< *IAR* *Compiler* \*/

  #*pragma* *data\_alignment*=4

#*endif*

/\* *USB* *Standard* *Device* *Descriptor* \*/

*\_\_ALIGN\_BEGIN* *uint*8*\_t* *USBD\_LangIDDesc*[*USB\_LEN\_LANGID\_STR\_DESC*] *\_\_ALIGN\_END* =

{

*USB\_LEN\_LANGID\_STR\_DESC*,

*USB\_DESC\_TYPE\_STRING*,

*LOBYTE*(*USBD\_LANGID\_STRING*),

*HIBYTE*(*USBD\_LANGID\_STRING*),

};

#*if* *defined* ( *\_\_ICCARM\_\_* ) /\*!< *IAR* *Compiler* \*/

  #*pragma* *data\_alignment*=4

#*endif*

*\_\_ALIGN\_BEGIN* *uint*8*\_t* *USBD\_StrDesc*[*USBD\_MAX\_STR\_DESC\_SIZ*] *\_\_ALIGN\_END*;

/\*\*

  \* @}

  \*/

/\*\* @*defgroup* *USBD\_DESC\_Private\_FunctionPrototypes*

  \* @{

  \*/

/\*\*

  \* @}

  \*/

/\*\* @*defgroup* *USBD\_DESC\_Private\_Functions*

  \* @{

  \*/

/\*\*

\* @*brief*  *USBD\_FS\_DeviceDescriptor*

\*         *return* *the* *device* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_DeviceDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

  \**length* = *sizeof*(*USBD\_FS\_DeviceDesc*);

*return* *USBD\_FS\_DeviceDesc*;

}

/\*\*

\* @*brief*  *USBD\_FS\_LangIDStrDescriptor*

\*         *return* *the* *LangID* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_LangIDStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

  \**length* =  *sizeof*(*USBD\_LangIDDesc*);

*return* *USBD\_LangIDDesc*;

}

/\*\*

\* @*brief*  *USBD\_FS\_ProductStrDescriptor*

\*         *return* *the* *product* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_ProductStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

*if*(*speed* == 0)

  {

*USBD\_GetString* (*USBD\_PRODUCT\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*else*

  {

*USBD\_GetString* (*USBD\_PRODUCT\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*return* *USBD\_StrDesc*;

}

/\*\*

\* @*brief*  *USBD\_FS\_ManufacturerStrDescriptor*

\*         *return* *the* *manufacturer* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_ManufacturerStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

*USBD\_GetString* (*USBD\_MANUFACTURER\_STRING*, *USBD\_StrDesc*, *length*);

*return* *USBD\_StrDesc*;

}

/\*\*

\* @*brief*  *USBD\_FS\_SerialStrDescriptor*

\*         *return* *the* *serial* *number* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_SerialStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

*if*(*speed*  == *USBD\_SPEED\_HIGH*)

  {

*USBD\_GetString* (*USBD\_SERIALNUMBER\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*else*

  {

*USBD\_GetString* (*USBD\_SERIALNUMBER\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*return* *USBD\_StrDesc*;

}

/\*\*

\* @*brief*  *USBD\_FS\_ConfigStrDescriptor*

\*         *return* *the* *configuration* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_ConfigStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

*if*(*speed*  == *USBD\_SPEED\_HIGH*)

  {

*USBD\_GetString* (*USBD\_CONFIGURATION\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*else*

  {

*USBD\_GetString* (*USBD\_CONFIGURATION\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*return* *USBD\_StrDesc*;

}

/\*\*

\* @*brief*  *USBD\_HS\_InterfaceStrDescriptor*

\*         *return* *the* *interface* *string* *descriptor*

\* @*param*  *speed* : *current* *device* *speed*

\* @*param*  *length* : *pointer* *to* *data* *length* *variable*

\* @*retval* *pointer* *to* *descriptor* *buffer*

\*/

*uint*8*\_t* \*  *USBD\_FS\_InterfaceStrDescriptor*( *USBD\_SpeedTypeDef* *speed* , *uint*16*\_t* \**length*)

{

*if*(*speed* == 0)

  {

*USBD\_GetString* (*USBD\_INTERFACE\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*else*

  {

*USBD\_GetString* (*USBD\_INTERFACE\_STRING\_FS*, *USBD\_StrDesc*, *length*);

  }

*return* *USBD\_StrDesc*;

}

/\*\*

  \* @}

  \*/

/\*\*

  \* @}

  \*/

/\*\*

  \* @}

  \*/

**ПРИЛОЖЕНИЕ Е**

(обязательное)

ПРОВЕРКА В СИСТЕМЕ АНТИПЛАГИАТ

