МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Рыбинский государственный авиационный технический университет

имени П. А. Соловьева»

Институт информационных технологий и систем управления

Кафедра вычислительных систем

Курсовой проект

по дисциплине «Микропроцессорные системы»

на тему «Генератор импульсов»

Пояснительная записка

Соискатель, студент группы ИВБ1–20 Морозов В. В.

Руководитель канд. техн. наук, профессор Комаров В. М.

Нормконтролер канд. техн. наук, профессор Комаров В. М.

Рыбинск 2023

Содержание

[Введение 3](#_Toc136719308)

[1 Анализ технического задания 4](#_Toc136719309)

[2 Декомпозиция программы 6](#_Toc136719310)

[3 Разработка структуры данных программы 9](#_Toc136719311)

[4 Алгоритмизация программы 11](#_Toc136719312)

Введение

При современном уровне развития элементной базы вычислительной техники наиболее целесообразно для построения устройств цифровой обработки информации использовать микропроцессоры. Использование микропроцессоров обеспечивает возможность построения таких устройств в виде одной печатной платы. При этом они имеют малые габариты и массу, небольшую стоимость и потребляемую мощность.

1 Анализ технического задания

Из анализа технического задания следует, устройство должно обеспечивать:

1. Генерацию импульсов прямоугольной формы;
2. Изменение полярности импульсов;
3. Изменение частоты от 20 Гц до 150 Гц с шагов в 10 Гц;
4. Изменение амплитуды импульсов от 0 В до 5 В с шагов в 1 В;
5. Изменение длительности импульсов от 1мс до 5 мс с шагов в 1 мс;
6. Вывод сгенерированных импульсов на дисплей, состоящий из 16 матричных индикаторов размерностью 8 строк и 8 столбцов;
7. Вывод заданной частоты на дисплей, состоящий из 3 знакосинтезирующих семисегментных индикаторов;
8. Вывод заданной амплитуды на дисплей, состоящий из 1 знакосинтезирующего семисегментного индикатора;
9. Вывод заданной длительности импульсов на дисплей, состоящий из 1 знакосинтезирующего семисегментного индикатора;
10. Для задания частоты используются кнопки без фиксации (+ 10 Гц и – 10 Гц)
11. Для задания амплитуды используются кнопки без фиксации (+ 1 В и – 10 В)
12. Для задания длительности импульсов используются кнопки без фиксации (+ 1 мс и – 1 мс);
13. Для изменения полярности используется кнопка без фиксации.

На основании перечисленных требований можно представить разрабатываемое устройство в виде «черной сферы» (рисунок 1.1), а также изобразить лицевую панель устройства (рисунок 1.2). Исходя из этих представлений начинается техническое проектирование устройства.



Рисунок 1.1 – Представление устройства охранной сигнализации в виде «черной сферы»

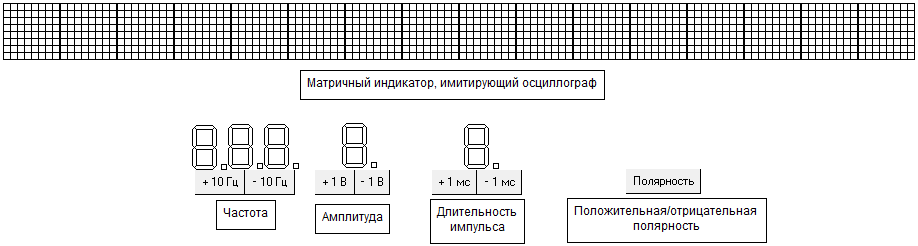


Рисунок 1.2 – Лицевая панель устройства для генерации импульсов

2 Декомпозиция программы

Анализируя требования к программе с учётом лицевой панели устройства, получим исходную схему представления поставленной задачи.



Рисунок 2.1 – Исходная схема представления задачи «Генератор импульсов»

Из этой схемы следует, что проектируемая программа должна обрабатывать входную информацию с кнопок управления, выходная информация с которых имеет два значения: лог. «0» и лог. «1», отображать выходную информацию на 3 дисплеях, состоящих из закосинтезирующих семисегментных индикаторов, и дисплее, состоящем из матричных индикаторов.

На первом этапе (рисунок 2.2) представим программу в виде двух задач: «Обработка входной информации», «Обработка выходной информации». Подзадачи «Обработка входной информации» и «Обработка выходной информации», выделенные на первом этапе, являются сложными и требуют дальнейшего разбиения.



Рисунок 2.2 – Статическая модель программы (после первого этапа декомпозиции)

На втором этапе (рисунки 2.3) подзадачу «Обработка входной информации» разобьем на подзадачи «Ввод с кнопок», «Формирование информации», «Контроль ввода»

Подзадачу «Обработка выходной информации» разобьем на подзадачи «Преобразование в десятичный код», «Формирование массива отображения», «Формирование изображения импульсов», «Вывод изображения импульсов на матрицу», «Вывод числовой информации».

Подзадачи, выделенные на втором этапе, являются простыми и не требуют дальнейшей декомпозиции.



Рисунок 2.3 – Статическая модель программы (после второго этапа декомпозиции)

3 Разработка структуры данных программы

Процесс решения любой задачи состоит из активации тех или иных программных модулей, выполняющих некоторую подзадачу. Передача информации между модулями осуществляется с помощью наборов данных.

Дальнейшее проектирование программы заключается в переходе от ее статической модели к динамической, то есть к ее алгоритмическому описанию.

| Наименование данных | Символьное имя | Формат данных | Кодирование данных | | | | | | | | Примечание |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *D*7 | *D6* | *D5* | *D4* | *D3* | *D2* | *D1* | *D0* |
| Амплитуда | *Amplitude* | Числовой байт | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | *A* = 5 В |
| Частота | *Frequancy* | Числовой байт | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | *f* = 150 Гц |
| Длитель-ность импульса | *Pulse-Duration* | Числовой байт | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | *t*и = 5 мс |
| Период | *Pulse-Period* | Числовой байт | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | *T* = 6 мс |
| Число полных импульсов | *Pulses-Count* | Числовой байт | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 21 |
| Длитель-ность паузы между импуль-сами | *Pause-Period* | Числовой байт | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | *t*п = 1 мс |
| Начальное положение указателя | *Start-Position* | Числовой байт | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 5 строка матрицы (20*h*) |
| Образ кнопки | *KeyImage* | Упакованный байт | \* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |
| Частота в десятичном коде | *FrequancyBCD* | Массив двоично-десятичных байтов  (2 байта) | 0  \* | 1  \* | 0  \* | 1  \* | 0  0 | 0  0 | 0  0 | 0  1 | 50  1 |
| Массив отобра-жения частоты | *FrequancyImage* | Массив байтов  (3 байта) | 0  0  0 | 0  0  0 | 0  0  0 | 0  0  0 | 0  0  0 | 0  1  0 | 0  0  0 | 0  1  1 | 0  5  1 |
| Массив отобра-жения импульсов | *Pulses-Image* | Массив байтов  (128 байтов) | \*  .  \* | \*  .  \* | \*  .  \* | \*  .  \* | \*  .  \* | \*  .  \* | \*  .  \* | \*  .  \* | 1 колонка  128 колонка |
| Флаг неактивных кнопок управления | *NoInput-ErrorFlag* | Флаговый байт | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | Есть нажатие  Нет нажатия |
| Флаг полярности | *Polarity-Flag* | Флаговый байт | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | 0  1 | Положительная полярность  Отрицательная полярность |

Таблица 3.1 – Структура данных программы

4 Алгоритмизация программы

Вычислительный процесс, реализуемый программой, представляет собой последовательность дискретных шагов по преобразованию данных.

Для решения поставленной задачи необходимы следующие программные модули с соответствующими символьными именами:

* «Функциональная подготовка» (*Initialization*);
* «Ввод с кнопок» (*KeyRead*);
* «Контроль ввода» (*KeyCheck*);
* «Формирование информации» (*DataSetting*);
* «Преобразование частоты из двоичного в десятичный код» (*BinaryToBCD*);
* «Формирование массива отображения» (*UnpackFrequancyBCD*);
* «Вывод числовой информации» (*DisplayData*);
* «Формирование изображения импульсов» (*PulseImageForming*);
* «Вывод изображения импульсов на матрицу» (*MatrixOutput*).

Символьные имена присвоены соответствующим программным модулям с целью их дальнейшего использования. Выбранные имена отражают содержательный смысл этих модулей.

Последовательность модулей в логической конструкции «Следование» определяется логикой решения задачи.

Алгоритм макроуровня программы проектируемого устройства в одноуровневом представлении приведен на рисунке 4.1. Для повышения информативности ГСА на ней изображены входные и выходные данные, обрабатываемые каждым программным модулем. Это позволяет проследить, что каждый модуль располагается в той точке программного кольца, в которой существуют все обрабатываемые им данные.

Далее необходимо перейти к двухуровневому представлению алгоритма. Этот переход осуществляется формально. Алгоритм программы проектируемого устройства в двухуровневом представлении приведен на рисунке 4.2. Из этого алгоритма следует, что на макроуровне программы находятся лишь команды вызова программных модулей.



Рисунок 4.1 – Алгоритм программы «Генератор импульсов» (одноуровневое представление)



Рисунок 4.2 – Алгоритм программы «Генератор импульсов» (двухуровневое представление)

После разработки алгоритма макроуровня требуется разработать алгоритмы всех программных модулей нижнего уровня. Алгоритмы программных модулей и подмодулей приведены на рисунках 4.3 – 4.17.

Модуль «Ввод с кнопок» (*KeyRead*)

Этот модуль служит для ввода состояния кнопок задания параметров с последующим размещением их образа в памяти.

ГСА этого модуля приведена на рисунке 4.3.

Электромеханическим переключателям свойственно явление дребезга контактов, который необходимо гасить. Гашение дребезга контактов можно выполнить программным путем. ГСА подмодуля гашения дребезга *VibrDestr* приведена на рисунке 4.4. Критерием окончания дребезга является считывание с кнопок одного и того же состояния заданное количество раз.





Рисунок 4.3 – Алгоритм модуля «Ввод с кнопок» (общий алгоритм)



Рисунок 4.4 – Алгоритм подмодуля «Гашение дребезга»

Модуль «Ввод с кнопок» (*KeyCheck*)

Этот модуль служит для проверки наличия нажатия кнопок с последующим формированием флага отсутствия нажатия.

ГСА этого модуля приведена на рисунке 4.5.





Рисунок 4.5 – Алгоритм модуля «Контроль ввода»

Модуль «Формирование информации» (*DataSetting*)

Этот модуль служит для задания параметров генерации импульсов исходя из образа нажатой кнопки и флага отсутствия нажатия. Для удобства этот модуль разбит на ряд подмодулей.

ГСА этого модуля приведена на рисунке 4.6.





Рисунок 4.6 – Алгоритм модуля «Формирование информации» (общий алгоритм)

Подмодуль «Увеличение частоты» (*FrequancyAddition*)

Этот подмодуль служит для увеличения параметра частоты импульсов.

ГСА этого модуля приведена на рисунке 4.7.



Рисунок 4.7 – Алгоритм подмодуля «Увеличение частоты»

Подмодуль «Уменьшение частоты» (*FrequancySubtraction*)

Этот подмодуль служит для уменьшения параметра частоты импульсов.

ГСА этого модуля приведена на рисунке 4.8.



Рисунок 4.8 – Алгоритм подмодуля «Уменьшение частоты»

Подмодуль «Увеличение амплитуды» (*AmplitudeAddition*)

Этот подмодуль служит для увеличения параметра амплитуды импульсов.

ГСА этого модуля приведена на рисунке 4.9.



Рисунок 4.9 – Алгоритм подмодуля «Увеличение амплитуды»

Подмодуль «Уменьшение амплитуды» (*AmplitudeSubtraction*)

Этот подмодуль служит для уменьшения параметра амплитуды импульсов.

ГСА этого модуля приведена на рисунке 4.10.



Рисунок 4.10 – Алгоритм подмодуля «Уменьшение амплитуды»

Подмодуль «Увеличение длительности импульсов» (*PulseDurationAddition*)

Этот подмодуль служит для увеличения параметра длительности импульсов.

ГСА этого модуля приведена на рисунке 4.11.



Рисунок 4.11 – Алгоритм подмодуля «Увеличение длительности импульсов»

Подмодуль «Уменьшение длительности импульсов» (*PulseDurationSubtraction*)

Этот подмодуль служит для уменьшения параметра длительности импульсов.

ГСА этого модуля приведена на рисунке 4.12.



Рисунок 4.12 – Алгоритм подмодуля «Уменьшение длительности импульсов»

Подмодуль «Установка полярности» (*PolaritySetting*)

Этот подмодуль служит для изменения параметра полярности импульсов.

ГСА этого модуля приведена на рисунке 4.13.



Рисунок 4.13– Алгоритм подмодуля «Установка полярности»

Подмодуль «Вычисление периода следования импульсов и паузы между импульсами» (*PulsePeriodAndPauseCalculation*)

Этот подмодуль служит для вычисления параметра периода следования импульсов исходя из заданной частоты и на основе уже известных периода и длительности импульсов вычисляется длительность паузы между импульсами.

ГСА этого модуля приведена на рисунке 4.14.



Рисунок 4.14– Алгоритм подмодуля «Вычисление периода следования импульсов и паузы между импульсами»

Подмодуль «Вычисление количества полных импульсов» (*PulsesCountCalculation*)

Этот подмодуль служит для вычисления количества полных импульсов, которые могут поместиться на матричном индикаторе, для последующего формирования изображения импульсов.

ГСА этого модуля приведена на рисунке 4.15.



Рисунок 4.15 – Алгоритм подмодуля «Вычисление количества полных импульсов»

Подмодуль «Установка начального положения» (*StartPositionSetting*)

Этот подмодуль служит для установки начального положения указателя исходя из заданной полярности импульсов, который используется для последующего формирования изображения импульсов.

ГСА этого модуля приведена на рисунке 4.16.



Рисунок 4.16 – Алгоритм подмодуля «Установка начального положения»

Модуль «Преобразование частоты из двоичного в десятичный код» (*BinaryToBCD*)

Этот модуль служит для перевода частоты из двоичного кода в десятичный упакованный *BCD* код. Десятичный код с последующей распаковкой необходим для табличного преобразования и последующего вывода значения частоты на дисплей.

ГСА этого модуля приведена на рисунке 4.17.



Рисунок 4.17 – Алгоритм подмодуля «Преобразование частоты из двоичного в десятичный код»

Модуль «Формирование массива отображения» (*UnpackFrequancyBCD*)

Этот модуль служит для распаковки значения частоты в десятичном *BCD* коде. Распакованный формат можно непосредственно использовать в качестве массива отображения для табличного преобразования и вывода значения частоты на дисплей.

ГСА этого модуля приведена на рисунке 4.18.



Рисунок 4.18 – Алгоритм подмодуля «Формирование массива отображения»

Модуль «Формирование изображения импульсов» (*PulseImageForming*)

Этот модуль служит для формирования изображения импульсов из заданных ранее параметров генерации. Для удобства этот модуль разбит на ряд подмодулей.

ГСА этого модуля приведена на рисунке 4.19.



Рисунок 4.19 – Алгоритм подмодуля «Формирование изображения импульсов»

Подмодуль «Формирование оси абсцисс» (*XAxisForming*)

Этот подмодуль служит для формирования оси абсцисс в зависимости от параметров полярности импульсов и амплитуды.

ГСА этого модуля приведена на рисунке 4.20.



Рисунок 4.20 – Алгоритм подмодуля «Формирование оси абсцисс»

Подмодуль «Формирование изображения фронта импульса» (*AmplitudeUpForming*)

Этот подмодуль служит для формирования изображения переднего фронта импульса в зависимости от параметров полярности импульсов и амплитуды.

ГСА этого модуля приведена на рисунке 4.21.



Рисунок 4.21 – Алгоритм подмодуля «Формирование изображения фронта импульса»

Подмодуль «Формирование изображения длительности импульса» (*PulseDurationForming*)

Этот подмодуль служит для формирования изображения длительности импульса в зависимости от параметра длительности импульса.

ГСА этого модуля приведена на рисунке 4.22.



Рисунок 4.22 – Алгоритм подмодуля «Формирование изображения длительности импульса»

Подмодуль «Формирование изображения среза импульса» (*AmplitudeDownForming*)

Этот подмодуль служит для формирования изображения среза импульса в зависимости от параметров полярности импульсов и амплитуды.

ГСА этого модуля приведена на рисунке 4.23.



Рисунок 4.23 – Алгоритм подмодуля «Формирование изображения среза импульса»

Подмодуль «Формирование изображения паузы между импульсами» (*PauseDurationForming*)

Этот подмодуль служит для формирования изображения паузы между импульса в зависимости от параметра паузы между импульсами.

ГСА этого модуля приведена на рисунке 4.24.



Рисунок 4.24 – Алгоритм подмодуля «Формирование изображения паузы между импульсами»

Модуль «Вывод числовой информации» (*DisplayData*)

Этот модуль служит для вывода заданных параметров генерации на соответствующие дисплеи. Для удобства этот модуль разбит на ряд подмодулей.

ГСА этого модуля приведена на рисунке 4.25.



Рисунок 4.25 – Алгоритм модуля «Вывод числовой информации»

Подмодуль «Отображение частоты на дисплей» (*DisplayFrequancy*)

Этот подмодуль служит для отображения заданной частоты импульсов на соответствующем дисплее.

ГСА этого модуля приведена на рисунке 4.26.



Рисунок 4.26 – Алгоритм подмодуля «Отображение частоты на дисплей»

Подмодуль «Отображение амплитуды на дисплей» (*DisplayAmplitude*)

Этот подмодуль служит для отображения заданной амплитуды импульсов на соответствующем дисплее.

ГСА этого модуля приведена на рисунке 4.27.



Рисунок 4.27 – Алгоритм подмодуля «Отображение амплитуды на дисплей»

Подмодуль «Отображение длительности импульсов на дисплей» (*DisplayPulseDuration*)

Этот подмодуль служит для отображения заданной длительности импульсов на соответствующем дисплее.

ГСА этого модуля приведена на рисунке 4.28.



Рисунок 4.28 – Алгоритм подмодуля «Отображение длительности импульсов на дисплей»

Модуль «Вывод изображения импульсов на матрицу» (*MatrixOutput*)

Этот модуль служит для вывода сформированных ранее изображений импульсов соответствующий дисплей, который имитирует осциллограф.

ГСА этого модуля приведена на рисунке 4.29.



Рисунок 4.29 – Алгоритм модуля «Вывод изображения импульсов на матрицу»

Модуль «Функциональная подготовка» (*Initialization*)

Этот модуль предназначен для начальной установки некоторых наборов данных или служебных ячеек, которые служат для хранения вспомогательной информации. Для удобства выделен один подмодуль.

ГСА этого модуля приведена на рисунке 4.30.



Рисунок 4.30 – Алгоритм модуля «Функциональная подготовка»

Подмодуль «Копирование массива цифр и таблицы преобразования из сегмента кода в сегмент данных» (*CopyArraysToDataSegment*)

Этот подмодуль служит для копирования данных, необходимых для табличного преобразования значений частоты, амплитуды и длительности импульса в их изображения, которые нужны для вывода на дисплеи.

ГСА этого модуля приведена на рисунке 4.31.



Рисунок 4.31 – Алгоритм подмодуля «Копирование массива цифр и таблицы преобразования из сегмента кода в сегмент данных»

5 Кодирование программы

Кодирование программы представляет собой запись алгоритмов проектируемой программы на языке ассемблер.

При кодировании программы используются ранее разработанные структура данных (таблица 3.1) и ГСА (рисунки 4.1 – 4.31).

Для полного кодирования программы остается лишь определить таблицу преобразования кодов десятичных цифр из массива отображения в семисегментные коды, которые выводятся на семисегментные знакосинтезирующие индикаторы для отображения соответствующей цифры.

Эта таблица зависит от выбранного варианта подключения сегментов индикатора к разрядам управляющего порта вывода. Для принятого варианта зажигания индикаторов нулём преобразование должно выполняться по таблице 5.1.

Исходный текст проектируемой программы приведен в Приложении А.

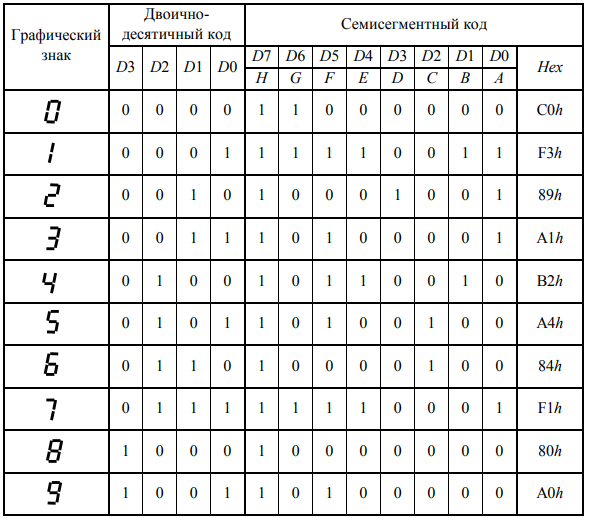


Таблица 5.1 – Кодирования кодов

6 Тестирование и отладка программы

Для тестирования и отладки разработанной программы будет использоваться интегрированная программная среда Design Microsystem. Эта среда позволяет на архитектурном уровне представить проектируемое устройство и работать с ним как с реальным устройством.

В состав архитектуры интерфейса устройства входят 1 порт ввода, 7 кнопок без фиксации для задания параметров генерации, 1 порт вывода для активации строк матриц, 1 порт вывода для активации колонок матриц, 2 порта вывода для подачи питания на матрицы, 1 порт вывода для подачи питания на знакосинтезирующие семисегментные индикаторы, 1 порт вывода для активации сегментов индикаторов, 16 матричных индикаторов для вывода на них изображения импульсов.

Для составления архитектуры устройства выберем необходимые интерфейсные элементы и разместим их на экране, соединяя в соответствии с архитектурой. После этого установим интерфейсным элементам необходимые свойства:

* для портов ввода/вывода зададим адреса в соответствии с исходным текстом программы;
* для знакосинтезирующих семисегментных индикаторов выберем уровень зажигания, равный «Зажигается нулём»;
* для кнопок определим исходное состояние, соответствующее ненажатой кнопке, как «нормально незамкнута».

Полученная в среде Design Microsystem архитектура электронного термометра приведена в приложении Б.

От архитектурного представления устройства легко перейти к его лицевой панели, которая представлена в приложении В.

Далее с помощью текстового редактора занесем исходный текст программы.

После занесения программы выполним ее ассемблирование и компоновку, то есть сборку программы, а затем выполним ее.

7 Руководство пользователя

При включении устройства на короткое время включается звонок и загораются все двоичные индикаторы.

Светодиодные индикаторы состояния помещений погашены. При постановке помещения под охрану светодиодный индикатор соответствующего помещения загорается.

Светодиоды, отображающие тип тревожной ситуации, погашены и загораются при нажатии на соответствующие кнопки «Взлом» или «Пожар».

Звонок выключен.

Устройство опрашивает датчики пожара, взлома, кнопки управления отображением и кнопки постановки комнат под охрану.

При наличии пожара в любой из комнат или взлома в комнате, находящейся под охраной, начинает мигать светодиод соответствующей комнаты, светодиод «Тревога», периодически включается звонок. Если пожар потушен или взлом ликвидирован, звонок перестает звенеть, светодиод соответствующей комнаты перестает мигать, светодиод «Тревога» гаснет.

Для точного определения типа тревожной ситуации нужно нажать одну из кнопок «Пожар» или «Взлом», при нажатии на которую загораются светодиодные индикаторы тех помещений, в которых обнаружен пожар или взлом.

После отжатия соответствующей кнопки постановки комнаты под охрану светодиод соответствующей комнаты гаснет, и устройство снова находится в режиме опроса датчиков, кнопок управления отображением и кнопок постановки под охрану.