ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ

«ИНЖЕНЕР УМНЫХ СИСТЕМ»

УТВЕРЖДАЮ

Генеральный директор
ООО «ИНУМСИС»
Семенов Д.А.
«15» августа 2024 г.

ДОПОЛНИТЕЛЬНАЯ ПРОФЕССИОНАЛЬНАЯ ПРОГРАММА ПОВЫШЕНИЯ КВАЛИФИКАЦИИ «ОПЕРАЦИОННАЯ СИСТЕМА FREERTOS»

Нормативный срок обучения: 16 часов
Категория обучающихся: программисты
Разработчик программы: Мединцев В.В.

г.Москва-2024г.

СОДЕРЖАНИЕ ПРОГРАММЫ

1. ОБЩАЯ ХАРАКТЕРИСТИКА ПРОГРАММЫ

1.1. Нормативные правовые основания разработки программы

Нормативную правовую основу разработки дополнительной профессиональной программы повышения квалификации «Операционная система FreeRTOS» (далее – программа) составляют:

- Федеральный закон от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Минобрнауки России от 01.07.2013 № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам».

Программа разработана с учетом профессионального стандарта «Программист» (утв. Приказом Минтруда России от 20.07.2022 № 424н «Об утверждении профессионального стандарта «Программист»).

Содержание дополнительного профессионального образования определяется образовательной программой, разработанной и утвержденной ООО «ИНУМСИС», с учетом потребностей лица, организации, по инициативе которых осуществляется дополнительное профессиональное образование.

ООО «ИНУМСИС» осуществляет обучение по дополнительной профессиональной программе на основе договора об образовании, заключаемого со слушателем и (или) с физическим или юридическим лицом, обязующимся оплатить обучение лица, зачисляемого на обучение, либо за счет бюджетных ассигнований федерального бюджета, бюджетов субъектов Российской Федерации.

1.2. Актуальность программы

Существует целый ряд классов задач, которые должны быть выполнены за строго определённое время и не более. Среди подобных задач можно назвать: срабатывание предохранительных клапанов, катапультирование пилота из терпящего бедствие воздушного судна, срабатывание подушек безопасности у автомобиля, промышленную робототехнику и т.д. В сфере программного обеспечения также существует целый ряд задач, которые должны быть выполнены точно срок — для этого и служат операционные системы реального времени, одной из которых является FreeRTOS.

В курсе представлена актуальная информация касательно использования операционной системы реального времени (ОСРВ) FreeRTOS при разработке проектов на микроконтроллере.

Курс начинается с вводной части: погружения в специфику задач, при решении которых возникает потребность использования ОСРВ (в чем причина, и как это работает). Затем от простого к сложному идет знакомство с «менеджером задач» и созданием задачи (Task). Весь материал подкрепляется практическими примерами с использованием микроконтроллеров STM32 и средой проектирования CubeMX.

Таким образом, этот курс подойдем всем, кто желает изучить возможности от использования ОСРВ в своих задачах, а также расширить свой опыт в области FreeRTOS и заполнить имеющиеся пробелы в знаниях.

1.3. Общая трудоемкость программы: составляет 16 академических часа за весь период обучения.

1.4. Требования к обучающимся: программа предназначена для лиц, имеющих среднее профессиональное или высшее профильное образование.

Категория обучающихся – программисты.

1.5. Форма обучения – заочная, исключительно с применением дистанционных образовательных технологий и электронного обучения.

1.6. Итоговый документ- удостоверение о повышении квалификации.

2. ЦЕЛЬ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

2.1. Цель освоения программы - реализация программы повышения квалификации направлена на повышение профессионального уровня в рамках имеющейся квалификации.

2.2. Планируемые результаты обучения

Перечень профессиональных компетенций в рамках имеющейся квалификации, качественное изменение которых осуществляется в результате обучения:

ПК 3.1. Способность осуществлять разработку процедур интеграции программных модулей.

ПК 3.2. Способность осуществлять интеграции программных модулей и компонентов и проверки работоспособности выпусков программного продукта.

В результате освоения программы повышения квалификации слушатель должен знать:

- основные понятия, функции, состав и принципы работы операционной системы FreeRTOS;
- особенности построения и функционирования семейств операционных систем FreeRTOS;
- принципы управления ресурсами в операционной системе FreeRTOS;
- основные задачи администрирования и способы их выполнения в изучаемой операционной системе FreeRTOS.

В результате освоения программы повышения квалификации слушатель должен уметь:

- управлять параметрами загрузки операционной системы FreeRTOS;
- выполнять конфигурирование аппаратных устройств;
- управлять учетными записями;
- настраивать параметры рабочей среды пользователя;
- управлять дисками и файловыми системами настраивать сетевые параметры;
- управлять разделением ресурсов в локальной сети.

В результате освоения программы повышения квалификации слушатель должен овладеть навыками:

- практической деятельности в операционной системе FreeRTOS.

3. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Начало занятий по мере комплектования учебных групп в течение всего календарного года.

Продолжительность обучения – 4 рабочих дня, продолжительность занятий в день не более 4 часов.

№ пп	Наименование модуля программы	Всего часов	Период обучения (учебные дни)
1.	Модуль 1. Операционная система FreeRTOS	15	1-4 день обучения
	Итоговая аттестация	1	4 день обучения
	Итого	**16**	

4. УЧЕБНЫЙ ПЛАН

№ пп	Наименование модуля программы	Всего часов	В том числе в СДО			Вид контроля/ Форма контроля
			ТЗ	ПЗ	СРС	
1.	Операционная система FreeRTOS	15	9	4	2	Промежуточный контроль Зачет
2.	Итоговая аттестация	1	-	1	-	Итоговый контроль Экзамен
	Итого	**16**	**9**	**5**	**2**	

ТЗ- теоретические занятия, ПЗ – практические занятия, СРС- самостоятельная работа слушателя, СДО- система дистанционного обучения

5. РАБОЧАЯ ПРОГРАММА УЧЕБНОГО МОДУЛЯ

5.1. Рабочая программа учебного модуля 1. «Операционная система FreeRTOS»

Учебно-тематические план рабочей программы учебного модуля 1. «Операционная система FreeRTOS»

№ пп	Наименование темы	Всего часов	В том числе		
			ТЗ	ПЗ	СРС
1.	Введение во FreeRTOS. Операционная система реального времени FreeRTOS. Лицензирование, состав, интеграция в проект.	2	1	0,5	0,5
2.	Управление задачами. Состояния задач. Приоритеты. Алгоритмы планирования.	3	2	0,5	0,5
3.	Управление очередями. Программные таймеры.	1,5	1	0,5	-
4.	Таймеры и управление кучей. Управление памятью. Программные таймеры.	1,5	1	0,5	-
5.	Ресурсы и прерывания. Двоичные и счетные семафоры. События. Группы событий. Нотификация.	2	1	0,5	0,5
6.	Оценка производительности. Отложенная обработка прерываний.	2	1	0,5	0,5
7.	Снижение энергопотребления. Управление питанием.	1,5	1	0,5	-
8.	Сложные случаи. Макросы. Стиль кода.	1	1	-	-
9.	Промежуточная аттестация	0,5	-	0,5	
	Итого	**15**	**9**	**4**	**2**

3. Содержание рабочей программы учебного модуля 1. «Операционная система FreeRTOS»

Тема 1.1. Введение во FreeRTOS. Операционная система реального времени FreeRTOS. Лицензирование, состав, интеграция в проект.

Теоретические занятия:

Операционная система FreeRTOS.

Многозадачность. Планирование. Переключение контекста. Приложения в реальном времени. Планирование в реальном времени. Структура кода. Создание собственного приложения. Конфигурационный файл.

Почему мы выбрали FreeRTOS.

Создание проекта. Создание задач.

Выводы.

Типы данных и именование функций. Типы данных. Имена переменных. Имена функций.

Практические занятия:

Домашнее задание. В среде разработки STM32CubeIDE создайте проект для имеющейся у вас в наличии платы Nucleo. Определите порты ввода-вывода к которым подключены светодиод и кнопка. Если светодиод или кнопка отсутствуют на имеющейся у вас в наличии плате, подключите внешние.

Повторите пример, разобранный в процессе занятия, организовав управление частотой переключения светодиода. Для этого создайте 2 независимые задачи.

В качестве результата выполнения домашнего задания представьте код файла main.c или freertos.c (в зависимости от компоновки созданного проекта).

Готовое задание загрузить и отправить на проверку.

Самостоятельная работа слушателя по теме:

1. FreeRTOS_Real_Tutorial_Guide.pdf
2. FreeRTOS_Reference_Manual_V10.0.0.pdf
3. RTOS.xls
4. Опрос катушек.drawio

Тема 1.2. Управление задачами. Состояния задач. Приоритеты. Алгоритмы планирования.

Теоретические занятия:

Задачи.

Состояния задач. Приоритеты задач. Реализация задачи. Задача простоя. Перехватчик простоя.

Сопрограммы.

Состояния сопрограмм. Приоритет сопрограмм. Реализация сопрограмм. Планировщик и сопрограммы. Совместное использование задач и сопрограмм.

Ограничения.

Совместное использование стека. Использование операторов switch. Пример использования сопрограмм . Мигание светодиодом.

Планировщик сопрограмм. Создание сопрограммы и запуск диспетчера.

Расширенный пример: Использование параметра индекс.

Выводы раздела.

Характеристики «задачи». Характеристики «сопрограммы».

API в примерах.

xTaskAbortDelay(). xTaskCreate(). vTaskDelay(). vTaskDelayUntil(). vTaskDelete(). taskENTER_CRITICAL(). xTaskGetCurrentTaskHandle(). uxTaskGetNumberOfTasks(). vTaskGetRunTimeStats(). uxTaskGetStackHighWaterMark(). xTaskGetTickCount(). vTaskStartScheduler().

Практические занятия:

Домашнее задание. Создайте проект в среде STM32CubeIDE. Создайте несколько задач с разным приоритетом. В теле задач реализуйте “нагрузочный код” (например, создайте нагрузку за счет длительных циклов операции __NOP() или проведения затратных по времени математических вычислений).

Соберите статистику о времени, выделенном каждой из задач за счет использования внутренних счётчиков.

В качестве результата выполнения домашнего задания представьте исходный код (файлы main.c или freertos.c в зависимости от компоновки проекта) и ваши выводы о причинах замеченных явлений.

Готовое задание загрузить и отправить на проверку.

Самостоятельная работа слушателя по теме:

1. FreeRTOS functions.docx.

Тема 1.3. Управление очередями. Программные таймеры.

Теоретические занятия:

Очереди. Характеристики очереди. Доступ. Блокировка. Использование очередей. Блокировка очередей.

Функции API. xQueueCreate(). vQueueDelete(). uxQueueMessagesWaiting(). xQueuePeek(). xQueueReceive(). xQueueReset(). xQueueSend(), xQueueSendToFront(), xQueueSendToBack().

Практические занятия:

Домашнее задание. Создайте проект в среде STM32CubeIDE. Создайте две задачи:

Задача управления светодиодом;

Задача, которая опрашивает кнопку.

Организуйте отправку данных из задачи обслуживания кнопки в задачу отображения при помощи очереди.

В качестве результата выполнения домашнего задания представьте исходный код (файлы main.c или freertos.c в зависимости от компоновки проекта).

Готовое задание загрузить и отправить на проверку.

Тема 1.4. Таймеры и управление кучей. Управление памятью. Программные таймеры.

Теоретические занятия:

Управление кучей.

Распределение динамической памяти и его значение для FreeRTOS.

Heap_1. Heap_2. Heap_3. Heap_4. Heap_5.

Отслеживание размера кучи. Программные таймеры. Функция обратного вызова.

Период программного таймера. Состояния таймера.

API функции. xTimerChangePeriod(). xTimerCreate(). xTimerDelete(). xTimerReset(). xTimerStart(). xTimerStop().

Практические занятия:

Домашнее задание.

Создайте проект в среде STM32CubeIDE, в проекте реализуйте программный таймер для управления светодиодом. Опробуйте использование программного таймера для обновления сторожевого таймера как признак работоспособности операционной системы.

В качестве результата выполнения домашнего задания представьте исходный код (файлы main.c или freertos.c в зависимости от компоновки проекта).

Готовое задание загрузить и отправить на проверку.

Тема 1.5. Ресурсы и прерывания. Двоичные и счетные семафоры. События. Группы событий. Нотификация.

Теоретические занятия:

Ресурсы и прерывания.

Прерывания. Приоритет задачи и приоритет прерывания. Функции API для использования в обработчике прерывания. Преимущества.

Недостатки. Параметр xHigherPriorityTaskWoken. Переключение контекста.

Макросы portYIELD_FROM_ISR () и portEND_SWITCHING_ISR (). Отложенная обработка прерывания. Совместный доступ к ресурсам.

Взаимное исключение. Критические разделы и приостановка планировщика. Приостановка (или блокировка) планировщика. Совместное использование ресурсов. Мэджик код.

Практические занятия:

Домашнее задание.

В дополнительных материалах к данному уроку имеется руководство пользователя и инструкция к прибору «Модули мониторинга и информирования серии GTex». Также, вместо приложенных документов вы можете использовать формальное описание любого другого прибора из сферы автоматизации.

В качестве домашнего задания к данному уроку предложите свою реализацию разделения кода проекта на задачи, очереди, прерывания. Кратко опишите функции каждой из задач и взаимодействие между ними.

Результатом выполнения домашнего задания является PDF документ с описанием функционала предлагаемых вами задач, очередей, обработчиков прерываний для реализации требуемого функционала.

Вопросы к практическому заданию:

1. Сколько будет задач?
2. Какие функции вы возложите на эти задачи?
3. GSM-модем подключён по интерфейсу UART. Как мы будем взаимодействовать с модемом?
4. Потребуется ли использование очередей?

Готовое задание загрузить и отправить на проверку.

Самостоятельная работа слушателя по теме:

1. GTex 3.pdf
2. Table of config.pdf

Тема 1.6. События, нотификация, семафоры.

Теоретические занятия:

Семафоры. Двоичные семафоры, используемые для синхронизации.

xSemaphoreCreateBinary(). xSemaphoreTake(). xSemaphoreGiveFromISR().

Счётные семафоры. xSemaphoreCreateCounting().

Пример синхронизации задачи и обработчика прерывания.

События. Характеристики группы событий. Группы событий, флаги событий и биты событий.

EventBits_t. Мультизадачный доступ.

Практический пример использования группы событий.

Управление событиями с использованием групп событий.

xEventGroupSetBits (). xEventGroupSetBitsFromISR(). xEventGroupWaitBits().

Синхронизация задач с помощью группы событий. xEventGroupSync().

Нотификация. Общение через посреднические объекты.

Уведомления. Преимущества и ограничения. Ограничения уведомлений. Использование уведомлений.

xTaskNotifyGive (). vTaskNotifyGiveFromISR (). ulTaskNotifyTake (). xTaskNotify () и xTaskNotifyFromISR (). xTaskNotifyWait().

Практические занятия:

Домашнее задание.

Достаточно часто в промышленной автоматике требуется реализация различной логики управления.

К примеру, выход активируется, если состояние входов соответствует той или иной маске событий.

Также часто требуется формировать управляющий сигнал с задержкой.

Используйте 4 кнопки, подключённые к плате Nucleo в качестве входов, а 4 светодиода — в качестве выходов. Используя механизмы операционной системы FreeRTOS, реализуйте программу, устанавливающую состояние выходов в зависимости от того, соответствует ли состояние входов запрограммированной «маске».

Пример логики:

Выход 1 должен срабатывать при событиях 1 и 4 входов;

Выход 2 при событии 1 входа;

Выход 3 при событии 3 и 4 входов;

Выход 4 при событии любого входа.

Сделайте маску событий изменяемой.

В качестве результата выполнения домашней работы представьте исходный код (файлы main.c или freertos.c в зависимости от компоновки проекта).

Готовое задание загрузить и отправить на проверку.

Самостоятельная работа слушателя по теме:

1. scmRTOS.ru.pdf.

Тема 1.7. Снижение энергопотребления. Управление питанием.

Теоретические занятия:

Снижение энергопотребления.

Режим Sleep. Режим Stop. Режим Standby.

FreeRTOS и режимы энергопотребления.

Макрос portSUPPRESS_TICKS_AND_SLEEP (). TICKLESS_IDLE в генераторе кода.

Реализация portSUPPRESS_TICKS_AND_SLEEP (). vTaskStepTick. xTaskCatchUpTicks. eTaskConfirmSleepModeStatus.

SysTick и частота ядра.

Генерация тактов от источника, отличного от SysTick.

Использование специфических функций с низким энергопотреблением.

Демонстрационный пример.

Выводы.

Практические занятия:

Домашнее задание.

Дополните ранее выполненный проект следующими элементами:

Подключите 4 кнопки в качестве имитации входных сигналов;

Подключите 4 светодиодных индикатора (или дисплей, на ваш выбор) в качестве индикации дискретных выходных сигналов;

Перепишите код проекта, обеспечивающий функционирование платы Nucleo с использованием операционной системы FreeRTOS;

Дополните страницу, отображаемую созданным WEB-сервером, механизмом ввода маски, логически соединяющей входы и выходы платы.

Например:

Выход 1 должен срабатывать при событиях 1 и 4 входов;

Выход 2 при событии 1 входа;

Выход 3 при событии 3 и 4 входов;

Выход 4 при событии любого входа.

Критерии оценки:

В ходе проверки работы будет оцениваться код, разработанный для отладочной платы Nucleo.

Ожидается что в процессе выполнения итоговой работы вы продемонстрируете знания, полученные на курсе по операционной системе FreeRTOS, а именно:

Адекватно задаче спланируете задачи.

Используете средства обмена информацией между задачами (очереди).

Готовое задание загрузить и отправить на проверку.

Тема 1.8. Сложные случаи. Макросы. Стиль кода.

Теоретические занятия:

Примеры. Получение данных из нескольких источников. Разные типы и длины данных в очереди.

Перехватчик запуска задачи демона. Откладывание работы на демона. xTimerPendFunctionCallFromISR().

Функционал для разработчиков. configASSERT().

Статистика времени выполнения. uxTaskGetSystemState(). vTaskList(). vTaskGetRunTimeStats(). xPortGetFreeHeapSize(). xPortGetMinimumEverFreeHeapSize().

Макросы-ловушки. Определение макросов трассировки.

Использование макросов.

4. Организационно-педагогические условия реализации рабочей программы учебного модуля 1. «Операционная система FreeRTOS»

Реализация рабочей программы учебного модуля должна обеспечить приобретение слушателями знаний и умений, необходимых для работы в операционной системе FreeRTOS». Выбор методов обучения для каждого занятия определяется преподавателем в соответствии с составом и уровнем подготовленности обучающихся, степенью сложности излагаемого материала, наличием и состоянием технических возможностей обучающихся.

Теоретические занятия проводятся с целью изучения нового учебного материала. Изложение материала необходимо вести в форме, доступной для понимания обучающихся.

Практическое занятие предусматривает выполнение практического задания, которое включает в себя отработку умений и навыков по всем темам модуля. Практическое задание выполняется после изучения каждой темы модуля (полностью).

Цель практических заданий – дать возможность обучающемуся проверить понимание теоретических положений изучаемого курса, знать основные понятия, классификации и т.п.

В процессе организации работы большое значение имеют самостоятельные работы слушателей. Самостоятельная работа предназначена не только для овладения конкретной темой программы, но и для формирования навыков самостоятельной работы вообще (способности самостоятельно решать проблемы, находить конструктивные решения).

Материально-техническое обеспечение рабочей программы учебного модуля:

Дополнительная профессиональная программа повышения квалификации реализуется полностью с применением дистанционных образовательных технологий (ДОТ). Для эффективного внедрения ДОТ и использования электронных образовательных ресурсов имеется качественный доступ педагогических работников и обучающихся к информационно-телекоммуникационной сети Интернет. Услуга подключения к сети Интернет предоставляется в режиме 24 часа в сутки 7 дней в неделю без учета объемов потребляемого трафика. Для использования ДОТ педагогическому работнику предоставляется свободный доступ к средствам информационных и коммуникационных технологий.

Рабочее место педагогического работника:

№ п/п	Наименование	Кол-во(шт.)
1.	Системный блок Intel Core i3, 2, 3400 МГц	1
2.	Монитор Philips	1
3.	Клавиатура	1
4.	Мышь проводная Genius	1
5.	Наушники	1
6.	Веб-камера	1

7.	Ноутбук	1
8.	Стол преподавателя	1
9.	Стул преподавателя	1

В состав программно-аппаратных комплексов установлено программное обеспечение, необходимое для осуществления учебного процесса: общего назначения (операционная система (операционные системы), офисные приложения, средства обеспечения информационной безопасности, архиваторы, графический, видео- и аудио-редакторы).

Формирование информационной среды осуществляется с помощью программной системы дистанционного обучения, расположенном на сайте, открытом для свободного ознакомления, публично доступном для физических и юридических лиц.

С помощью системы дистанционного обучения (далее - СДО):

- преподаватели разрабатывают и размещают содержательный контент; планирует свою педагогическую деятельность: выбирает из имеющихся или создает нужные для обучающихся ресурсы и задания;

- педагогические работники и обучающиеся обеспечиваются доступом к полной и достоверной информации о ходе учебного процесса, промежуточных и итоговых результатах благодаря автоматическому фиксированию указанных позиций в информационной среде;

- обучающиеся выполняют задания, предусмотренные образовательной программой, при необходимости имеют возможность обратиться к педагогическим работникам за помощью;

- все результаты обучения сохраняются в информационной среде, на их основании формируются портфолио обучающихся и педагогических работников.

Система дистанционного обучения содержит образовательный контент доступный для Пользователей по интернет-адресу СДО с использованием настольных компьютеров или ноутбуков. Образовательная организация осуществляет электронную систему учета контингента обучающихся, путем регистрации обучающихся в системе дистанционного обучения, учет по прохождению обучающимися процесса обучения, формирует статистический отчет и данные о прохождении промежуточной и итоговой аттестации.

Требования к рабочим местам обучающихся:

Наличие ПК с Операционной системой Windows 7, Windows 8, Windows 8.1,Windows 10 и более поздней версии, а также MacOS 9 и выше или Ubuntu 18+;

Оперативная память не менее 4 Гб;

Монитор с разрешением от 1366x768 и выше, 256 цветов;

Установленный интернет-браузер на базе Chromium (Chrome 64.0 или выше, Яндекс.Браузер 17.6.1 или выше; Opera версии 60.x и выше);

Обеспечение устойчивого и качественного доступа обучаемых в информационно-коммуникационную сеть Интернет на период обучения;

Наличие личного E-mail для каждого пользователя или доступ к мессенджеру Telegram со своих рабочих компьютеров или личных мобильных устройств.

5. Требования к кадровому обеспечению образовательного процесса рабочей программы учебного модуля 1. «Операционная система FreeRTOS»

Реализация рабочей программы учебного модуля обеспечивается научно-педагогическими работниками организации, а также лицами, привлекаемыми к реализации программы на иных законных основаниях.

Квалификация руководящих и научно-педагогических работников организации соответствует квалификационным характеристикам, установленным в Едином квалификационном справочнике должностей руководителей, специалистов и служащих,

раздел «Квалификационные характеристики должностей руководителей и специалистов высшего профессионального и дополнительного профессионального образования», утвержденном приказом Министерства здравоохранения и социального развития Российской Федерации от 11 января 2011 г. №1н.

6. Учебно-методическое и информационное обеспечение учебного модуля 1. «Операционная система FreeRTOS»

Основная литература, использованная при разработке образовательной программы:

1.Мединцев Владимир М42 Операционные системы микроконтроллеров: На примере операционной системы реального времени FreeRTOS/ Владимир Мединцев. 16. м.]: Издательские решения, 2023. - 280 с. ISBN 978-5-0060-5990-0

2. The FreeRTOS Reference manual / 2017 г. – URL: https://www.freertos.org/Documentation/FreeRTOS_Reference_Manual_V10.0.0.pdf – Текст: электронный.

3. Гордеев, А.В. Операционные системы/ А.В. Гордеев – 2-е изд. – СПб.: Питер, 2004.

4. Курниц, А. FreeRTOS – операционная система для микроконтроллеров / А. Курниц // Компоненты и технологии – 2011 г.

5. FreeRTOS. Краткое описание / URL: http://easyelectronics.ru/freertos_manual.html– Текст: электронный.

Дополнительная литература:

1. Операционные системы реального времени / И.Б. Бурдонов, А.С. Косачев, В.Н. Пономаренко – Москва: ИСП РАН, 2006 г. – URL: http://citforum.ru/operating_systems/rtos/1.shtml – Текст: электронный.

2. Операционные системы реального времени и Windows NT / Е. Хухлаев – 1997 г. – URL: https://www.osp.ru/os/1997/05/179265 – Текст: электронный.

3. Most Popular Open-source RTOS Comparison for Embedded Systems / URL: https://scienceprog.com/most-popular-open-source-rtos-comparison-for-embedded-systems/ – Текст: электронный.

4. Обзор ОСРВ FX-RTOS / URL: https://www.eremex.ru/products/fx-rtos/ – Текст: электронный.

5. Операционная система Eremex FX-RTOS Руководство пользователя / Версия 3.0 – 2018г. – URL: https://www.eremex.ru/upload/iblock/1ee/FXRTOS_GUIDE.pdf – Текст: электронный.

6. ФОРМЫ АТТЕСТАЦИИ

С целью контроля и оценки результатов подготовки и учета индивидуальных образовательных достижений, обучающихся применяются следующие виды контроля: промежуточная и итоговая аттестация.

6.1. Промежуточная аттестация

Целью проведения промежуточной аттестации является объективное установление фактического уровня освоения и достижения результатов учебного модуля образовательной программы. Промежуточный контроль заканчивается зачетом в форме тестирования с присвоением каждому обучающемуся результата «зачет / незачет».

Оценивание ответа на промежуточной аттестации осуществляется следующим образом:

В тесте содержится 19 вопросов.

В каждом вопросе представлено 3 варианта ответа. Необходимо выбрать один верный.

За каждый верный ответ начисляется по 1 баллу. Для успешного прохождения теста необходимо набрать не менее 80% правильных ответов (16 баллов и выше).

6.2. Итоговая аттестация

Освоение дополнительной профессиональной программы повышения квалификации завершается итоговой аттестацией обучающихся в форме экзамена (тестирование) с присвоением каждому обучающемуся результата «зачет / незачет».

Оценивание ответа на итоговой аттестации осуществляется следующим образом:

Оценки теста:

- оценка «отлично» выставляется слушателю, если задание выполнено на 91-100%;
- оценка «хорошо» выставляется слушателю, если задание выполнено на 81-90%;
- оценка «удовлетворительно» выставляется слушателю, если задание выполнено на 70-80%;
- оценка «неудовлетворительно» выставляется слушателю, если задания выполнено менее чем на 70%.

К итоговой аттестации допускается обучающийся, не имеющий задолженности и в полном объеме выполнивший учебный план по дополнительной образовательной программе.

Итоговая аттестация не может быть заменена оценкой уровня знаний на основе промежуточной аттестации обучающихся.

Критерии оценивания итоговой аттестации:

Обучающиеся, освоившие дополнительную профессиональную программу повышения квалификации и прошедшие итоговую аттестацию, получают удостоверение о повышении квалификации установленного образца.

Лицам, не прошедшим итоговой аттестации или получившим на итоговой аттестации неудовлетворительные результаты, а также лица освоившим часть дополнительной профессиональной программы и (или) отчисленным из организации, выдается справка об обучении или о периоде обучения по образцу, самостоятельно устанавливаемому организацией.

Документ о квалификации выдается на бланке, образец которого самостоятельно устанавливается организацией.

7.ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

7.1. Перечень вопросов для промежуточной аттестации

1. Как называется ядро операционной системы FreeRTOS?

a) KernelRTOS
b) FreeCore
c) FreeRTOS

2. Какие два основных параметра определяются при создании задачи в FreeRTOS?

a) Частота и период
b) Приоритет и стек
c) Имя и цвет

3. Что представляет собой «планировщик» FreeRTOS?

a) Часть ядра операционной системы, определяющая то, какая задача получит доступ к ядру микроконтроллера
b) Инструмент для анализа кода
c) Функция для определения типа данных

4. Что такое переключение контекста во FreeRTOS?

a) Процесс установки операционной системы на устройство
b) Процесс сохранения состояния одной задачи и восстановления состояния другой задачи
c) Процесс обновления ядра операционной системы

5. Что такое «управление памятью» в контексте операционной системы FreeRTOS?

a) Процесс обновления программных драйверов
b) Механизм для определения объема оперативной памяти на устройстве
c) Управление распределением и освобождением динамической памяти для задач и объектов FreeRTOS

6. Какие две модели распределения памяти поддерживаются FreeRTOS?

a) Жесткое и мягкое управление
b) Автоматическое и ручное распределение
c) Статическое и динамическое выделение

7. Что такое «куча» (heap) FreeRTOS?

a) Структура данных для хранения общих ресурсов
b) Область памяти, предназначенная для динамического распределения
c) Отдельное ядро для обработки задач

8. Какие функции используются для выделения и освобождения динамической памяти в FreeRTOS?

a) pvMalloc и vFree
b) pvPortMalloc и vPortFree
c) xMalloc и xFree

9. Какой компонент FreeRTOS отвечает за планирование и переключение задач?

a) TaskScheduler
b) ContextManager
c) Планировщик ядра (Kernel Scheduler)

10. Какая функция используется для приостановки выполнения задачи в FreeRTOS?

a) vTaskSuspend
b) vTaskPause
c) vTaskStop

11. Каким образом можно изменить приоритет задачи во время выполнения в FreeRTOS?

a) Приоритеты задач не могут быть изменены после создания
b) С помощью функции vTaskPrioritySet
c) Только администратор может изменить приоритеты

12. Какая функция используется для удаления задачи в FreeRTOS?

a) vTaskTerminate
b) vTaskDelete
c) vTaskStop

13. Что такое «очередь» (queue) в операционной системе FreeRTOS?

a) Буфер для хранения файловых данных
b) Интерфейс для взаимодействия с сетевыми устройствами
c) Механизм для обмена данными между задачами или прерываниями

14. Какой тип данных может быть помещен в очередь FreeRTOS?

a) Только целые числа
b) Только строки текста
c) Любой пользовательский тип данных

15. Как задачи могут поместить данные в очередь в FreeRTOS?

a) Задачи могут обращаться к памяти очереди напрямую
b) Задачи используют функцию xQueueSend
c) Очереди могут быть заполнены только прерываниями

16. Как создать программный таймер в FreeRTOS?

a) С помощью функции xTimerCreate
b) С помощью функции vTimerInit
c) Программные таймеры создаются автоматически

17. Какая функция используется для запуска программного таймера в FreeRTOS?

a) xTimerStart
b) xTimerRun
c) vTimerBegin

18. Какие параметры определяются при создании программного таймера?

a) Период и частота
b) Приоритет и длительность
c) Период и колбэк-функция

19. Что происходит, если очередь заполнена, и задача пытается поместить данные в нее в FreeRTOS?

a) Данные будут потеряны
b) Задача заблокируется до тех пор, пока место не освободится
c) Система автоматически увеличит размер очереди

7.2. Перечень вопросов для итоговой аттестации

1. Какая конструкция в FreeRTOS обеспечивает потокобезопасность при доступе к разделяемым ресурсам?

a) Механизм защиты (shielding mechanism)
b) Критическая секция (critical section)
c) Защитный барьер (protective barrier)

2. Какая функция используется для выхода из критической секции в FreeRTOS?

a) vCriticalExit
b) xSemaphoreGive
c) taskEXIT_CRITICAL

3. Что такое «семафор» (semaphore) в операционной системе FreeRTOS?

a) Механизм для регистрации событий пользовательского интерфейса
b) Инструмент для отладки многозадачных приложений
c) Механизм для синхронизации между задачами и прерываниями

4. Какой компонент FreeRTOS отвечает за управление очередями?

a) Очереди управляются автоматически
b) QueueManager
c) QueueController

5. Какой компонент FreeRTOS отвечает за управление семафорами?

a) SemaphoreManager
b) SemaphoreController
c) Семафоры управляются автоматически

6. Как осуществляется остановка программного таймера?

a) xTimerPause
b) xTimerStop
c) vTimerHalt

7. Что произойдет, когда истечет время, установленное для программного таймера?

a) Задача, связанная с таймером, будет приостановлена
b) Зарегистрируется событие, и колбэк-функция таймера будет вызвана
c) Операционная система автоматически перезагрузит таймер

8. Какая функция используется для входа в критическую секцию в FreeRTOS?

a) vCriticalEnter
b) xSemaphoreTake
c) taskENTER_CRITICAL

9. Что такое «энергопотребление» в контексте операционной системы FreeRTOS?

a) Количество выделенной оперативной памяти.
b) Общее количество потребляемой электроэнергии системой.
c) Пропускная способность сетевых интерфейсов.

10. Какая основная цель снижения энергопотребления?

a) Увеличение скорости выполнения задач.
b) Максимизация использования процессорного времени.
c) Эффективное использование энергии для увеличения времени автономной работы устройства.

11. Какая функция используется для установки битов в группе событий в FreeRTOS?

a) vEventGroupSetBits
b) xEventGroupSet
c) vEventGroupSignal

12. Что произойдет, если задача попытается установить биты, которые уже установлены в группе событий?

a) Биты будут переключены на противоположное состояние
b) Ничего не произойдет, биты останутся неизменными
c) Задача будет заблокирована до освобождения битов

13. Как можно использовать макрос для создания пользовательской функции, возвращающей максимальное значение?

a) MAX_VALUE()
b) #define MAX_VALUE(a,b) ((a) > (b) ? (a) : (b))
c) vMaxValue(a,b)

14. Могут ли прерывания отправлять уведомления в FreeRTOS?

a) Да, прерывания могут использовать функцию xTaskNotifyFromISR()
b) Нет, уведомления могут отправлять только задачи
c) Прерывания могут только уведомлять о событиях, но не отправлять уведомления

15. Как создать мьютекс в FreeRTOS?

a) С помощью функции xSemaphoreCreateMutex
b) С помощью функции vMutexInit
c) Мьютексы создаются автоматически

16. Какая функция используется для захвата (блокирования) мьютекса в FreeRTOS?

a) vMutexCapture
b) xSemaphoreTake
c) vMutexLock

17. Что такое «колбэк-функция» (callback function) программного таймера в FreeRTOS?

a) Функция, которая вызывается в случае переполнения таймера
b) Функция, которая вызывается по истечении времени таймера
c) Функция, которая вызывается только по запросу пользователя

18. Какой макрос используется для определения максимального приоритета задачи в FreeRTOS?

a) taskMAX_PRIORITY
b) configMAX_PRIORITIES
c) vTaskMaxPriority

19. Какая функция используется для отправки уведомления?

a) xTaskNotify
b) vNotificationSend
c) vNotifyPost

20. Какая функция используется для приема уведомления задачей в FreeRTOS?

a) vNotificationReceive
b) xTaskNotifyWait
c) vNotifyAwait

21. Как осуществляется очистка (сброс) битов в группе событий?

a) xEventGroupClearBits
b) xEventGroupClear
c) vEventGroupReset

22. Как можно проверить состояние определенных битов в группе событий в FreeRTOS?

a) С помощью функции vEventGroupTestBits
b) С помощью функции xEventGroupGetBits
c) Состояние битов не может быть проверено в группе событий

23. Как создать макрос в FreeRTOS?

a) С помощью функции vMacroCreate
b) С помощью функции #define
c) Макросы создаются автоматически

24. Как создать группу событий в FreeRTOS?

a) С помощью функции xEventGroupCreate
b) С помощью функции vGroupInit
c) Группы событий создаются автоматически

25. Как задачи могут захватить (заблокировать) двоичный семафор в FreeRTOS?

a) Функция xSemaphoreLock
b) Функция xSemaphoreTake
c) Функция xSemaphoreAcquire

8. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

Методические рекомендации для слушателей

Терминология

Задача - Во FreeRTOS каждый поток выполнения называется «задачей». В отличии от сообщества Linux/Unix систем в embedded нет единого мнения относительно терминологии. Это будет особенно заметно если после прочтения официальной документации FreeRTOS вы прочитаете документацию Microsoft на ThreadX. Однако, лично я предпочитаю «задачу» «потоку», поскольку в некоторых областях применения поток может иметь более конкретное значение.

API (Application Programming Interface) - интерфейс программирования приложений. Это совокупность инструментов и функций, которые представляют собой интерфейс для создания новых приложений. И это замечательное определение из мира ПК. В нашем случае мы имеем дело с кодом написанным на языке Си для микроконтроллеров. И операционная система, и драйвера периферийных устройств (HAL), и создаваемое прикладное программное обеспечение будут выполняться в едином адресном пространстве микроконтроллера. По этой причине понятие API сводится к набору функций, определенных в коде операционной системе и предназначенных для взаимодействия ОС, и прикладного программного обеспечения.

RTOS - Операционная система реального времени (ОСРВ, англ. realtime operating system) — тип специализированной операционной системы, основное назначение которой — предоставление необходимого и достаточного набора функций для проектирования, разработки и функционирования систем реального времени на конкретном аппаратном оборудовании.

Портирование – адаптация некоторой программы или её части, чтобы она работала в другой среде, отличающейся от той среды, под которую она была изначально написана с максимальным сохранением её пользовательских свойств. Процесс портирования также называют портированием или переносом, а результат — портом.

Но в любом случае главной задачей при портировании является сохранение привычных пользователю интерфейса и приёмов работы с программой. Добавление новых или удаление части имеющихся свойств при портировании программных продуктов не допускается.

Необходимость в портировании возникает обычно из-за различий в системе команд процессора, различий между способами взаимодействия операционной системы и программ, принципиальных различий в архитектуре вычислительных систем, либо по причине некоторых несовместимостей или даже полного отсутствия используемого языка программирования в целевом окружении.