Средства диспетчеризации

Тема: Исследование примитивных диспетчеров

Цели работы:

Изучение механизмов создания и управления процессами в кооперативном и вытесняющим режиме;

Изучение внутреннего устройства механизмов кооперативной и вытесняющей диспетчеризаций;

**Краткие теоретические сведения**

**Диспетчеризация процессора** – предоставление всем процессам в системе по очереди в определенном порядке квантов процессорного времени.

**Главной целью** диспетчеризации является максимальная и равномерная загрузка процессора.

Работа любого процесса в системе представляется как последовательность чередований фаз активности процессора и активности ввода-вывода. Частота периодов активности процессора обратно пропорциональна их длительности.

**Операционная система (ОС)** — это комплекс программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

Операционная система выполняет следующие основные функции, связанные с управлением задачами:

- создание и удаление задач;

- планирование процессов и диспетчеризация задач;

- синхронизация задач, обеспечение их средствами коммуникации.

**Планировщик** – компонента ОС, планирующая выделение квантов времени процессам по определенной стратегии. Различаются стратегии с **прерыванием процессов** **(вытесняющие)** (когда при вводе нового более короткого или более приоритетного процесса в систему текущий процесс прерывается) и **без прерывания процессов (кооперативные)**.

**Диспетчер** – компонента ОС, выполняющая само переключение процессора с одного процесса на другой. Время, которое на это требуется, называется скрытой активностью (латентностью) диспетчера и должно быть минимизировано.

**Диспетчеризация -** задача динамического (краткосрочного) планирования процессов для наиболее эффективного распределения ресурсов, возникающих практически при каждом событии.

**Долгосрочный планировщик** решает, какой из процессов, находящихся во входной очереди, должен быть переведен в очередь готовых процессов в случае освобождения ресурсов памяти. Он выбирает процессы из входной очереди задач с целью создания неоднородной мультипрограммной смеси. Это означает, что в очереди готовых к выполнению процессов должны находиться – в разной пропорции – как процессы, ориентированные на ввод/вывод, так и процессы, ориентированные на преимущественную работу с центральным процессором.

**Краткосрочный планировщик решает**, какая из задач, находящихся в очереди готовых к выполнению, должна быть передана на исполнение. В **большинстве современных операционных систем** долгосрочный планировщик отсутствует.

**Основные критерии диспетчеризации**:

1. Использование процессора – максимизируется;
2. пропускная способность системы – максимизируется;
3. среднее время обработки одного процесса – минимизируется;
4. среднее время ожидания одним процессом – минимизируется;
5. среднее время ответа системы – минимизируется;
6. равномерность загрузки процессора – максимизируется.

Для иллюстрации стратегий диспетчеризации используются диаграммы Ганта с изображением временной линейки, имен и периодов активности каждого процесса.

Различают два больших класса стратегий обслуживания – **бесприоритетные** и **приоритетные**. Приоритет, присвоенный задаче, **может быть постоянным** и приоритет **может изменяться** в процессе её решения. Классификация стратегии диспетчеризации приведена на рисунке 1.

* **Бесприоритетное обслуживание**. Выбор задачи производится в некотором заранее установленном порядке без учета их относительной важности и времени обслуживания.
* **Приоритетное обслуживание**. При обслуживании отдельных задач, задачам предоставляется преимущественное право попасть в состояние исполнения.

Классификация диспетчеров по способу распределения процессорного времени

**Диспетчеризация без перераспределения процессорного времени**, то есть не вытесняющая многозадачность (non-preemptive multitasking) (**кооперативная диспетчеризация**) – это такой способ диспетчеризации процессов, при котором активный процесс выполняется до тех пор, пока процесс полностью не отдаст управление диспетчеру задач для выбора из очереди другого, готового к выполнению процесса или потока. Дисциплины обслуживания FCFS, SJN, SRT относятся к не вытесняющим.

Рисунок 1 – Классификация дисциплин диспетчеризации

**Диспетчеризация с перераспределением процессорного времени** между задачами, то есть вытесняющая многозадачность (preemptive multitasking) – это такой способ, при котором решение о переключении процессора с выполнения одного процесса на выполнение другого процесса принимается диспетчером задач, а не самой активной задачей. При вытесняющей многозадачности механизм диспетчеризации задач целиком сосредоточен в операционной системе, и программист может писать свое приложение, не заботясь о том, как оно будет выполняться параллельно с другими задачами. При этом операционная система выполняет следующие функции:

- определяет момент снятия с выполнения текущей задачи, сохраняет её контекст в дескрипторе задачи (или в отведенном сегменте памяти),

- выбирает из очереди готовых задач следующую и запускает её на выполнение, предварительно загрузив её контекст. Дисциплина RR и многие другие, построенные на её основе, относятся к вытесняющим.

Число переключений контекста с процесса на процесс возрастает с уменьшением выделяемого кванта времени. Время оборота зависит от кванта времени более сложным образом. Программная функция сохранения и восстановления контекста процессора делается на ассемблере.

Для обработки процессов различных классов и приоритетов (например, пакетных и интерактивных) ОС создает многоуровневые аналитические очереди процессов, каждая из которых обслуживается по различным стратегиям и (или) предоставляет процессам кванты времени различного размера. Процесс при необходимости может быть переведен из одной очереди в другую.

При планировании загрузки многопроцессорных систем учитывается их симметричность или асимметричность. Планирование их загрузки гораздо более сложно. В асимметричных системах не требуется синхронизировать процессы по системным структурам данных, так как они доступны процессу только на одном процессоре.

Для систем реального времени наиболее важным является предоставление наивысших приоритетов критическим процессам реального времени, решающим основную задачу системы.

Стратегия планирования определяет, какие процессы планируются на выполнение для того, чтобы достичь поставленной цели. Известно большое количество различных стратегий выбора процесса, которому необходимо предоставить процессор. Среди них, прежде всего, можно назвать следующие стратегии:

Стратегий диспетчеризации

1. Стратегия «в порядке живой очереди» - First-Come-First-Served (FCFS) – предоставление ресурсов процессора процессам в порядке их ввода в систему, независимо от их длительности. При этом время ожидания может оказаться большим, особенно если первым в систему вводится более длительный процесс (что называется эффектом сопровождения). Ситуация, когда первыми вводятся более короткие процессы, более благоприятна.

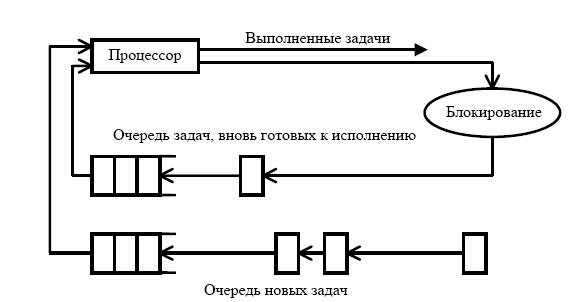


Рисунок 2 - Стратегия диспетчеризации FCFS

Образуются две очереди: одна очередь образуется из **новых задач**, а вторая очередь – из ранее выполнявшихся, но попавших в состояние ожидание. Такой подход позволяет реализовать стратегию обслуживания «по возможности заканчивать вычисления в порядке их появления». Эта дисциплина обслуживания не требует внешнего вмешательства в ход вычислений, при ней не происходит перераспределение процессорного времени.

**К достоинствам** этой стратегий, прежде всего, можно отнести простоту реализации и малые расходы системных ресурсов на формирование очереди задач.

**Недостатки.** Стратегия приводит к тому, что при увеличении загрузки вычислительной системы растет и среднее время ожидания обслуживания, причем короткие задания (требующие небольших затрат машинного времени) вынуждены ожидать столько же, сколько и трудоёмкие задания. Избежать этого недостатка позволяют дисциплины SJN и SRT.

1. Стратегий «первый с кратчайшими сроками выполнения работы / следующий с кратчайшим заданием» - Shortest Job First (SJF) / Shortest Job Next (SJN) или (Shortest Process Next) (SPN) –дисциплина планирования **без переключения**, согласно которой следующим для выполнения выбирается ожидающий процесс с минимальным оценочным рабочим временем - SRTF (Shortest-Remaining-Time-First).

Данная стратегия обеспечивает минимальное среднее время ожидания процессов и является кооперативной стратегией!

Общее название данных стратегий – «кратчайшая работа следующей» - SJN (Shortest job next) - алгоритм **краткосрочного** планирования, который выбирает для выполнения процесс с наименьшим временем выполнения.

1. Стратегия «следующий по наименьшему остающемуся времени» SRTN/SRTF - (shortest remaining time next / shortest remaining time first, следующее задание требует меньше всего времени для своего завершения**). Это вытесняющая** (с переключениями) версия стратегии SJN, где процессор выделяется ресурсы для задачи, у которой время ближе всего к завершению (оставшееся время). Применяется в системах с разделением времени.

Метод экспоненциального усреднения позволяет вычислить предсказываемую длину следующего периода активности по фактическим и предсказанным длинам предыдущих периодов активности.

Оставшееся время — это разность между временем, запрошенным пользователем (временем выполнения), и временем, которое процесс уже получил и которое измеряется системой с помощью аппаратного таймера. По принципу SRTN/SRTF первым всегда выполняется процесс, **имеющий минимальное оценочное время до завершения**, причем с учетом новых поступающих процессов. Если в соответствии с алгоритмом SPN процесс, запущенный в работу, выполняется до своего завершения, то по алгоритму SRT выполняющийся процесс **может быть прерван при поступлении нового процесса**, имеющего более короткое оценочное время работы. Чтобы механизм SRT был эффективным, опять-таки нужны достаточно точные оценки будущего, причем разработчик системы должен позаботиться о мерах против неправильного использования прикладными программистами особенностей стратегий диспетчеризации.

Механизм SRT характеризуется более высокими накладными расходами, чем SPN. Механизм SRT должен следить за текущим временем обслуживания выполняющегося задания и обрабатывать возникающи­е прерывания. Поступающие в систему небольшие процессы будут выполняться почти немедленно. Однако более длительные процессы будут иметь даже большее среднее время ожидания и большую дисперсию времени ожидания, чем в случае SPN.

Реализация принципа SRT требует, чтобы­ регистрировались истекшие временные интервалы обслуживания задач, а это при­вод­ит к увеличению накладных расходов. Теоретически алгоритм SRT обеспечивает минимальные времена ожидания. Однако из­-за издержек на переключения может оказаться так, что в определенных ситуациях в действительности лучшие показатели будет иметь SPN.

1. Стратегия «следующий с наибольшим относительным временем ответа» - HRRN (Highest Response Ratio Next). Алгоритм компенсирует некоторые из недостатков, присущих дисциплине SPN, в частности чрезмерное задержку в обслуживании против длинных процессов и чрезмерно быстрый отклик на новые короткие задачи.

HRRN - это стратегия диспетчеризации без переключения, согласно которому приоритет каждого процесса является не только функцией времени выполнения этого процесса, но также времени, затраченного процессом на ожидание выполнения. После того как процесс получает в свое распоряжение центральный процессор, он выполняется до завершения. Динамические приоритеты при дисциплине HRRN ­ вычисляются по формуле:

**Приоритет = (время ожидания + время выполнения)/время выполнения.**

Поскольку время выполнения находится в знаменателе, предпочтение будет оказываться более коротким процессам. Однако, поскольку в числителе имеется время ожидания, более длинные процессы, которые уже довольно долго ждут, будут также получать определенное предпочтение.

5. Стратегия «циклический перебор / карусельная» - Round Robin (RR) предоставляет всем процессам по очереди одинаковые кванты времени процессора. Квант времени не должен быть слишком мал, иначе накладные расходы на переключение процессов оказываются сравнимыми с полезным временем процессора. Стратегия RR обеспечивает лучшее время ответа, чем SJF, но худшее время оборота.

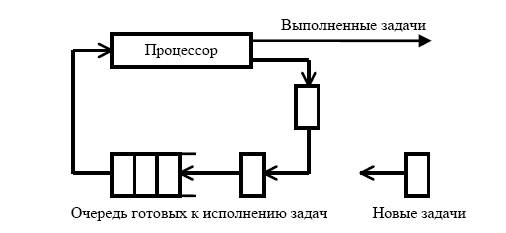


Рисунок 3 - Стратегия диспетчеризации Round Robin

6. Стратегия диспетчеризации по приоритету - предоставляет первым ресурсы процессора более высокоприоритетному процессу. Чтобы избежать ситуации "голодания", ОС постепенно повышает приоритеты процессов, длительное время находящихся в системе.

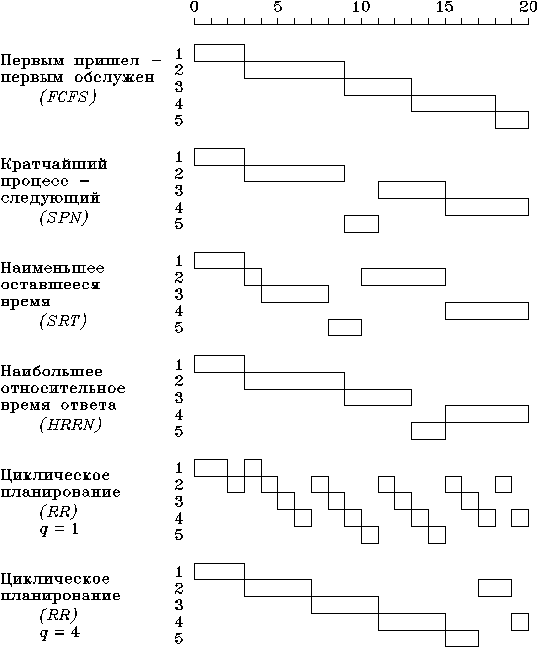


Рисунок 4 - Сравнение алгоритмов диспетчеризации с одной очередью

Задание.

1. Используйте вариант с практической работы № 5 «Процессы» дисциплины «Системное программирование». Реализуйте вариант с помощью примитивных средств диспетчеризации - кооперативного и/или вытесняющего диспетчера. В работе используйте нижеизложенные ограничения.
2. Изучите теоретический материал из лекции «Введение в операционные системы.  
   Кооперативная и вытесняющая многозадачность» и данного методического указания.
3. Реализуйте задачу в среде AVR studio (Microchip Studio).
4. Составьте диаграмму Гранта для Вашего варианта.
5. Данная работа выполняется коллективами **до двух человек.**

Общие ограничения по работе программы.

1. Не используйте динамическую память и функций malloc, realloc, calloc. Память микропроцессора находится на кристалле и подключение внешней памяти не предусмотрено в этом классе микропроцессоров. Процессор считает всю память доступной изначально, тем более это не операционная система.

**Создавайте массивы как в стековой памяти**

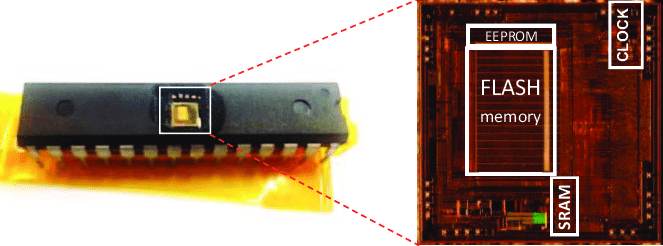


Рисунок 5 – Физическое размещение памяти на однокристальной ЭВМ

1. Работа выполняется в среде AVR studio (Microchip Studio), которая является средой разработки управляющих микропрограмм для микроконтроллеров/микропроцессоров Atmel AVR и Microchip. **Не нужно пытаться открыть/создать/закрыть файл. Эмулируйте файлы посредством массивов во FLASH или RAM памяти.**
2. **Осуществите работу диспетчера в режиме «псевдо многозадачности» вызывая задачи в определённой последовательности и с определенными временными интервалами.** Количество потоков (задач) задайте самостоятельно.
3. Для передачи данных между задачами (потоками) используйте ОЗУ или кольцевой буфер (FIFO)
4. Учитывайте, что simpleDispatcher является **кооперативной системой** и он не предназначен для работы с длительными задачами. Если Вам недостаточно кванта времени диспетчера в 1мС, перенастройте таймер на другой период срабатывания.
   1. displacement Dispatcher является **вытесняющим диспетчером**, что позволяет сохранять в памяти и восстанавливать контекст прерываемой задачи. Данный диспетчер корректно работает с задачами у которых большое время выполнения.

Критерии оценивания задачи.

1. Выполнение основной части данной работы при помощи **только кооперативного диспетчера оценивается максимум в 7 баллов.**
2. Выполнение данной работы при помощи кооперативного диспетчера оценивается до **10 баллов.**
3. Для демонстрации взаимодействия работоспособности программы и взаимодействия микропроцессора с внешним миром при помощи симулятора ISIS Proteus начисляются дополнительные баллы.

Общие рекомендации по организации программы и процесса отладки

**Организация отладки**

Включение отладочного окна

1. Для удобства работы создайте окно для вывода служебных отладочных сообщений. Перейдите во вкладку View и выберите пункт Output.

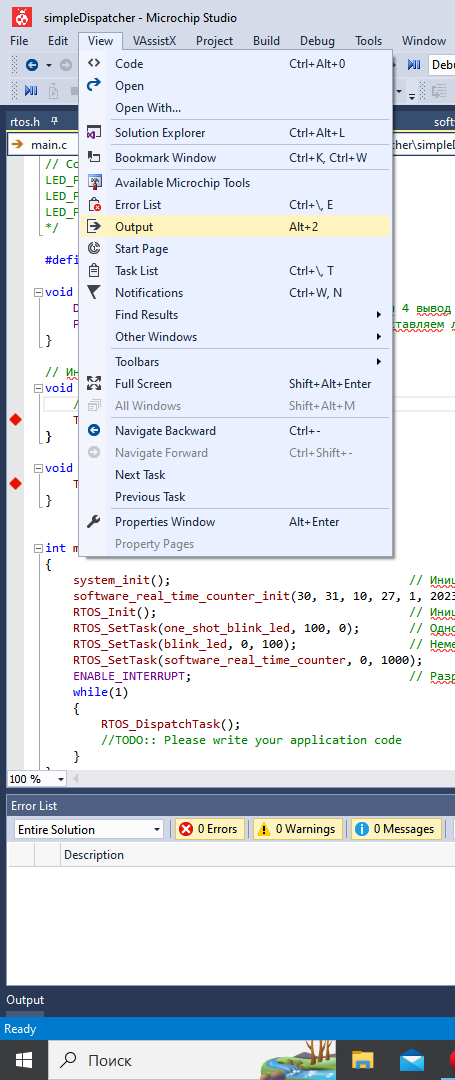
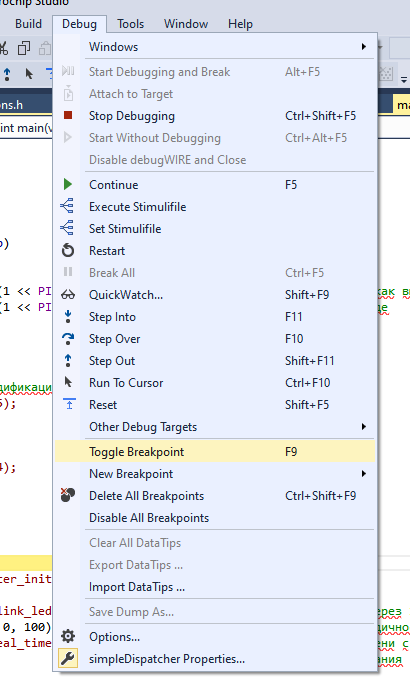
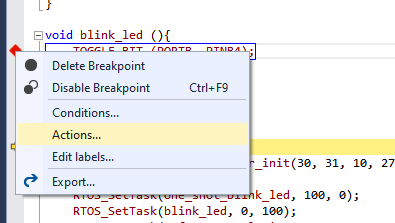


Рисунок 6 – Окно View

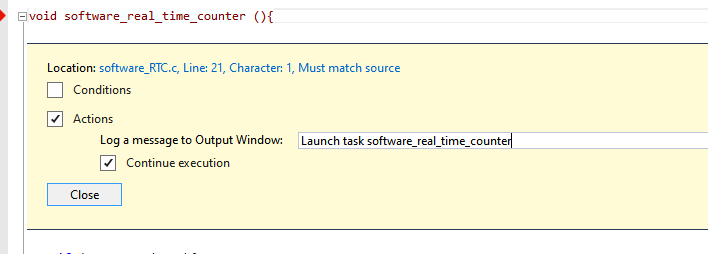
1. Установите точку остановки программы (breakpoint) горячей клавишей «F9» или перейдя во вкладку «Debug» выбрав пункт «Toggle Breakpoint». Установив точку остановки программы, Вы сможете задать отладочное сообщение, которое будет выведено в окно Output.



1. Кликните по интересующей точки останова и выберите пункт Actions



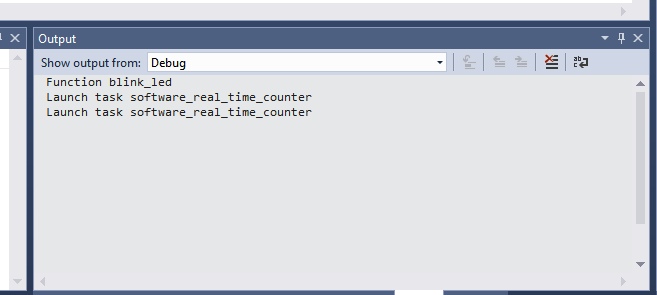
1. Введите интересующею Вас отладочную информацию



Более подробно о работе отладочных инструментов можно ознакомиться на сайте компании Microchip –

https://onlinedocs.microchip.com/pr/GUID-ECD8A826-B1DA-44FC-BE0B-5A53418A47BD-en-US-12/index.html?GUID-DC66A281-31D5-484F-BE7B-AADF8F65D0D9

1. Теперь Вы можете увидеть отладочное сообщение в окне Output



Примерный перечень вопросов

1. Что такое диспетчеризация процессора?
2. В чем основная цель диспетчеризации процессора?
3. Как зависит частота периодов активности процессора от их длительности?
4. Что такое планировщик?
5. Какие разновидности стратегий, с точки зрения прерывания или избежание прерывания процессов, использует планировщик?
6. Что такое стратегия без прерывания процессов?
7. Что такое стратегия с прерыванием процессов?
8. Что такое диспетчер?
9. Что такое латентность диспетчера и каким образом следует оптимизировать данный показатель?
10. Каковы основные критерии диспетчеризации?
11. Что такое диаграмма Ганта?
12. В чем суть стратегии FCFS и каковы ее недостатки?
13. В чем суть стратегии SJF (и SRTF) и оптимальность по какому критерию она обеспечивает?
14. Каким образом и по каким формулам вычисляется предсказание длины следующего периода активности процессора?
15. В чем суть диспетчеризации по приоритетам?
16. Что такое проблема голодания процессов и каково ее решение в ОС?
17. В чем суть стратегии RR, оптимальность по какому критерию она обеспечивает и по какому критерию она хуже, чем SJF?
18. Как зависит число контекстных переключений от величины кванта времени?
19. Как зависит время оборота от величины кванта времени?
20. Что такое многоуровневая аналитическая очередь и процессы каких классов обрабатываются с помощью многоуровневых очередей?
21. Каковы особенности планирования загрузки многопроцессорных систем?
22. Каковы особенности планирования в системах реального времени?