Выводы по лекции 1

– термин «система реального времени» в настоящее время может быть записан так: “Системой реального времени является такая система, корректность функционирования которой определяется не только корректностью выполнения вычислений, но и временем, в которое получен требуемый результат. Если требования по времени не выполняются, то считается, что произошел отказ системы”.

– использование термина «система реального времени» для обозначения интерактивных и высокопроизводительных систем неверно;

– практически все системы промышленной автоматизации являются системами реального времени;

– термин «квазиреальное время» соответствует функционированию в мягком реальном времени;

– принадлежность системы к классу систем реального времени никак не связана с ее быстродействием.

Например, если система предназначена для контроля уровня грунтовых вод, то даже выполняя измерения с периодичностью один раз за полчаса, она будет работать в реальном времени.

Тема 2

Задачи в системах реального времени. Алгоритмы планирования задач в реальном режиме времени

Определение 1. Задачей называется набор операций (машинных инструкций), предназначенный для выполнения логически законченной функции системы.

Задача является одиночным объектом, управление которым осуществляется оболочкой СРВ.

При этом задача конкурирует с другими задачами за получение контроля над ресурсами вычислительной системы.

Задачи классифицируют по двум категориям:

1) Требование по времени функционирования.

2) Вид функционирования.

1) По требованию к времени функционирования выделяют:

- задачи, функционирующие в жестком реальном времени;
- задачи, функционирующие в мягком реальном времени;
- задачи, функционирующие в «нереальном времени».

Задача жесткого РВ – это задача, чье логически правильное или своевременное исполнение считается критическим для действия всей системы.

Предельный срок исполнения называется жестким сроком исполнения. Неспособность удовлетворять этому требованию ведет к отказу всей системы.

Задача мягкого РВ – это задача, в которой исполнение не критично по времени, но ее исполнение желательно для системы (предельный срок исполнения – мягкий крайний срок исполнения задается диапазоном).

Задача «нереального времени» - это задача, для которой нет требований по своевременному выполнению.

2) По виду функционирования выделяют:
- периодические задачи;
- апериодические задачи (асинхронные);
- спорадические задачи;
- фоновые задачи.

Периодические задачи

Определение 2. Периодические задачи – это задачи, которые переходят в состояние выполнения через строго заданный период и выполняется каждый цикл функционирования в системе. Например, обработка и контроль сигнала.

Для систем реального времени требуется четкое и своевременное выполнение каждой периодической задачи.

Периодическая задача выполняется в строго отведенное ей время каждый цикл. Запуск периодической задачи может осуществляться несколько раз за цикл в зависимости от количества меток (сколько меток, столько раз можно запускать цикл). Характеризуется жестким крайним сроком исполнения.

Апериодические задачи

Определение 3. Апериодические задачи – это задачи, имеющие минимальный приоритет в системе и выполняющиеся по событию, характеризуется либо отсутствием крайнего срока исполнения, либо наличием мягкого крайнего срока исполнения.

Функционирование осуществляется только в том случае, если периодические задачи не выполняются.

Функции диагностики и выдача справочной информации, сохранение информации на внешнем носителе.

Спорадические задачи

Определение 4. Спорадические задачи – это апериодические задачи с жестким крайним сроком исполнения.

Приоритет устанавливается на уровне периодических задач, имеют непредсказуемый характер.

Фоновые задачи

Определение 5.Фоновые задачи – это задачи, для которых предельный срок исполнения не задается, либо устанавливается мягкий крайний срок исполнения.

Функционируют в конце каждой метки и только при условии простоя вычислительного узла (при отсутствии других задач).

Может исполняться несколько циклов функционирования системы.

Состояние (статус) задачи.

С точки зрения любой системы, задача может находиться в нескольких состояниях. Число и название этих состояний различаются от одной системы к другой. Практически в любой системе реального времени загруженная на выполнение задача может находиться, по крайней мере, в трех состояниях.

1. Активная задача – это задача, выполняемая системой в текущий момент времени.

2. Готовая задача – это задача, готовая к выполнению и ожидающая у планировщика своей «очереди».

3. Блокированная задача – это задача, выполнение которой приостановлено до наступления определенных событий (освобождение необходимого задаче ресурса, поступление ожидаемого сообщения, завершение интервала ожидания и т. п.)

Планирование задач системах реального времени

Планирование задач – алгоритм построения очереди задач на выполнение.

Алгоритм планирования задач реализуется при помощи планировщика задач.

Существует 2 вида алгоритмов планирования: ***статические*** и ***динамические***.

1. ***Статические алгоритмы*** основаны на применении основных характеристик задач и подразумевают построение примерного плана их исполнения.

Преимущества статических алгоритмов

1. Предсказуемость (если система предсказуема на первом шаге, то она предсказуема на всех остальных);
2. Простота обнаружения ошибок (любая ошибка является следствием неправильно построенной последовательности задач);

Недостатки статического алгоритма:

1. Использование в каждом цикле одной и той же последовательности задачи

2. Не допускается изменение очередности исполнения

3. В результате действия алгоритма суммарная нехватка времени накапливается, так как свободные участки времени не могут использоваться под другие задачи.

2. ***Динамический алгоритм*** предназначен для исполнения последовательности задач во время функционирования системы. Изменение последовательности происходит перед новым тактом и требует от узла дополнительных ресурсов для пересчета последовательности.

Преимущества динамического алгоритма:

1. Оптимальное распределение временных участков для задач;
2. Возможность дополнения списка задач в процессе функционирования системы;

Недостатки динамического алгоритма :

1. Сложность реализации алгоритма;
2. Повышенные требования к вычислительному узлу;
3. Предсказуемость системы зависит от алгоритма на каждом этапе планирования.

Основные алгоритмы планирования периодических задач

Планирование задач связано с разработкой последовательности выполнения задач, выполняемых на одном вычислительном узле.

Существует 2 подхода к планированию периодических задач:

1. ***Фиксированный приоритет*** задачи (приоритет вычисляется один раз до запуска и остается неизменным)

2. ***Динамически назначаемый приоритет*** (может быть установлен во время функционирования задачи, назначение динамического приоритета производится крайним сроком исполнения задач).

В связи с этим были разработаны следующие группы планирования:

1. Алгоритмы планирования задач с фиксированным приоритетом
2. Вытесняющий алгоритм планирования задач (вытеснение одной задачи другой в зависимости от приоритета)

Основные алгоритмы планирования периодических задач:

1) ***RM (Rate Monotonic)*** **–** алгоритм с фиксированным приоритетом. Алгоритм назначается по следующему принципу: чем меньше вызывается периодическая задача, тем больше приоритет. Данный алгоритм всегда формирует оптимальную последовательность задач.

2) ***EDF (Earliest Deadline First)*** – алгоритм с динамическим планированием задачи. Чем меньше срок выполнения, тем выше приоритет. Каждый раз задачи выстраиваются заново в зависимости от критического срока выполнения. Реализованный алгоритм зависит от количества задач в определенный момент времени.

3) ***LSTF (least slack time first)*** – алгоритм планирования. Приоритет назначается по следующему принципу: чем меньше время связывания задачи, тем выше ее приоритет.

Время связывания задачи – разница между крайним критическим сроком и временем исполнения.

Основные алгоритмы планирования апериодических и спорадических задач.

Существует 5 алгоритмов планирования спорадических задач:

1. Планирование спорадической задачи как фоновой задачи

а) Выделяется отдельная фоновая задача, которая отвечает за выполнение всех спорадических запросов, таким образом, все спорадические и апериодические задачи исполняются тогда, когда не исполняются периодические задачи.

б) Планирование спорадической задачи как фоновой без создания дополнительного процесса.

Отличие: в (а) выделяют фиксированное время, а не любое свободное; в (б) задача будет иметь приоритет общий с другими фоновыми задачами и ставится в очередь фоновых задач.

2. «Политика выбора»

Создается периодический процесс, который характеризуется установленным приоритетом. Данный процесс отвечает за выполнение всех апериодических и спорадических задач.

Недостатки: Несовместимость циклического характера алгоритма и случайного характера спорадических задач

Достоинство: Позволяет четко спланировать время исполнения всех задач (самоопределяющий алгоритм).

3. Обмен приоритетом – создание отдельного процесса обслуживания спорадической задачи с динамическим приоритетом.

Таким образом, приоритет позволяет изменить его в процессе функционирования системы.

Для процесса исполняющего спорадические задачи устанавливается самый высокий приоритет.

Таким образом, система обменивает приоритеты между самым высокоприоритетным периодическим процессом и процессом, обслуживающим спорадические задачи.

Обмен производится в начале цикла функционирования системы.

4. Деферабельный сервер

Основан на создании процесса обработки спорадических задач с четко установленным заданным приоритетом. Приоритет устанавливается на самом высоком уровне до запуска системы. Перед повторным запуском данный приоритет может изменить приоритет на самый высокий в данной системе.

В данном алгоритме процесс сохраняет ресурс, выделенный для обслуживания спорадических задач, приостанавливает выполнение спорадических задач, в случае если этот ресурс исчерпан.

В процессе функционирования системы данная задача может прерывать периодические задачи несколько раз.

5. Спорадический сервер

создание периодического процесса исполнения спорадических задач, но приоритет этого процесса устанавливается на уровне приоритета этой задачи. При поступлении спорадической задачи оценивается приоритет текущей задачи: если приоритет спорадической задачи меньше, чем у периодической, то она становится в очередь периодических задач. Сама задача времени исполнения должна иметь наивысший приоритет, данный сервер включает лишь одну задачу – изменение очереди задач.

Каждый алгоритм можно оценить с точки зрения производительности. Для ее оценки используют три параметра:

- нагрузка системы на отказ (BU – BreakDown Utilization);
- нормализованное среднее время ответа (NMRT);
- гарантированная скорость обработки задач (GR).

Параметр нагрузки на отказ BU является степенью использования ресурсов при которой система может гарантировать, что все задачи не будут выполнены в заданные сроки. Чем больше значение BU, тем больше время процессора, которая выполняется задача.

Параметр NMRT – представляет сбой отношение между интервалом времени от готовности задач к выполнению до ее окончания.

$$NMRT = \frac{t_{овз} - t_{нвз}}{t_{обр}}$$

$t_{овз}$ – окончание выполнения задачи;

$t_{нвз}$ – начало выполнения задачи;

$t_{обр}$ - фактическое время процессора

Чем больше значение параметра NMRT, тем больше время простоя задачи.

$$GR = \frac{n_{гар}}{N}$$

GR – оценка производительности системы для задач

$n_{гар}$ – гарантированное количество задач;

N – общее количество задач, ожидающих выполнение.

Если GR>1, то система расписабельная

Если GR<1, то система нерасписабельная

Чем больше GR, тем больше запас времени для выполнения задач.

Планировщик заданий

Планировщик заданий – средство, которое предназначено для использования на вычислительном узле.

Он является средством для обеспечения реализации алгоритмов планирования периодических и спорадических задач.

Выделяют следующие виды планировщиков: ***глобальный*** и ***местный***.

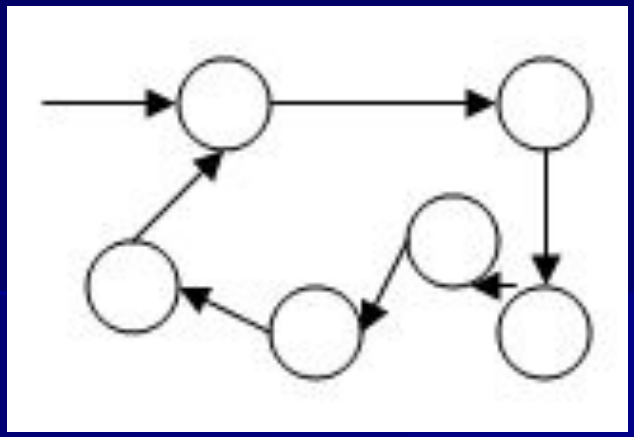
Задачи глобального планировщика:

1. Распределение задач между несколькими вычислительными узлами в распределенных вычислительных системах. Реализация планировщика осуществляется на узле типа сервер, либо на одном из узлов децентрализованного управления на котором нагрузка минимальна.
2. Создание алгоритма формирования образов узла (функций узла с входными и выходными данными). Построение алгоритма для вычисления параметров производительности.

Задачи местного планировщика:

1. Реализация на каждом узле с целью распределения задач на заданный цикл функционирования. Местные планировщики различных узлов являются независимыми программами. Любой местный планировщик является задачей для глобального планировщика и так же подлежит планированию.

Основные функции планировщика: обеспечение последовательности выполнения задач на разных уровнях системы. Каждый цикл функционирования узла планировщик может определять новую последовательность задач, независимую от предыдущей.
Последовательность выполняется внешними условиями и состоянием узла на текущий момент, но при этом должна быть достоверность и последовательность.



Особенностью реализации планировщика является обеспечение максимально наихудших характеристик максимально наихудшее состояние гарантирует функционирование системы с использованием построенного алгоритма.

2. Распределение ресурсов между задачами связано с понятием «гонок». «Гонка» - ситуация по захвату доступа к ресурсу задачей с немаксимальным приоритетом. Понятие «гонка» связано с операционной системой реального времени. В данной ситуации необходимо обеспечить распределение ресурсов или сервер ресурсов.

3. Распределение времени между задачами

Выделение заданного количества тиков для задачи, исполняемой на узле.

Тик – минимальная измеряемая единица времени на узле. Тик зависит от частоты и архитектуры процессора. Для распределения времени может разрабатываться сервер времени – задачу, которого входят выделение тиков.

Алгоритм функционирования планировщика

Планировщик является частью операционной системы.

Из всех задач строится таблица запуска. Определяются списки задач по их виду. В зависимости от типа задачи в списке задач устанавливаются параметры групп запуска.

Для периодических и фоновых задач должны быть установлены следующие параметры:

- стартовая метка запуска данной группы задач;
- период запуска данной группы задач;
- крайний критический срок исполнения.

Имя группы	Стартовая метка	Крайний критический срок исполнения	Период
ptl0	0	5	1
ptl1	5	5	1
ptl2	10	5	1

Для фоновых задач эта таблица расширится относительно стартовой метки, то есть расширится диапазон запуска. Также расширяется крайний критический срок исполнения (min и max).

Имя группы	Старт. метка min	Старт. метка max	Край. крит. срок исп. min	Край. крит. срок исп. max	Период

Для апериодических и спорадических задач также создаются таблицы, но они включают одну строку, если приоритет данного типа задач не учитывается.

Апериодические задачи:

apt <список задач>

метка не описывается

Спорадические задачи:

spt <список задач> запуск задач

Если задачи необходимо делить по приоритетам, то в таблице апериодических и спорадических задач необходимо указывать различные метки для различных приоритетов.

Определение. ***Задачи аппендиксы*** – это задачи, которые исполняются до старта ОС и имеют приоритет выше, чем сама ОС.

Данные задачи связаны с доступом к аппаратуре, например, установка триггеров, регистров и временных меток.

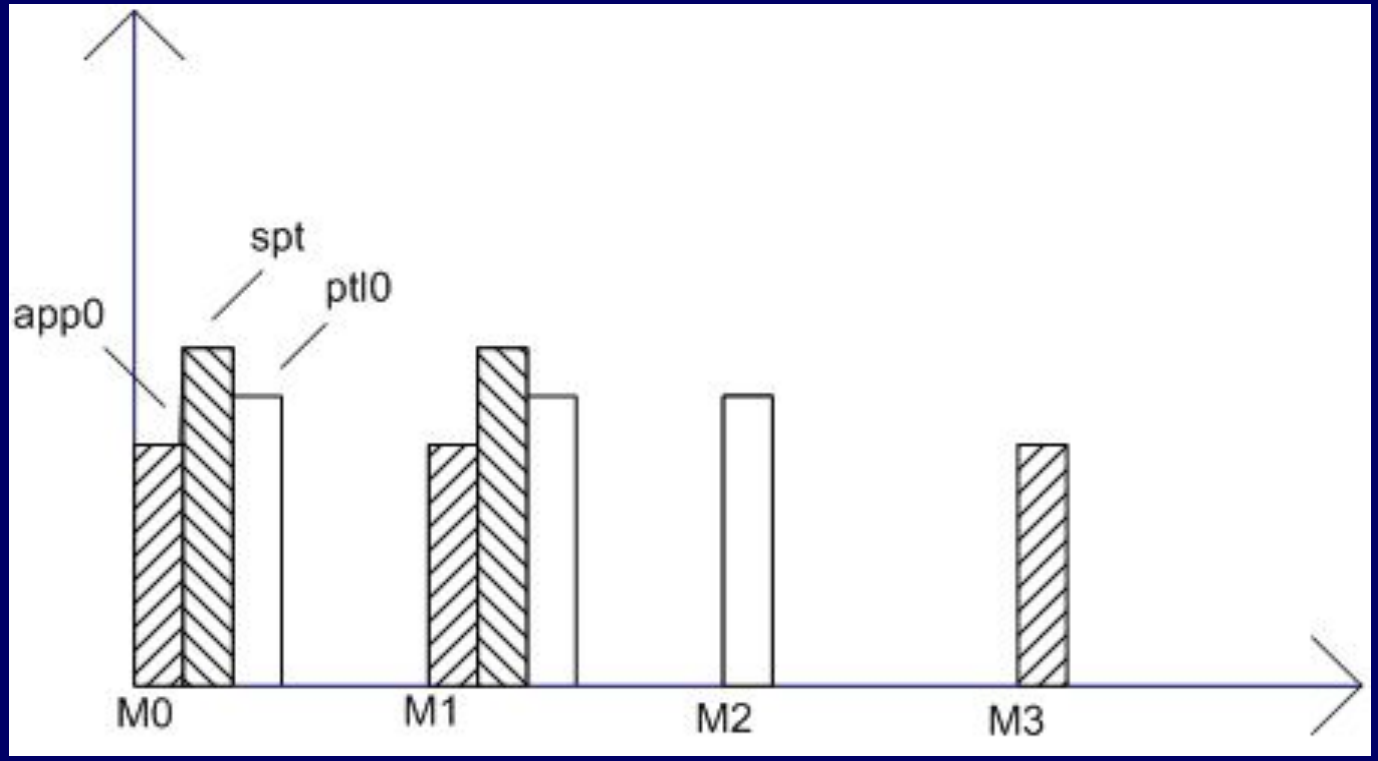
Задачи-аппендиксы описываются в виде таблицы, включающей 1 параметр – метку запуска.

Имя группы	Метка запуска
App0	0
App1	5

app1 <список задач>

Анализ таблиц планировщика

На основании этих таблиц строится список задач на каждом цикле исполнения.



Для аппендиксов, периодических и фоновых задач номер метки должен совпадать с тем, который поставлен в таблице для построения списка.

Список задач может изменяться или не изменяться. Изменение зависит от типового алгоритма планирования.

Исполнение планировщика - завершающая стадия процедуры планирования задач.

Местный планировщик является задачей - аппендиксом. Глобальный планировщик является операционной системой и исполняется по нулевой метке.

1-ый шаг: Для выполнения списка задач создается бесконечный цикл. Для него устанавливается время, определяющее такт функционирования.

2-ой шаг: Производится анализ таблицы запуска. Если планирование статическое, то анализ выполняется один раз. Если планирование динамическое, то анализ выполняется каждый раз при запуске.

3-ий шаг: На основании анализа строится список задач для каждой метки по типам.

4-ый шаг: Вызов функций по заданному списку. Запуск задачи означает вызов функции, либо внешней программы. Последовательность выполнения в рамках списка определяется последовательностью, указанной в таблице. После вызова каждой функции производится проверка временных характеристик и в случае необходимости переход на следующую метку и формирование заключения о невыполнении следующих задач.

Примечание 1: Планировщик во время запуска задач обязан контролировать занятость ресурса текущей задачи, а также должен осуществлять контроль временных характеристик задач, указанных в таблице.

Примечание 2: Если временные характеристики нарушены, должно сформироваться отказное состояние системы.

Спасибо за внимание