Письмо тети

М--

Мы до сих пор не оправились от скоропостижной смерти дяди Рэнди. Пока еще ждем результатов вскрытия и изо всех сил стараемся не упасть духом. Я решила навести порядок в его вещах, главным образом - разобрать компьютеры. Конечно, я заглянула и в гараж, но все, что стоит там, на мой взгляд - просто куча мусора. Пришлю фотографии, как будет время. В посылке находится устройство, стоявшее на его рабочем столе, когда он умер. Возможно, тебе и удастся понять, для чего эта штука и что дядя с ней делал. Он был бы очень рад, если бы его работа не пропала зря.

С любовью,
тетя Дорис.

Вступление

Машина Tessellated Intelligence System (Тесселированная Система Анализа) имеет в своей основе многопроцессорную вычислительную архитектуру, состоящую из неравномерно взаимосвязанных разнородных узлов. **TIS идеально подходит для приложений, требующих потоковой обработки комплексных данных, например, автоматизированной валютной торговли, сбора массивов данных или анализа поведения граждан.** (полужирным здесь и далее обозначено выделение текста хозяином оригинального документа - прим. ред.)

Примечание: Подобные примечания будут встречаться в данном руководстве в случаях, требующих особого внимания, или для ссылки на другие документы, содержащие более подробную информацию по теме.

Архитектура и организация системы

TIS состоит из большого числа независимых локально соединенных узлов (для определения точного числа узлов в конкретной модели устройства обратитесь к его инструкции по эксплуатации). Типы узлов можно условно поделить на обрабатывающие и запоминающие, с несколькими подтипами в рамках каждой категории.

Узлы соединены между собой портами. К каждому узлу может быть подключено до 4 других узлов. Порты обеспечивают обмен информацией между узлами, но их функциональность серьезно ограничена. Сообщение через порты организовано так, что любой узел может объявить о считывании или записи данных в какой-либо порт, что приостановит выполнение программы до тех пор, пока операция не будет одобрена соответствующим узлом.

Примечание: Если два узла будут выполнять одну и ту же команду (запись или чтение) в порту между собой, то они "зависнут" и произойдет аппаратная ошибка. Ознакомьтесь с отдельным руководством "Практическое использование Tessellated Intelligence System и примеры узловой коммуникации", чтобы узнать эффективные и безопасные методы использования портов.

Примечание: Если узел запрашивает команду, которая никогда не сможет быть выполнена другим узлом, то он "зависнет" и произойдет аппаратная ошибка (из этого правила есть исключение; за подробностями обратитесь к документации по конкретным типам узлов). Ознакомьтесь с отдельным руководством "Практическое использование Tessellated Intelligence System и примеры узловой коммуникации", чтобы узнать эффективные и безопасные методы использования портов.

Примечание: Данный документ не указывает время и пропускную способность узловых коммуникационных операций или команд, так как они зависят от модели и модификации оборудования. Обратитесь к руководству конкретной модели за детальным описанием характеристик устройства.

Узел типа T20 - ЗАРЕЗЕРВИРОВАНО

> Примечание: Использование данного типа узлов ограничено конкретными моделями Tessellated Intelligence System, поэтому они не описываются в данном документе. Документация узлов типа T20 распространяется исключительно вместе с системами, содержащими данный тип узлов. **Информация о всех несанкционированных запросах о предоставлении копий документов, описывающих этот узел, поступает в бюро государственной безопасности - в соответствии с законом.** ??? (вопросы проставлены хозяином оригинального руководства - прим. ред.)

Узел типа T21 - Базовый Исполнительный Узел

Архитектура

Базовый Исполнительный Узел отвечает за координацию работы Tessellated Intelligence System. Обработка данных может проходить как внутри Базового Исполнительного Узла, так и перенаправляться на специализированные узлы для обработки или хранения.

Базовый Исполнительный Узел выполняет программу, заданную в наборе команд базового исполнительного узла. Программа Базового Исполнительного Узла состоит из вычислительных и коммуникационных операций. Операции выполняются последовательно, начиная с первой команды в программе. **После выполнения последней команды программы узел автоматически возвращается к первой команде.** Эта способность выполнять записанные команды в непрерывном цикле является основой работы Базовых Исполнительных Узлов.

Помимо коммуникационных портов, общих для всех узлов Tessellated Intelligence System, Базовый Исполнительный Узел содержит ряд регистров, которые могут быть задействованы в ходе выполнения программы. Базовые Исполнительные Узлы не могут использовать дополнительную память. Если программе требуется дополнительное место, узел должен скоординироваться с другим Базовым Исполнительным Узлом или узлом хранения данных.

Узел типа T21. Регистры и порты

Все регистры хранят целочисленные данные от -999 до 999 включительно. Хранение значений в регистрах определяется реализацией, знание принципа хранения не требуется для составления программы Базового Исполнительного Узла.

1. ACC

   Тип: Внутренний

   Описание: ACC - основной регистр хранения Базового Исполнительного Узла. ACC используется как операнд-источник и операнд-получатель по умолчанию во многих операциях, включая арифметические и условные команды.

2. BAK

   Тип: Внутренний (неадресуемый)

   Описание: BAK является временным хранилищем значений из ACC. К нему можно обратиться только с помощью команд SAV и SWP, **и нельзя провести запись или чтение из него напрямую.**

3. NIL

   Тип: Внутренний (особенный)

   Описание: Чтение NIL дает нулевое значение. Запись в NIL не дает никакого результата. NIL вполне может использоваться как операнд-получатель, если возникнет необходимость в его побочном эффекте - сбрасывании результата вычисления.

4. LEFT, RIGHT, UP, DOWN

   Тип: Порт

   Описание: Четыре коммуникационных регистра UP, DOWN, LEFT и RIGHT соответствуют четырем портам, которые каждый Базовый Исполнительный Узел использует для связи с топологически близкими узлами. Некоторые порты отключены у конкретных узлов аппаратно и приведут к "зависанию", если на них подать команды чтения или записи. Обратитесь к схеме соединения узла, чтобы определить, какие порты доступны для использования.

5. ANY

   Тип: Порт (псевдопорт)

   Описание: Когда ANY используется как операнд-источник, алгоритм будет считывать первое значение, пришедшее на любой из портов. Когда ANY выступает как операнд-получатель, результат будет отправлен на узел, который первым выполнит считывание информации с любого порта данного узла.

6. LAST

   Тип: Порт (псевдопорт)

   Описание: LAST обозначает порт, с которым псевдопорт ANY взаимодействовал в последний раз. Во всем остальном его поведение полностью идентично указанию конкретного порта. Считывание или запись в LAST до того, как он был успешно задан функцией чтения или записи псевдопорта ANY, вызовет ненормальное поведение, определяемое реализацией. Обратитесь к отдельному документу "Практическое

использование Tessellated Intelligence System и примеры узловой коммуникации", чтобы посмотреть примеры безопасной работы с псевдопортом LAST.

Узел типа T21. Набор команд

Операнды <SRC> (источник) и <DST> (получатель) могут обозначать порт или внутренний регистр. Любое использование порта блокируется до тех пор, пока соответствующий узел, связанный с этим портом, не закончит считывать или записывать значение. Кроме того, операнд <SRC> может быть целым числом от -999 до 999 включительно. BAK не может использоваться ни как <SRC>, ни как <DST>. Доступ к BAK можно получить только с помощью команд SAV и SWP. <МЕТКИ> - это произвольные текстовые обозначения, использующиеся для указания цели команд перехода.

1. Комментарии

   Синтаксис: # <ТЕКСТ КОММЕНТАРИЯ>

   Описание: Весь текст, идущий после символа комментария (#), игнорируется.

   ```
   Примечание: Текст, идущий после двух символов комментария (##), будет использоваться в качестве названия программы, в которой он находится. Он будет отображаться в отладчике, чтобы упростить работу с несколькими программами.
   ```

2. Метки

   Синтаксис: <ИМЯ МЕТКИ>: <содержание>

   Описание: Метки используются для обозначения целей команд перехода. Если задать метку как цель команды перехода, то следующей командой после перехода будет та, что идет сразу после метки.

   Примеры:

   ```
   LOOP:                   Строка не содержит ничего, кроме метки.

   L: MOV 8, ACC           Метка находится на одной строке вместе с командой.
   ```

3. NOP

   Синтаксис: NOP

Эквивалентный синтаксис: ADD NIL

Описание: NOP является псевдокомандой, не влияющей на внутреннее состояние узла и его коммуникационных портов. NOP автоматически преобразуется в команду ADD NIL.

4. MOV

Синтаксис: MOV <SRC>, <DST>

Описание: происходит считывание из <SRC>, а полученное значение записывается в <DST>.

Примеры:

`MOV 8, ACC`	`Буквальное значение 8 записывается в регистр ACC.`
`MOV LEFT, RIGHT`	`Значение считывается из порта LEFT и записывается в порт RIGHT.`
`MOV UP, NIL`	`Значение считывается из порта UP и сбрасывается.`

5. SWP

Синтаксис: SWP

Описание: Эта команда меняет местами значения регистров ACC и BAK.

6. SAV

Синтаксис: SAV

Описание: Значение регистра ACC записывается в BAK.

7. ADD

Синтаксис: ADD <SRC>

Описание: Значение <SRC> добавляется к значению ACC, результат сохраняется в ACC.

Примеры:

`ADD 16`	`Буквальное значение 16 добавляется к текущему значению регистра ACC.`

```
ADD LEFT    Считанное из порта LEFT значение добавляется к текущему значению
            ACC.
```

8. SUB

   Синтаксис: SUB <SRC>

   Описание: Значение <SRC> вычитается из значения ACC, результат сохраняется в ACC.

   Примеры:

```
SUB 16      Буквальное значение 16 вычитается из текущего значения регистра
            ACC.

SUB LEFT    Считанное из порта LEFT значение вычитается из текущего значения
ACC.
```

9. NEG

   Синтаксис: NEG

   Описание: Значение ACC заменяется арифметически противоположным. Если это значение равно нулю, то ничего не происходит.

10. JMP

    Синтаксис: JMP <МЕТКА>

    Описание: Команда безусловного перехода. Следующей будет выполняться команда, идущая за меткой <МЕТКА>.

11. JEZ

    Синтаксис: JEZ <МЕТКА>
    Описание: Команда условного перехода. Если значение ACC равно нулю, следующей будет выполняться команда, идущая за меткой <МЕТКА>.

12. JNZ

    Синтаксис: JNZ <МЕТКА>

Описание: Команда условного перехода. Если значение ACC не равно нулю, следующей будет выполняться команда, идущая за меткой <МЕТКА>.

13. JGZ

Синтаксис: JGZ <МЕТКА>

Описание: Команда условного перехода. Если значение ACC положительно (больше нуля), следующей будет выполняться команда, идущая за меткой <МЕТКА>.

14. JLZ

Синтаксис: JLZ <МЕТКА>

Описание: Команда условного перехода. Если значение ACC отрицательно (меньше нуля), следующей будет выполняться команда, идущая за меткой <МЕТКА>.

15. JRO

Синтаксис: JRO <SRC>

Описание: Команда безусловного перехода. Следующей будет выполняться команда, чей номер строки отличается от текущей на величину <SRC>.

Примеры:

`JRO 0`	`Следующей будет выполняться эта же самая команда, узел войдет в бесконечный цикл.`
`JRO -1`	`Следующей будет выполняться предыдущая команда.`
`JRO 2`	`Следующая строка будет пропущена, а работа продолжится с команды за ней.`
`JRO ACC`	**`Следующая выполняемая команда алгоритма будет определяться значением ACC.`**

Узел типа T21. Примеры

Нижеследующая программа считывает последовательность значений из порта LEFT, удваивает каждое значение при считывании и отправляет результат в порт RIGHT. Базовые Исполнительные Узлы по умолчанию зациклены, и после выполнения последней команды алгоритма они перейдут к первой

```
MOV LEFT, ACC          Считывание значения из порта LEFT и его запись в регистр
                       ACC.
ADD ACC                Добавление значения ACC к значению ACC, иными словами -
                       удвоение.
MOV ACC, RIGHT         Запись значения регистра ACC в порт RIGHT.
```

Нижеследующая программа показывает алгоритм считывания значений из порта UP с последующей записью положительных значений в порт RIGHT, а отрицательных - в порт LEFT. Нулевые значения отбрасываются.

```
START:
      MOV UP, ACC      Считывание значения из порта UP и его запись в регистр ACC.
      JGZ POSITIVE     Если значение ACC больше нуля, выполняется переход к метке
                       "POSITIVE" JLZ NEGATIVE        Если значение ACC меньше
                       нуля, выполняется переход к метке "NEGATIVE"
      JMP START        Если значение не было больше или меньше нуля, переход к
                       метке "START"
POSITIVE:
      MOV ACC, RIGHT   Запись значения регистра ACC в порт RIGHT.
      JMP START        Возврат к метке "START"
NEGATIVE:
      MOV ACC, LEFT    Запись значения регистра ACC в порт LEFT.
      JMP START        Возврат к метке "START"
```

Узел типа T30 - Узел с Аппаратно-Реализованным Стеком

Архитектура

Узел с Аппаратно-Реализованным Стеком позволяет производить чтение и запись целого ряда значений при помощи простого коммуникационного протокола. (Для уточнения вместимости Узлов с Аппаратно-Реализованным Стеком в конкретной модели устройства обратитесь к его инструкции по эксплуатации)

Коммуникационный протокол

Все взаимодействия с Узлом с Аппаратно-Реализованным Стеком осуществляются через порты. Запись в Узел с Аппаратно-Реализованным Стеком переносит значение в верхушку стека. Если стек полон, команда записи заблокирует узел до появления свободного места. Считывание из Узла с Аппаратно-Реализованным Стеком дает значение с верхушки стека, а потом убирает это значение из самого стека. Если стек пуст, команда чтения заблокирует узел до появления значения.

Узлы с Аппаратно-Реализованным Стеком обычно соединены с несколькими другими узлами. Все подсоединенные узлы могут использовать Узел с Аппаратно-Реализованным Стеком. Порядок действий при одновременном исполнении команд чтения и записи в Узлах с Аппаратно-Реализованным Стеком формально не задан, однако каждая отдельная команда будет выполнена в соответствии с указанным коммуникационным протоколом. Ознакомьтесь с отдельным руководством "Практическое использование Tessellated Intelligence System и примеры узловой коммуникации", чтобы узнать эффективные и безопасные методы использования стековых узлов в системах с несколькими узлами.

Узел типа T31 - Узел с Произвольной Выборкой

```
Примечание: Узлы с Произвольной Выборкой на данный момент не поддерживаются стандартными устройствами Tessellated Intelligence System. Заинтересованным пользователям мы можем предложить программные эмуляторы и экспериментальные образцы устройств с такими узлами. На данный момент конечные данные о характеристиках и поведении данного типа узлов не утверждены, и поэтому в этом документе они не описаны.
```

Список дел

- Понять, кто продал TIS-100 продавцу на блошином рынке
- ~~Восстановить усилитель сигнала~~
- Найти книгу по микрооптимизации
- Получить новые номера

Встроенный интерактивный отладчик

Горячие клавиши

Интерактивный отладчик использует данные сочетания клавиш:
Ctrl+Z: Отменить последнее изменение
Ctrl+Y: Восстановить последнее изменение
Ctrl+X: Вырезать выделенный текст и вставить его в буфер обмена
Ctrl+C: Скопировать выделенный текст в буфер обмена
Ctrl+V: Вставить текст из буфера обмена
Ctrl+стрелка: Переход к соседнему исполнительному узлу
F1: Вызов краткой справки по инструкциям
F2: Статус защитного сертификата ?? (так в оригинале - прим. ред.)
F5: Запустить текущую программу
F6: Приостановка текущей программы или переход на следующий шаг

Точки останова (breakpoints)

Чтобы задать точку останова, поставьте восклицательный знак (!) в начале требуемой строки. После задания точки приостановки программа сама приостановится перед выполнением команды в выбранной строке, позволяя вам с легкостью перейти к отладке программы в таких точках, до которых было бы слишком утомительно идти по одному шагу.

```
MOV LEFT, ACC
!ADD ACC            Программа остановится перед выполнением этой команды
MOV ACC , DOWN
```

Модуль Визуализации

Использование Модуля Визуализации

Модуль Визуализации TIS-100 позволяет программно создавать и выводить изображения на дисплей. Содержимое модуля можно изменять с помощью последовательностей команд, включающих в себя начальную координату X, начальную координату Y, одно или несколько цветовых значений и отрицательную величину, обозначающую конец данной последовательности (обычно -1). **Начало системы координат (0, 0) находится в левом верхнем углу дисплея.**

Графический модуль поддерживает следующие цвета:

0: Черный
1: Темно-серый
2: Светло-серый
3: Белый
4: Красный

Разрешение Модуля Визуализации

Стандартный модуль визуализации TIS-100 имеет **ширину в 30 символов и высоту в 18 символов.**

"Графическая песочница" содержит увеличенный модуль визуализации: 36 символов в ширину и 22 в высоту.

Примеры алгоритмов:

```
0,0,3,-1 Вывод одного белого пикселя в левом верхнем углу дисплея модуля.
0,0,4,4,4,4,4,-1 Вывод горизонтальной красной линии в левом верхнем углу дисплея.
```