Задача 2. Упростить выражение с каналами.

Параллельные процессы это.
? – ввод из канала.
id? - Вычисление выражения приостанавливается, до вывода значения в канал id другим процессом. После ввода из канала id, принятое значение возвращается как результат вычисления выражения ввода. Тип выражения ввода совпадает с типом значений канала.

В выражении можно специфицировать вывод значения в канал выражением вывода вида:
id!value_expr
выражение value_expr, вычисляется, и результат выводится в канал id. Вычисление выражения приостанавливается, до ввода значения в канал другим процессом. Тип value_expr должен совпадать с типом канала. Тип всего выражения вывода Unit.

value_expr1 || value_expr2 – два параллельных процесса вычисления.
Комбинатор параллельной композиции || коммутативен и ассоциативен.

X:=c? || c!5
X:=5

Комбинатор внешнего выбора |=| служит для спецификации выбора между различными типами взаимодействий
Синтаксис:
Value_expr1 |=| value_expr2
Оба выражения дожны иметь одинаковый тип, который также является типом всего выражения. Обычно value_expr1 и value_expr2 начинаются с некоторого типа взаимодействия. Только одно из выражений будет вычисленно, в зависимости от того какой тип взаимодействий хочет произвести окружение.
Комбинатор внешнего выбора ассоциативен и коммутативен.

(x:=c? |=| d!5) || c!1
X:=1

В дополнение к комбинатору внешнего выбора RSL предоставляет комбинатор внутреннего выбора |^|, который специфицирует внутренний выбор между выражениями:
Value_expr1 |^| value_expr2
Выбор value_expr1 или value_expr2 зависит от внутреннего выбора, на который не может повлиять окружение. Оба выражения должны иметь одинаковый тип, который является типом всего выражения.
Комбинатор внутреннего выбора ассоциативен и коммутативен. Рассмотрим следующее выражение:
(x;=c? |^| d!5) || c!1

Выражение c!1 влияет на то какое из выражений x:=c? или d!5 выбирается. Если внутренний выбор выпадает на x:=c?, то результат - выражение:
x:= c? || c!1
потенциально приводящее к взаимодействию через канал c. Если выбор падает на d!5, то результат - выражение:
d!5 || c!1
препятствующее внутренним взаимодействиям.

Interlock комбинатор имеет вид:
value_expr1 ++ value_expr2
Выражение вычисляется вычисленем компонент value_expr1, value_expr2: компоненты вычисляются параллельно, пока одно из них не завершится, далее продолжается вычисление другого (как для комбинатора ||). Однако, в течении параллельного вычисления не происходит взаимодействий с окружением (после того как один закончился, другой может взаимодействовать с окружением). В нашем примере внешнеее взаимодействие с add!i1 не происходит.

Примеры применения комбинатора interlock (++):
value e,e1,e2 : T
channel c,c1,c2 : T
variable x : T

x:=c? ++ c!e is x:=e
Т.е. так как выражения x:=c? и c!e могут взаимодействовать, то они будут взаимодействовать.
При использовании параллельного комбинатора взаимодействия более сложные:
x:=c? || c!e is (x:=e) |^| ((x:= c?; c!e)|=|(c!e; x:=c?)|=|(x:=e))
В зависимости от внутреннего выбора:
либо x:=e – если выполняется как interlock,
либо вычисление завит о внешнего выбора.

(x:=c1? |=| c2!e2) ++ c1!e1 is x:=e1
interlock комбинатор выбирает выражение x:=c1?.
При внутреннем выборе может произойти останов:
(x:=c1? |^| c2!e2) ++ c1!e1 is x:=e1 |^| stop

Пример показывающий как невозможность взаимодействий с окружением приводит к останову:
x:=c1? ++ c2!e is stop
в отличии от параллельного комбинатора:
x:=c1? || c2!e is (x:=c1?; c2!e) |=| (c2!e; x:=c1?)

Во-первых тема cmc-online по теме каналов: http://www.cmc-online.ru/forum/study/?subject=view&msg=1742∞
Вот мои идеи по этому поводу:
основные обозначения: <id>? - узнать значение переменной <id>
<id>!<значение> - записать значение <id>
особенности чтения/записи: чтение происходит "паралельно" с записью, нельзя 2 раза записать или прочитать без противоположной операции. Чем то напоминает стек с мах длинной в 1 элемент.

|^|- рандомный выбор, будет выбран только один вариант. То есть отдельно расписываем кажжый случай.
|| - разделитель паралельных действий, выполняются в произвольном порядке
|=| - выбирается один в зависимости от "внешних" условий(кто готов читать или писать)
Вопрос на засыпку:
(x := 3) ++ (b!4 || y := a?) чему равно и почему?
Ответ: (x := 3) || b!4 || (y := a?)

++ - выполняются оба, но не взамодействуют с окружающими частями, до тех пор пока хотя бы одна не выполнится

```
ВОПРОС: Ну и как он (интерлок) раскрывается в общем случае, т.е. если есть

a||(b ++ c), где c – выполнилось, то это будет
a||(c;b) или как? (если b выполнилось, то a||(b;c) ) ?
```

Думаю, что "a || b || c" в любом случае. А по части приведения этой записи к чему-то с ";" хз.

(вариант № 6)
Условие: Максимально упростить выражение с учетом того, что все возможные обмены сообщениями произошли:

```
(
        y := b? + a?
        |=|
        (case a? of
              0, 1 -> x := a? + 1,
              2 -> x := b? |^| a?
        end)
) || a!0 || b!2
```

Решение:
В первой части (y := b? + a?) требуется, чтобы были записаны a и b, во второй ((case a? of 0,

1 -> x := a? + 1,2 -> x := b? |^| a? end)) требуется запись только a. То есть может выполняться любой из блоков. Рассмотрим оба случая:

1. Выполняется первый блок

   ```
   y := b? + a? || a!0 || b!2
   ```

   записали a, b, их же прочитали получили y := 0 + 2, упрощая;) y := 2.

2. Выполняется второй блок

```
3. (case a? of
4. 0, 1 -> x := a? + 1,
5. 2 -> x := b? |^| a?
   end) ||  a!0  || b!2
```

   Записали a, прочитали a:

   ```
   (case 0 of
   0, 1 -> x := a? + 1,
   2 -> x := b? |^| a?
   end) || b!2
   ```

   Уходим по ветке case'a:

   ```
   x := a? + 1 || b!2
   ```

6. Соединяем решение через знак |^| так как оба варианта равноправны и однозначно определить единственный правильный невозможно.

Ответ:

```
y := 2 |^| x := a? + 1 || b!2
```

Альтернативный вариант от PivoRu, пока не форматированный. Причешем как только, так сразу.

Я так изгаляться с оформлением не способен. Не обессутьте, но постараюсь написать понятно.
Это условия. Написанные на непонятном языке.

(y := b? + a?
|=|

(case a? of

0, 1 -> x := a? + 1,

2 -> x := b? |^| a?

end)

) || a!0 || b!2

Собственно говоря || - это параллелизм.
a!0 - записать в канал a ноль.
a? читать из канал a.
|^| аналог лецонного(я на лекциях не был ни разу) П - внутренний выбор, то есть может произойти либо первой либо второе.
|=| внешний выбор.
++ - interlock появляется вместо ||, означает наивысшей приоритет выражения идущего за ним.
пример. e1 || e2 || e3 ++ e4, сначала произойдут всевозможные обмены чтения записи между e3 и e4, а потом интрлок заменится на || и дальше всё по отработанной схеме :)).

Стоит отметить, что запись в канал и чтение происходят единовременно. А именно если в канале появилась 2, например, затем её кто-то считал, то дальше канал пуст, и все остальные читатели ждут.

Теперь по задаче. По шагам.
1 Смотрим что там может произойти, либо в a запишется 0, либо в b запишется 2.
1-1-1 Пусть сначала произойдёт запись в a. [a!0]

y := b? + a?
|=|

(case a? of

0, 1 -> x := a? + 1,

2 -> x := b? |^| a?

end)

тут либо y := b? + a?, но b не прочитано так что нет,
либо (case a? of

0, 1 -> x := a? + 1,

2 -> x := b? |^| a?

end)

у нас было a!0, значит в кейсе выбираем x := a? + 1
и получаем кусочек ответа x:=a? + 1. Внимание!!! именно так. Так как в канале a ничего нет. мы туда записали 0, но он прочитался при входе в кейс и там снова пусто!!! (я сам не сразу понял это)
ну и дописываем то что осталось || b!2. Получается, что кусок (y := b? + a?
|=|

(case a? of

0, 1 -> x := a? + 1,

2 -> x := b? |^| a?

end)мы уже выполнили. По выбору пошли на кейс. значит присваивание y выполняться не будет и так и останется запись в b 2.

1-1-2 Теперь сначала произошла запись в b. [b!2]
тогда совершенно естественно ожидать что y := b? + a? и следовательно это преобразится в y:=2. Аналогично, из большого куска выполнилась первая половинка, кейс не выполняется. тут два чтения и следовательно прошла запись как в b так и в a. вот и параллельной прицепки не будет.

Теперь записываем ответ собираем его из половинок и соединяем их П (либо |^|):
x:=a?+1 || b!2 П y:=2.
чего и требовалось доказать.

Решение задачи с http://www.cmc-online.ru/forum/study/?subject=view&msg=1742∞ как я (SkiFf) ее вижу
Комментарии приветствуются!

Условие: Максимально упростить выражение с учетом того, что все возможные обмены сообщениями произошли:

```
a!(5+b?) ||
(
        ( x :=
          (if true |^| false then x:=b?;1
               else b!3;x:=2;6 end
```

```
                ) + x )
        ++
        (b!4 || y:=b?)
)
```

Первый шаг - раскрываем внутренний выбор true |^| false

1. true |^| false is true

```
2.   a!(5+b?) ||
3.   (
4.           (x:= (x:=b?;1)+x)
5.           ++
6.           (b!4 || y:=b?)
     )
```

Теперь время заняться интерлоком.
Первая скобка хочет читать из b, во второй параллельно чтение и запись. Возможны варианты:
1.1)Вторая скобка успешно разобралась внутри себя, записала и прочитала из b и завершилась, теперь первая может взаимодействовать с окружением:

```
a!(5+b?) ||
(
        y:=4; (x:= (x:=b?;1)+x)
)
```

Дальше упрощать нечего, первая часть ответа.
Них... не так нах!!! не ";" после y:=4, а "||" - AlexHAX

1.2)Вторая скобка записала в b, первая скобка прочитала из b и завершилась.

```
a!(5+b?) ||
(
        (x:= 5; y:=b?)
)
```

Дальше упрощать нечего, вторая часть ответа.
true |^| false is false

```
a!(5+b?) ||
(
        (x:= (b!3;x:=2;6)+x)
        ++
        (b!4 || y:=b?)
)
```

Снова раскрываем интерлок. Опять возникают варианты с раскрытием параллельности:
2.1)Вторая скобка успешно разобралась внутри себя, записала и прочитала из b и завершилась, теперь первая
может взаимодействовать с окружением:

```
a!(5+b?) || (y:=4; (x:= (b!3;x:=2;6)+x) )
```

Упростим, т.к. первая читает из b а вторая в него пишет
(вообще, с || так можно делать если мы предполагаем что у нас нет окружения, а вот так ли это, я не знаю...)

```
a!8 || (y:=4; x:=8 )
```

Третья часть ответа.
2.2)Первая скобка записала в b, вторая скобка прочитала в у 3

```
a!(5+b?) || (x:= 8; y:=3; b!4)
```

Упростим, как в предыдущем пункте:

```
a!9 || (x:= 8; y:=3;)
```

Четвертая часть ответа.
Ответ:

```
     a!(5+b?) || (y:=4; (x:= (x:=b?;1)+x)) |^|
                 a!(5+b?) || ((x:= 5; y:=b?)) |^|
                 a!8 || (y:=4; x:=8 ) |^|
                 a!9 || (x:= 8; y:=3;)
```

Вопрос:

От интерлока после раскрытия должна оставаться параллельность (||), как она превращается в ";"?

Ответ:

К сожалению, я не знаю правила на этот счет. Я руководствовался тем, что выражения после преобразования должны быть эквивалентны, и поэтому ((x:= (x:=b?;1)+x) + + (b!4 || y:=b?)) в случае 1.1
я раскрыл как (y:=4; (x:= (x:=b?;1)+x)), так как здесь вторая скобка целиком выполниться раньше чем первая, а если ставить параллельно, это так не будет. Фсё НАХ!!! интерлок фсегда заменяецца "||"

Знающие люди, проверьте плиз!!!!!!

Comment

Фсё нах!!! Атветы фсе крывые!!! ++ никада не привращаецца в ; - эта фсё нажухлили в атветах нах!!! ++ превратчаеца тока в || - эта сказала Пятренка. И ващще, йа гаварил с Вадимом - эта чел, каторый кансультацию вёлл - он этти задачи ришил и у няго адин атфет

палучился, а патом йаа паказал етти же задачи Пятренке - у няго друггой. Кароче, фсё зависитт от того, кто задачу решает :)
-- AlexHAX (2005-12-18 17:16:25)
Саш, это почти как искусство: каждый понимает по-своему и каждый находит в этом что-то для себя :)
-- AlenA (2005-12-18 20:15:57)
Определение интерлока в студию! И правило, куда он превращается.
Все, что я знаю:
"Выражение вычисляется вычисленем компонент value_expr1, value_expr2: компоненты вычисляются параллельно, пока одно из них не завершится, далее продолжается вычисление другого (как для комбинатора ||). Однако, в течении параллельного вычисления не происходит взаимодействий с окружением (после того как один закончился, другой может взаимодействоваь с окружением)."

Поэтому, когда один стопорится из-за желания взаимодействоваь с окружением, второй должен закончиться палюбому, паетому то я и не вижу ничего неправильного его через ; написать

Хотя если Петренко сказал, то это весомый аргумент.. Забьем на смысл, напишем || и будем счастливы
-- SkiFf (2005-12-18 21:14:42)
Если написать первой часть без каналов, и через запятую вторую часть, которая не завершилась, то она не сможет взаимодействова со средой, а если бы вместо точки с запятой был знак || то могла бы. Это меняет ответ задачи.
Кроме того мало понятно как решать "вопрос на " наверху страницы, если использовать ";", а не ||
-- YaroslavNedumov (2005-12-18 21:34:28)
"Если написать первой часть без каналов, и через запятую вторую часть, которая не завершилась, то она не сможет взаимодействовасо "

А вот это поподробней! То есть если есть
b!3 || (y:=b? ++ x:=4)
и преобразовать в
b!3 || (x:=4; y:=b?)

то y:=b? не сможет взаимодействова со средой? :)

Пример, когда через ; отражает смысл:

b!5 || (x:=b? ++ x:=3)

раскроем через ||:
b!5 || x:=b? || x:=3
ясно, что в зависимости от того, что раньше выполниться,
x:=3 или x:=5
Но по определению интерлока такого произойти не может!!! Так как x:=3 ДОЛЖНО выполниться раньше, потому что только потом x:=b? сможет взаимодействоваь со средой!
Поэтому правильно b!5 || (x:=3; x:=b?)

все это мое имхо, конечно.