Тема 15.

Списки користувача та блоки для їх формування. Групи і сімейства транзактів

15.1. Списки користувача та блоки для їх формування

Під час руху в моделі транзакти можуть бути заблоковані, наприклад, блоками GATE або TEST. Якщо заблоковані транзакти перебувають у списку поточних подій, то у випадку, коли їх багато, інтерпретатор витрачає велику кількість часу на перегляд списку з метою вибору чергового транзакта для просування. Для економії машинного часу заблоковані транзакти доцільно заносити до ***списків користувача*** й залишати їх там до того часу, поки не будуть виконані умови, які дозволять подальший рух цих транзактів. Крім того, розташування транзактів у списках користувача дає змогу організувати різні дисципліни черг, які відрізняються від дисципліни "першим прийшов – першим обслуговується" (FIFO), реалізованої у списку поточних подій.

Список користувача є деяким буфером, до якого можуть тимчасово заноситися транзакти, виведені зі списку поточних подій. На відміну від списків поточних і майбутніх подій транзакти вводяться у список користувача й виводяться з нього не автоматично, а згідно з рішенням користувача відповідно до логіки моделі за допомогою спеціальних блоків.

Для введення транзактів до списку користувача призначений блок LINK (ввести у список), який може бути використаний у двох режимах:

умовному й безумовному. Обмежимося розглядом лише безумовного режиму, в якому блок LINK має такий формат запису:

LINK A,B,C

Операнди:

A – ім’я або номер списку користувача, до якого автоматично заноситься транзакт після входження у блок LINK. Може бути ім'ям, виразом у дужках, СЧА, СЧА* параметр, додатним цілим числом;

B – визначає принцип упорядкованості, тобто визначає, в яке місце списку користувача треба внести активний транзакт. Допустимі значення: FIFO (транзакт передається в кінець списку); LIFO (транзакт передається на початок списку); PR (транзакти впорядковуються за спаданням пріоритету); P (транзакти розташовуються у порядку зростання значення параметра); M1 (транзакти

розташовуються в порядку зростання відносного часу перебування в моделі).

У полі операнда B можна використовувати й інші СЧА, крім зазначених вище СЧА транзактів: арифметичну змінну, функцію, а також вираз у дужках. У цьому випадку здійснюється обчислення вказаного СЧА для активного транзакта й для решти транзактів, які вже перебувають у списку користувача, починаючи з початку черги. Після цього відбувається впорядкування транзактів у списку користувача за спаданням обчисленого значення. Прямої можливості побудувати список за спаданням значень параметра немає, проте опція BACK блока UNLINK при виборі зі списку сповна її замінює. При визначенні порядку включення у список пріоритет поточного транзакта в загальному випадку ігнорується;

C – блок призначення для активного транзакта у випадку, коли він не приєднується до списку користувача (індикатор зв'язку списку користувача встановлений у нуль); індикатор зв’язку вмикається (встановлюється в 1), якщо він був вимкнений.

У найпростішому варіанті (C не використовується) транзакт завжди поміщається у список користувача A і видаляється з усіх інших списків, виключаючи групи транзактів і списки переривань.

Наприклад, блок

```
LINK  5,FIFO
```

заносить транзакти в кінець списку користувача номер 5 у порядку їхнього надходження до блока. Блок

```
LINK  Spisok,P$Znach
```

заносить транзакти у список користувача з іменем Spisok, впорядковуючи їх за зростанням значення параметра з іменем Znach.

Умови, унаслідок виконання яких транзакт заноситься до списку користувача, в безумовному режимі перевіряються засобами, передбаченими розробником моделі. Наприклад, направити транзакт до списку користувача за умови зайнятості пристрою можна так:

```
……….
… GATE      NU
Prilad,Mit SEIZE
           Prilad
Mit  LINK      Spisok,FIFO
……….
```

Якщо пристрій Prilad зайнятий, то блок GATE не допускає транзакта до

блока SEIZE, а направляє його до блока LINK з іменем Mit, і транзакт заноситься у кінець списку користувача з іменем Spisok.

У цьому прикладі припускалося, що список користувача необмежений, тобто до нього можна заносити будь-яку кількість транзактів. У реальних системах список користувача можна використовувати для імітування, наприклад вхідного накопичувача, ємкість якого, зазвичай, обмежена. Це обмеження можна реалізувати так:

```
Emn  EQU      5
     GATE     NU  Prilad,Mit
     SEIZE    Prilad
Mit  TEST     L  CH$Spisok, Emk,Out
     LINK     Spisok,FIFO
Out  TERMINATE
```

Якщо пристрій Prilad зайнятий, то блок GATE не допускає транзакта до блока SEIZE, а направляє його до блока TEST з іменем Mit, який розташований перед блоком LINK. Якщо поточна довжина списку користувача з іменем Spisok менша від заданої (Emk), то транзакт проходить до списку користувача; в іншому випадку він направляється до блока TERMINATE з міткою Out.

Для виведення одного або кількох транзактів зі списку користувача й занесення їх назад до списку поточних подій призначений блок UNLINK (вивести зі списку), який має такий формат:

UNLINK X A,B,C,D,E,F

Блок виводить транзакти зі списку користувача.

Операнди:

X – умовний оператор відношення між значеннями D і E, при якому відбувається видалення;

A – номер або ім'я списку користувача, з якого виводять транзакти;

B – мітка блока, до якого переходять виведені зі списку транзакти;

C – вказує кількість транзактів, які виводяться зі списку користувача, або ключове слово ALL для виведення всіх транзактів, які є у списку (за замовчуванням ALL);

D – порівнюваний атрибут (атрибут транзакта, який підлягає перевірці, булева змінна для перевірки або BACK для виведення членів з кінця списку користувача);

E – довідкове значення, з яким порівнюється операнд D (не потрібний,

якщо D – булева змінна);

F – номер або ім’я блока, куди переходить транзакт, який виходить з блока UNLINK, якщо зі списку користувача не виведено жодного транзакта. Якщо операнд C не використовується, то транзакт, який виходить, переходить до наступного блока незалежно від кількості виведених транзактів.

Операнди D і E разом з умовним оператором X визначають спосіб і умови виведення транзактів зі списку користувача. Значення оператора X такі самі, як і у блоці TEST. У випадку, коли умовний оператор X потрібно використовувати, але він не вказаний, то за замовчуванням він набуває значення E (рівність). Якщо операндів D та E немає, то не використовують й умовний оператор X. У цьому випадку транзакти виводяться з початку списку, а кількість таких транзактів визначається обов’язковим операндом C.

Операнди D, E й умова необов'язкові. Якщо вони відсутні, виводяться всі транзакти від початку списку до його кінця або до досягнення ліміту (операнд C).

Розглянемо три можливі варіанти операнда D.

1. Якщо операнд D є булевою змінною. Тоді операнд E і оператор X не використовуються. Булева змінна обчислюється відносно транзакта зі списку користувача. Якщо результатом є одиниця, тобто умова виведення виконується, то транзакт виводиться. Кількість транзактів, які виводяться, визначає операнд C. Однак виведено може бути і менше, ніж вказано операндом C: за кількістю результатів обчислення мулевої змінної, які не дорівнюють нулю. Крім того, і транзактів у списку користувача може бути менше, ніж зазначено операндом C.

2. Якщо операнд D є ключовим словом BACK, то витягуються транзакти з кінця списку в кількості, визначеній операндом C. У цьому випадку операнд E й умовний оператор X не використовуються.

3. У решті випадків операнд D обчислюється стосовно транзакта зі списку користувача і використовується як номер параметра цього транзакта, значення якого і є кінцевий результат. Цей результат порівнюється з результатом обчислення операнда E.

Якщо операнд D задає параметр, а операнд E не використовується, то значення параметра транзакта зі списку користувача порівнюється зі значенням такого самого параметра транзакта, який ініціює виведення. Якщо вони рівні, то транзакт виводиться зі списку користувача. У цьому випадку кількість транзактів, які виводяться, також визначає операнд C.

Наприклад, блок

UNLINK 5,Mit,1

виводить транзакт зі списку користувача з номером 5 і направляє його до блока з міткою Mit.

Блок

UNLINK Spisok,Sat,1,BACK

виводить зі списку користувача з іменем Spisok один транзакт з кінця списку і направляє його до блока з міткою Sat. Блок

UNLINK E P$Obl,Mit1,ALL,Rist,P$Rist,Mit2

виводить зі списку користувача, номер якого записаний у параметрі Obl транзакта, що ініціює виведення, і направляє до блока з іменем Mit1 всі транзакти, значення параметра Rist яких дорівнює значенню однойменного параметра транзакта, який ініціює виведення. Якщо таких параметрів у списку немає, то транзакт, який ініціює виведення, буде направлений до блока з іменем Mit2, а в іншому випадку – до наступного блока.

Зазначимо такі особливості виконання блока UNLINK. По-перше, якщо операнди D і E містять посилання на СЧА транзактів, то операнд C обчислюється відносно транзактів у списку користувача, а операнд E – відносно активного транзакта. По-друге, після виведення транзактів зі списку планувальник продовжує або починає просування транзакта з найвищим пріоритетом, а якщо вони рівні, то віддає перевагу транзакту - ініціатору виведення.

Приклад, який показує використання списків користувача для організації нестандартних дисциплін обслуговування. В одноканальній СМО з очікуванням потрібно організувати таку дисципліну обслуговування, коли пріоритет віддається замовленням з найменшим часом обслуговування

```
        GENERATE    (Uniform(3,2,10))
        ASSIGN      Tmin,(Uniform(4,1,15))
        GATE  NU    Prilad,Mit
Mit2    SEIZE       Prilad
        ADVANCE     P$Tmin
        RELEASE     Prilad
        UNLINK      Spisok,Mit2,1
        TERMINATE
Mit     LINK        Spisok,P$Tmin
        GENERATE    1000
        TERMINATE   1
```

START 1

До параметра Tmin активних транзактів у блоці ASSIGN записується випадковий час обслуговування, обчислений за допомогою вмонтованого генератора рівномірного розподілу. Якщо пристрій Prilad вільний, то блок GATE пропускає транзакт до блока SEIZE, і пристрій займається на час P$Tmin. Якщо ж у момент надходження транзакта пристрій зайнятий, то блок GATE направляє транзакт до блока LINK з міткою Mit, де транзакт вводиться до списку Spisok із впорядкуванням за зростанням часу обслуговування, записаного в параметрі P$Tmin. Блок UNLINK у момент звільнення пристрою виводить з початку списку транзакт з найменшим часом обслуговування, забезпечуючи тим самим задану дисципліну.

BUFFER

Блок поміщає активний транзакт у список поточних подій услід за транзактами з таким же пріоритетом, і перегляд списку поточних подій починається спочатку. Остання обставина і є основною метою застосування BUFFER.

15.2. Групи і сімейства транзактів

Між транзактами, що перебувають у моделі, може існувати деякий зв'язок.

Наприклад, процеси промислового виробництва будь-якого складного виробу – "залізного" або програмного – зазвичай проходять три стадії: "зверху вниз" при проектуванні, паралельне виготовлення і збирання з комплексною відладкою. Незалежні роботи, які виконуються паралельно, не можуть забезпечуватися одним транзактом. Транзакт, що запускає їх, повинен розщеплюватися з подальшим збиранням компонентів. Очевидною є необхідність розпізнавання приналежності останніх до єдиного агрегату. Операції об'єднання необхідні також для систем з порогом обслуговування (наприклад, зібрати складний виріб можливо за наявності всіх деталей; автобус відправиться, коли буде зайнято в ньому не менше заданої кількості місць, тощо). З іншого боку, бажано мати можливість виконувати деякі групові дії над однорідними транзактами.

Усі транзакти в моделях належать до множин, які називають ***сімействами*** (***сім'ями, ансамблями***). Як правило, в моделі співіснують декілька різних сімейств. Кожен транзакт є членом тільки одного сімейства. Транзакт, введений у модель блоком GENERATE, отримує числову мітку –

номер сімейства і стає першим і поки єдиним членом нового сімейства. Його нащадки (і можливо нащадки цих нащадків), що формуються при входженні у блок SPLIT, включаються до того ж сімейства.

Сімейства корисні, коли процес повинен чекати настання деяких подій. З поняттям сімейства пов'язані ***списки синхронізації*** – у цьому випадку транзакти чекають у них членів свого сімейства.

Управляти рухом транзактів залежно від їх приналежності до сімейств можна за допомогою блоків ASSEMBLE, GATHER, MATCH. За допомогою блока GATE і логічних умов M / NM вхід транзакта в будь-який блок можна поставити в залежність від наявності елементів того ж сімейства в іншому блоці.

Розглянемо блоки, призначені для обробки транзактів, що належать до одного сімейства.

SPLIT A,B,C

Блок SPLIT (розщепити) виконує функцію копіювання транзакта, що входить до нього і називається ***початковим*** або ***породжуючим***. Операнди:

A – задає число створюваних копій. Усі копії формуються в момент входу транзакта до блока SPLIT. Сам транзакт робить спробу перейти до наступного за номером блока;

B – задає номер або ім’я наступного блока, до якого переходять копії транзакта-оригінала, причому значення операнда B обчислюється для кожної копії окремо;

C – задає номер параметра, що використовується для присвоєння копіям послідовних номерів. Якщо операнд C не використовується (за замовчуванням), номерів транзактів, що виходять з блока SPLIT, в їх параметрах немає.

Після проходження блока SPLIT транзакт-оригінал переходить до наступного блока, а всі копії направляються за адресою, вказаною операндом B, або до наступного блока, якщо операнда B немає (за замовчуванням).

Таким чином, якщо операнд A дорівнює k, то з блока вийдуть k+1 транзакт. Надалі породжуючий транзакт і копії є рівноправними і можуть проходити знову через будь-яке число блоків SPLIT. Кожна нова копія, створена блоком SPLIT, стає членом сімейства транзактів, породженого одним вхідним транзактом, згенерованим блоком GENERATE. Транзакти, що належали до одного сімейства, об'єднуються планувальником у список. Якщо блоком SPLIT транзакти не були пронумеровані, то усередині сімейства

транзактів неможливо встановити, який із транзактів сімейства породжуючий. Число транзактів в сімействі може бути довільним. Якщо копія транзакта входить до блока SPLIT, то вторинна копія стає членом того ж сімейства, що й первинна. Крім значень параметрів до кожної копії записуються значення пріоритету та позначка часу початкового транзакта. Копії почергово надходять до списку поточних подій, причому кожна копія розміщується в кінець відповідного пріоритетного класу.

Кожен транзакт, створений блоком GENERATE, є окремим сімейством (номер транзакта дорівнює номеру сімейства, тобто кількість сімейств дорівнює кількості згенерованих у моделі транзактів). Таким чином, кількість сімейств у системі довільна і сімейство існує доти, поки в ньому є хоча б один транзакт.

Приклади:

SPLIT 7

SPLIT 12,Mitka

SPLIT 3,Mitka,7

У першому прикладі з блока SPLIT вийде 1+7 транзактів. Сім копій слідком за породжуючим транзактом будуть направлені до наступного блока, оскільки операнд B не використовується. Номери в параметр породжуючого і копій транзактів не записуються, оскільки операнд C не заданий. Копії мають той же пріоритет, значення параметрів і час входу в модель, що і породжуючий транзакт.

У другому прикладі блок SPLIT згенерує 1+12 транзактів. Породжуючий транзакт перейде до наступного блока, а 12 копій – до блока з міткою Mitka. Номери в параметри транзактів теж не записуються.

У третьому прикладі блок SPLIT генерує три копії, які направляються до блока з міткою Mitka. Кожна копія має той же пріоритет, час входу в модель і значення параметрів, що і породжуючий, за винятком параметра номер 7. У параметр номер 7 кожної копії буде записаний порядковий номер. Якщо параметр номер 7 породжуючого транзакта не визначений заздалегідь, він буде створений, і йому буде привласнене значення 0. Припустимо, що в даному прикладі значення параметра номер 7 породжуючого транзакта дорівнює 0. Після виходу з блока SPLIT у параметр номер 7 породжуючого транзакта буде записана 1, а в параметр номер 7 копій – 2, 3 і 4 відповідно. Якщо ж параметр номер 7 породжуючого транзакта визначений заздалегідь і його значення дорівнює n, тоді в параметрі номер 7 породжуючого транзакта і копій після

виходу з блока SPLIT будуть записані n+1, n+2, n+3 і n+4 відповідно.

SPLIT 3,P$Adresa,Adresa

Цей приклад демонструє можливість направлення копій транзактів до послідовно розташованих у моделі блоків. Якщо параметр з іменем Adresa містить номер блока, припустимо n, то перша копія буде направлена до блока n+2, друга – до блока n+3, третя – до блока n+4. Ці ж номери будуть записані в параметр з ім'ям Adresa. Породжуючий транзакт зі значенням n+1 параметра з ім'ям Adresa перейде до наступного блока.

ASSEMBLE A

Блок ASSEMBLE (об'єднати) об'єднує задане число транзактів, що належать до одного сімейства, в один транзакт (тобто здійснює збирання заданого числа транзактів). Після об'єднання з блока виходить тільки один транзакт, який переходить до наступного за номером блока. При цьому транзакт зберігає всі свої колишні властивості.

Операнд А задає лічильник транзактів, що беруть участь у збиранні. Значення лічильника необов'язково має дорівнювати чисельності сімейства. Операнд А може бути ім'ям, додатнім цілим числом, виразом у дужках, СЧА, СЧА*параметром. Зауважимо, що:

для кожного сімейства в даному блоці ASSEMBLE одночасно може виконуватися тільки одна операція об'єднання;

у даному блоці ASSEMBLE може паралельно виконуватися операція об'єднання двох і більше сімейств;

для кожного сімейства операції об'єднання можуть одночасно виконуватися в кількох блоках ASSEMBLE.

Оскільки збирання продовжується протягом якогось проміжку модельного часу, блок ASSEMBLE має список синхронізації (парності). У ***список синхронізації*** поміщаються перші транзакти кожного сімейства, що увійшли у блок ASSEMBLE. Вони очікують приходу транзактів зі своїх сімейств. При входженні транзакта у блок ASSEMBLE перевіряється, чи є в списку синхронізації транзакт того ж сімейства. Можливі два випадки: транзакта даного сімейства у списку немає або він є. Розглянемо ці випадки.

1. Транзакта такого ж сімейства у списку синхронізації немає, тобто транзакт, що увійшов, буде першим з сімейства, яке збирається. Тоді обчислюється операнд А, округлюється до цілого числа і зменшується на 1 (один транзакт вже прибув). Залежно від отриманого результату S, що

зберігається в комірці (лічильник збирання) першого транзакта збираємого сімейства, виконуються такі дії:

S<0 – відбувається зупинка через помилку "Лічильник збирання не додатний";

S=0 – транзакт зразу робить спробу увійти в наступний блок (тобто необхідно було зібрати тільки один транзакт);

S≥1 – транзакт виключається зі списку поточних подій і поміщається у список синхронізації блока ASSEMBLE.

2. У списку синхронізації вже є транзакт того ж сімейства, що і транзакт, який щойно увійшов у блок ASSEMBLE. Тоді транзакт, який прибув, знищується. Лічильник збирання, що зберігається в комірці транзакта, який перебуває у списку синхронізації, зменшується на 1. Коли цей лічильник збирання стане рівним нулю (тобто у блок ASSEMBLE увійшло задане число транзактів), то транзакт, що очікує, виводиться зі списку синхронізації. Якщо обслуговування цього транзакта не було перерване жодним з пристроїв, він робить спробу перейти до наступного блоку і у випадку успіху повертається у список поточних подій.

Зауваження. Перерваному транзакту забороняється покидати блок ASSEMBLE доти, поки всі переривання не будуть закінчені. Тому будувати модель рекомендується таким чином, щоб транзакти, обслуговування яких було перерване без видалення (звільнення пристрою), не входили у блок ASSEMBLE.

Приклади:

Блок

ASSEMBLE 5

збирає 5 транзактів одного сімейства, 4 знищується, один переходить у наступний блок. Блок

ASSEMBLE *4

збирає число транзактів, що дорівнює значенню параметра номер 4 транзакта сімейства, який увійшов у блок ASSEMBLE першим.

Приклад використання блоків SPLIT і ASSEMBLE [17].

Розглянемо фрагмент програми моделювання паралельних операцій введення-виведення:

```
SEIZE     CPU
SPLIT     1,PCA2
ADVANCE   FN$TIM1
RELEASE   CPU
```

```
PAE1  ASSEMBLE 2
PCA2 SEIZE        ARM
     ADVANCE   FN$TIM2
     SEIZE        CHAN
     ADVANCE   FN$TIM3
     RELEASE    CHAN
     RELEASE    ARM
     TRANSFER ,PAE1
```

Транзакт займає пристрій (центральний процесор – CPU), і породжує одну копію, за допомогою якої моделюється операція введення-виведення. Потім вихідний транзакт моделює виконання програми процесором, затримуючись на відповідний час у блоці ADVANCE, після чого транзакт вивільнює процесор і входить до блока ASSEMBLE, в якому очікує закінчення операції введення-виведення. Тим часом копія моделює виконання операції введення-виведення і займає пристрій доступу (пристрій ARM), канал, затримується на час моделювання операцій зчитування або запису у блоці ADVANCE, потім вивільнює канал і пристрій доступу та надходить до блока ASSEMBLE. Після цього вихідний транзакт може продовжувати рух.

Блок GATHER (зібрати) діє аналогічно ASSEMBLE і має ті ж властивості. На відміну від ASSEMBLE в ньому при накопиченні заданого в першому операнді числа транзактів рух продовжують ***всі*** вони в порядку FIFO. Формат блока:

GATHER A

Операнд А задає число належних одному сімейству транзактів, яке необхідно зібрати при їх русі одним шляхом (початкове число лічильника збирання). Операнд А може бути ім'ям, додатним цілим числом, виразом у дужках, СЧА, СЧА*параметром.

Блок GATHER може одночасно виконувати збирання транзактів кількох сімейств і також має список синхронізації. При входженні транзакта у блок GATHER перевіряється, чи є у списку синхронізації транзакт того ж сімейства. Можливі два випадки: транзакта даного сімейства у списку немає або він є. Розглянемо ці випадки.

1. Транзакта того ж сімейства у списку синхронізації немає, тобто транзакт, що увійшов, буде першим з сімейства, яке збирається. Тоді обчислюється операнд А, округлюється до цілого числа і зменшується на 1 та запам’ятовується отриманий результат S у комірці транзакта –

лічильнику збирання. Залежно від отриманого результату S можливі такі дії:

S<0 – відбувається зупинка через помилку "Лічильник збирання не додатний";

S=0 – транзакт зразу робить спробу увійти в наступний блок (тобто необхідно було зібрати тільки один транзакт);

S≥1 – транзакт поміщається у список синхронізації блока для очікування прибуття інших транзактів свого сімейства.

2. У списку синхронізації вже є транзакти того ж сімейства, до якого належить транзакт, який щойно увійшов у блок GATHER. Він також поміщається у список синхронізації, а лічильник збирання зменшується на 1. Якщо його значення стане рівним нулю (тобто у блок GATHER увійшло задане число транзактів даного сімейства), всі зібрані транзакти виключаються зі списку синхронізації і поміщаються у список поточних подій.

Зауваження. Якщо серед зібраних транзактів одного сімейства виявляться перервані транзакти, то після закінчення збирання вони не переводяться у список поточних подій. Перервані транзакти перебуватимуть у блоці GATHER, але не у списку синхронізації, доти, поки всі перервані обслуговування не закінчаться. Тому в моделі необхідно передбачити, щоб перервані без видалення (звільнення пристрою) транзакти не входили без необхідності у блок GATHER.

Блок MATCH (синхронізувати) призначений для синхронізації руху транзактів одного сімейства, які рухаються різними шляхами. Для синхронізації необхідні два блока MATCH, які знаходяться у відповідних місцях моделі і називаються спряженими. Формат запису:

MATCH A

Операнд A кожного блока MATCH вказує мітку або номер спряженого йому блока. Операнд може бути іменем, додатним цілим числом, виразом у дужках, СЧА, СЧА*параметром.

Наприклад:

Mit1	MATCH	Mit2
Mit2	MATCH	Mit1

У моделі ці блоки будуть поміщені окремо в паралельних шляхах руху транзактів.

При вході транзакта у блок MATCH операнд A обчислюється і округляється до цілого числа. За набутим значенням визначається спряжений блок MATCH. При його відсутності відбувається зупинка за помилкою.

Якщо спряжений блок є, перевіряється наявність в ньому транзакта з того ж сімейства. Якщо у блоці немає жодного транзакта даного сімейства, транзакт, що надійшов, поміщається у список синхронізації і чекатиме в ньому входу транзакта свого сімейства у спряжений блок MATCH.

Під час надходження такого транзакта у спряжений блок MATCH обидва транзакта одного сімейства будуть виключені зі списку синхронізації й одночасно будуть пропущені у наступні за блоком MATCH блоки.

Одна і та ж пара блоків MATCH може одночасно синхронізувати будь-яке число пар транзактів з різних сімейств. Транзакти одного сімейства також можна синхронізувати в будь-якому числі пар блоків MATCH.

Зауваження 1. Якщо один з транзактів, що синхронізуються, будучи перерваним увійшов у блок MATCH, йому не дозволяється вийти з нього до тих пір, поки всі його перервані обслуговування не будуть закінчені. Тому будувати модель потрібно так, щоб транзакти, обслуговування яких перерване без видалення (без звільнення пристроїв), не входили в блоки MATCH.

Зауваження 2. Блок MATCH може бути спряженим сам собі. При цьому його дія буде аналогічною дії блока GATHER з операндом A, що дорівнює 2. Приклад запису в моделі:

Mit1 MATCH Mit1

У сімейств немає імен, на які можна було б посилатися при описі логіки моделі. За допомогою блока ADOPT можна змінити приналежність активного транзакта до сімейства. Формат блока:

ADOPT A

Операнд A задає ім'я або номер сімейства, до якого буде належати активний транзакт.

Приналежність транзактів до сімейств визначається тільки *генетично*. З іншого боку, є можливість їх *довільного групування*. Кожен транзакт може одночасно належати будь-якому числу різних груп. Транзакт стає членом групи, пройшовши через блок JOIN (приєднати). Формат блока:

JOIN A,B

Блок додає активний транзакт до групи транзактів або число – до числової групи. Операнди:

A – номер групи, в яку буде доданий новий член; B –

значення, яке включається в числову групу.

Якщо B специфікує число, блок JOIN оперує з числами, інакше – з транзактами. Працюючи з числами, JOIN обчислює операнди A і B. Потім

число, задане в операнді B, включається в числову групу, вказану A. Оперуючи з транзактами, блок JOIN включає вхідний транзакт у групу, специфіковану операндом A.

Нові члени поміщаються у список групи в порядку їх надходження. Групи чисел відрізняються від груп транзактів, навіть якщо мають однакові номери. Після блока JOIN активний транзакт переходить у наступний блок.

На відміну від транзактів, що входять у списки, включений в групу транзакт не стає пасивним і продовжує рухатися, залишаючись в групі до завершення своєї траєкторії. Видалення транзакта з групи можливе блоком TERMINATE, об'єднанням його з іншими транзактами в блоці ASSEMBLE або видаленням з групи (блок REMOVE).

REMOVE X A,B,C,D,E,F

Блок видаляє члени з груп. Його операнди:

X – відношення між D і E, при якому виконується видалення; A – номер групи, з якої проводиться видалення;

B – максимальне число транзактів (може бути ALL, ім'я, константа, СЧА), що видаляються;

C – значення, яке має бути видалене з числової групи, в режимі транзактів поле C не використовується;

D – номер параметра транзакта, який визначає члени групи, що видаляються, або PR, якщо ознакою для видалення є пріоритет;

E – довідкове значення, з яким порівнюється операнд D активного транзакта;

F – альтернативний блок для активного транзакта, коли F не заданий, а також при результативному видаленні, тразакт продовжує рух по моделі.

Блок REMOVE діє в числовому режимі, якщо використовується операнд C. Інакше він працює з групою транзактів.

Найбільш використовувані комплексні варіанти:

– самовидалення - якщо B, D, E не задані, транзакт видаляється;

– лічильник – D, E не задані, кількість транзактів вказана операндом B, видаляється з групи (якщо не вистачає – всі). ALL в цьому операнді видаляє всю групу;

– управління за параметром або пріоритетом - використовуються операнди B, D, E. Операнд B указує кількість (можливо, ALL), D – параметр, що вивчається, або PR, E – порівнюваний СЧА. Умова видалення записується після REMOVE, за замовчуванням мається на увазі рівність;

– екстремаль – умова має вигляд MAX або MIN. Аналогічно попередньому випадку B, C, E не застосовуються;

– у чисельній моді REMOVE використовує операнди A, C і, можливо, F. Операндом C вказується шукане числове значення.

Знайдений елемент покидає групу, за відсутності – активний транзакт переходить за адресою F.

Шлях транзакта по моделі можна зробити залежним від його приналежності до групи, включивши в маршрут блок EXAMINE.

EXAMINE A,B,C

Блок застосовується для перевірки членства в числовій групі або в групі транзактів і визначає рух транзакта, не змінюючи склад групи. Він працює або в "числовому" режимі, або в режимі "транзакта". Якщо використовується операнд B, режим числовий. Операнди:

A – ідентифікатор групи, яка перевіряється;

B – вказує, яке числове значення розшукується;

C – альтернативний блок для активного транзакта, якщо члени групи не знайдені.

Приклад:

EXAMINE Color,P$Col,Necol

Якщо числова група з ім'ям Color не містить значення, записане в параметрі транзакта з ім'ям Col, то активний транзакт перейде до блока з міткою Necol. Якщо значення виявиться членом групи, активний транзакт переходить до наступного по порядку блока.

У режимі транзактів перевіряється приналежність активного транзакта до групи, вказаної операндом A.

SCAN X A,B,C,D,E,F

Блок працює тільки з групами транзактів. Він дозволяє тому транзакту (не обов'язково членові групи A), що увійшов до блока, вибрати перший елемент групи, параметр або пріоритет (B) якого задовольняють умові. Операнди:

X – відношення між B і C для вибору члена групи; A –

група, що перевіряється;

B – атрибут (PR або параметр), що перевіряється;

C – довідкове значення, з яким порівнюватиметься операнд B;

D – PR або номер параметра, значення якого має бути призначене параметру активного транзакта;

E – номер параметра, що набуває значення;

F – нове призначення для того транзакта, що увійшов до блока SCAN, при неуспіху.

Операнд B вказує, який атрибут транзакта перевірятиметься. Блок SCAN вибирає з групи перший транзакт, який задовольняє умові, і запам'ятовує заданий атрибут у параметрі транзакта, що увійшов у блок SCAN. Якщо активний транзакт не має такого параметра, то він створюється.

Якщо використовується умова І/АБО для операндів B, C, то перевіряється перший член групи. Якщо умовою є MIN або MAX відшукується транзакт з екстремальним значенням вказаного атрибута. Якщо таких транзактів декілька, то береться перший з них. Якщо умовою є MIN або MAX, то операнд C не використовується. За умовчанням операнд C дорівнює нулю. Якщо умови перевірки не задані, береться перший транзакт групи. Після вибору транзакта, його атрибут, специфікований операндом D, записується в параметр транзакта, заданий операндом E. Далі транзакт, що увійшов у блок SCAN, переходить до наступного блока.

Операнди B і D завжди відносяться до члена групи, який перевіряється. Тут може використовуватися будь-який СЧА. Якщо для обчислення СЧА потребується транзакт, то використовується той, що перевіряється. Результатом є номер параметра, значення якого і є кінцевим результатом. Якщо умовний операнд X не заданий (за замовчуванням мається на увазі рівність), але задані B і C, то порівнювані значення мають бути рівними.

Якщо у групі не знайдеться жодного транзакта, який задовольняє умові, то транзакт, що увійшов у блок SCAN, переходить відповідно до операнда F, або до наступного блока.

Приклади:

```
SCAN    MIN  Grup,Rozmir,,Nomer,50
```

У цьому прикладі переглядаються всі транзакти з групи Grup і вибирається той, який має найменше значення параметра Rozmir. Якщо таких транзактів декілька, то береться перший з них. Для нього обчислюється операнд D. Потім параметру з номером 50 активного транзакта привласнюється значення параметра Nomer. Якщо група виявиться пустою, то ніяких дій не виконується.

```
SCAN    GE   Part,P$Nabir,7,P$Nomer,Mit
```

У цьому прикладі при вході транзакта у блок SCAN група транзактів з іменем Part переглядатиметься виявлення першого транзакта з параметром Nabir, який перевищить або дорівнюватиме 7. Коли транзакт буде знайдений,

значення його параметра Nomer буде передане відповідному параметру транзакта, що увійшов. Потім активний транзакт переходить до наступного блока. Якщо у групі не виявиться жодного такого члена, то наступним буде блок з міткою Mit.

Модифікувати параметр або пріоритет групи в цілому або її окремих членів можна за допомогою блока ALTER. При цьому місцеперебування членів групи є несуттєвим.

ALTER X A,B,C,D,E,F,G

Блок змінює пріоритет або параметр вибраних членів групи транзактів. Операнди:

X – покажчик відношення (буквенний!);

A – група транзактів, члени якої перевіряються на необхідність зміни;

B – максимальне число змін транзактів (за замовчуванням ALL);

C – змінюваний атрибут (PR або номер, або ім'я параметра тран закта – члена групи);

D – замінююче значення;

E – PR або параметр, який використовується для перевірки відношення;

F – базове значення, з яким порівнюється операнд E;

G – альтернативний вихід для активного транзакта за відсутності змін.

Блок ALTER вибирає транзакти з групи і змінює один з атрибутів цих транзактів. Вказаному в C атрибуту відібраних членів групи привласнюється значення D. Змінені транзакти не змінюють свою траєкторію руху, але можуть бути перенаправлені згідно з операндом G.

При відсутності покажчика відношення й операндів E або F (без перевірок) змінюються всі транзакти – до верхньої межі (операнд B). Операнд E визначає, який атрибут перевірятиметься. Він може порівню- ватися з мінімумом або максимумом атрибутів всіх членів групи при MIN або MAX як вказівника відношення. Тоді будуть змінюватися всі перевірені транзакти, які мають мінімальний або максимальний атрибут. У цьому випадку операнд F не використовується.

Атрибут члена групи можна порівняти з операндом F за вказаним відношенням. В операнді E мається на увазі параметр транзакта, який перевіряється. У випадку, якщо будь-який інший операнд є СЧА, що відноситься до транзакта, то він обчислюється для транзакта, який увійшов у

блок.

Відношення між атрибутом транзакта (операнд E) і перевірюваним значенням (операнд F) визначає умови зміни транзакта. За замовчуванням покажчиком відношення вважається рівність.

Операнд G вказує транзакту, який увійшов у блок, альтернативний маршрут коли: 1) жоден транзакт не був змінений; 2) задане операндом B значення лічильника змін не може бути досягнене. Якщо операнд G не використовується, транзакт переходить до наступного по порядку блока.

Приклади:

ALTER Grup,ALL,Znak,123

У цьому прикладі всім транзактам – членам групи Grup, встановлюється параметр з іменем Znak рівним 123.

ALTER NE Grup,5,Znak,45,P$Param,12,MitOut

У групі з іменем Grup відшукуються транзакти, у яких параметр з іменем Param не дорівнює 12. Першим 5 транзактам, які зустрінуться при перевірці, параметр з іменем Znak встановиться в 45. Якщо 5 транзактів не можуть бути знайдені при перевірці, транзакт робить спробу увійти в блок з міткою MitOut, інакше транзакт переходить до наступного блока.

Контроль приналежності транзактів до групи відбувається через списки їх номерів. У блоках JOIN, REMOVE, EXAMINE є можливість працювати в числовій моді – з вільно призначеними числами замість номерів. Розробник моделі, наприклад, затребувати список всіх зайнятих пристроїв. Блоки SCAN і ALTER цей режим не підтримують.

На кожну групу створюється FIFO-список. У режимі транзактів у списку фігурують номери транзактів, у числовому – довільні числа, назначені користувачем. У межах групи числа мають бути різними. Числова мода зазвичай використовується для збереження номерів пристроїв і БКП. Вибір моди визначається першим блоком, що посилається на групу у процесі прогону. Блоки JOIN, REMOVE, EXAMINE працюють з групою в будь-якому режимі, SCAN і ALTER – тільки в режимі транзактів. Крім транзактів, групи можуть складатися і з пристроїв і з БКП (нарізно), що задовольняють умові відбору. Відповідно все сказане вище про транзакти (виключаючи рух) відноситься і до них.

Висновки

1. Списки користувача дозволяють моделювати будь-яку дисципліну

обслуговування крім "першим прийшов – першим обслугований".

2. Розщеплення транзактів з подальшим їх збиранням дозволяє моделювати роботи, що паралельно виконуються над одним транзактом.
3. Операції об'єднання дозволяють моделювати системи з порогом обслуговування, а також дають можливість виконувати групові дії над однорідними транзактами.
4. Приналежність транзактів до сімейств визначається тільки генетично, але розробник моделі має можливість довільно групувати їх. Транзакт може належати одночасно до кількох груп, але тільки до одного сімейства.
5. Транзакти, включені у групи, залишаються активними, тобто продовжують рухатися і залишаються у групі до завершення своєї траєкторії.

Контрольні запитання та завдання

1. Дайте визначення списку користувача. Для чого він призначений?
2. Які блоки призначені для формування списків користувача? Назвіть особливості роботи цих блоків.
3. Дайте визначення групи і сімейства транзактів. Назвіть відмінності між ними. Для чого вони використовуються в GPSS World?
4. Які блоки призначені для роботи з групами та сімействами транзактів? Назвіть особливості роботи цих блоків.