АСИНХРОННИ ЦИКЛИЧЕСКИ МИКРОКОНВЕЙЕРНИ СТРУКТУРИ С МНОГОТАКТОВИ ТЕЛА

Димитър С. Тянев

Резюме: Представен е нов метод за хардуерна реализация на циклически алгоритмични структури с асинхронна микроконвейерна организация. Изследването е върху цикъли със следусловие и многотактови циклически тела с произволна структура. За цикли с отнапред неизвестен брой повторения представената конвейерна организация на циклическото тяло е единствено възможната.. Формулирани са и са решени четири нови за конвейерната организация задачи. Представен е синтезът и принципни логически схеми на необходимите конвейерни автомати.

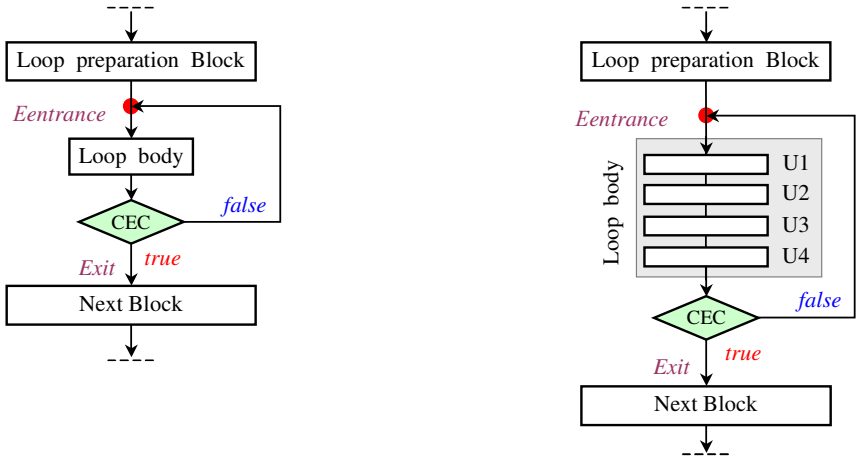
Asynchronous micro-pipeline loop-structures with multi-stages body's

Dimitar S. Tyanev

Abstract: A new method for hardware realization of algorithmic loop-structures with asynchronous micro-pipeline organization is representative. The study was on a loops with post-condition and body's with common structures. For loops with beforehand unknown number of repetitions, presented pipeline organization is only possible. Four new tasks for pipeline organization are formulated and solved. The synthesis and logical circuit of the necessary pipeline controllers are representative.

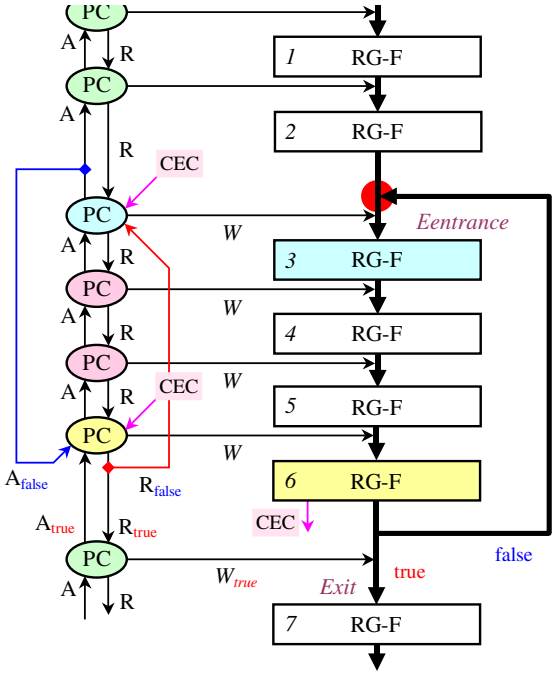
1. Постановка на задачата

Разглежда се микроконвейерна реализация на циклически алгоритмични структури, които притежават линейно многотактово тяло. На фиг.1 е илюстрирано разбирането за тази структура. Цикъл с еднотактово микроконвейерно звено в тялото е илюстриран чрез лявата блок-схема. Многотактовият вариант на тялото на цикъла е показан чрез дясната блок-схема, като 4-степенна линейна последователност от звената *U1*, *U2*, *U3* и *U4*. Принципна разлика между двата варианта няма. Целта на изследването е разработване на метод за управление на такива циклически структури с асинхронна микроконвейерна организация. Първоначалната представа за поставената цел е илюстрирана на фиг.2. Както може да се види, микроконвейерните звена (представени чрез своите регистри фиксатори RG-F) се управляват от конвейерни автомати *PC* (*Pipeline Controller*), които генерират импулсите за запис на данни след успешно "ръкостискане" със своите съседи.



Фиг. 1. *Алгоритмична структура цикъл със следусловие*

Ръкостискането се основава на сигналите заявка *Request* (*R*) и потвърждение *Acknowledgement* (*A*), които автоматите използват в диалога.



Фиг. 2. *Асинхронна конвейерна организация на цикъл*

Изобразените звена са номерирани от 1 до 7. Тези номера ще бъдат използвани и за означаване на конвейерните автомати, на техните сигнали или връзки, но няма да имат нищо общо с поредовия номер на зарежданите в конвейера задачи.

2. Хардуерен аспект на анализа

В циклическата структура от фиг.2 има две съществени точки. Това са входната точка (*Entrance*) и изходната точка (*Exit*). По своята природа, тези две точки са свързани чрез обратната връзка, която съответства на алгоритмичния преход "лъжа" на условието за край на цикъла *CEC* (*Condition of End Cycle*). Управлението на микроконвейерните звена в тези две точки е в зависимост от логическата стойност на условието за край *CEC*. Конвейерните автомати на входното и на изходното звена остават свързани през цялото време на циклическото изчисление. Заявката, която генерира автоматът в изходната точка, трябва да се разклонява, пренасочвайки актуалната си стойност в зависимост от условието *CEC* или към автомата на входното звено (R_{false}), или към автомата на звеното в изхода на цикъла (R_{true}). В същото време този автомат ще трябва да формира своето потвърждение въз основа на две входни за него потвърждения – това, което идва от автомата на входното звено (A_{false}) и това, което идва от автомата на звеното по изхода на цикъла (A_{true}).

Що се отнася до конвейерния автомат, управляващ входното звено на цикъла, той е свързан с две входни заявки – тази, която идва от звеното, предхождащо входната точка (R), и идващата от автомата на изходното звено (R_{false}). В отговор трябва да връща потвърждение или към предходния автомат (A) или към автомата в изходната точка (A_{false}).

Началният анализ на логическата структура от фиг.2 изявява две задачи. Задачите са за синтез на принципните логически схеми на конвейерните автомати в двете характерни точки, тъй като те имат значително по-сложна система за диалог в сравнение с останалите автомати.

3. Организационен аспект на анализа

Формулираните по-горе задачи се отнасят само до чисто хардуерния аспект на цялостния проблем, при това в един начален и неокончателен за него вариант. Разглежданата циклическа структурата е интересна още в организационен аспект. Нейното

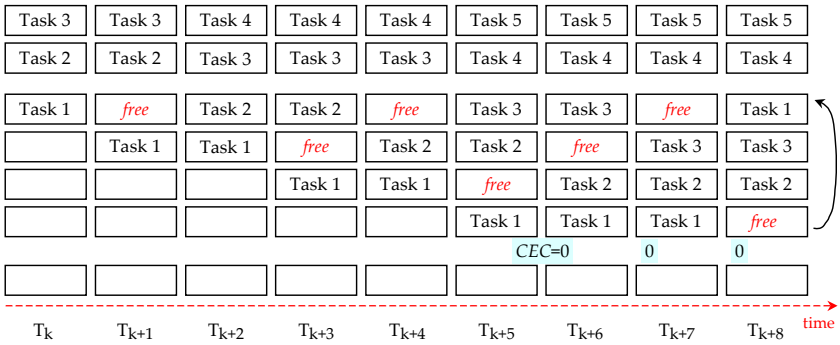
функциониране следва да бъде организирано конвейерно. Разглеждана обобщено като "едрозърнеста", структурата се определя като многотактова, тъй като е обхваната от обратна връзка. В същото време обаче като "дребнозърнеста", нейното циклическо тяло е многотактово, представено в примера чрез линеен конвейерен участък с 4 степени. При тези условия, функционирането на циклическата структура може да се организира по два различни начина.

Първият начин съответства на функционирането на еднотактовата циклическа структура (фиг.1), за която са познати различни варианти [1]. Става дума за функциониране, при което, когато една задача постъпи в структурата, тя не я освобождава, докато изчисленията не завършат. Дори когато циклическото тяло е многотактово, както бе определено, задачата се превърта многократно, като преминава последователно от звено към звено в тялото. Този вид организация се определя като ниско производителна, защото във всеки момент в конвейерното тяло работи само едно звено, а останалите се намират в изходно състояние. Този вариант на организация е практически безинтересен и ще бъде използван единствено като основа за сравнение.

По-високата производителност изисква ангажирането на всички свободни звена в циклическото тяло на структурата. Конвейерната организация позволява в тялото на цикъла да се въртят едновременно повече от една задачи. Така например, след навлизане на задача №1 в тялото на цикъла, при което звеното във входната точка се освобождава, а входната даннова шина все още не е изключена, в цикъла може да постъпи задача №2. Аналогично, когато тя, следвайки задача 1, освободи звеното във входната точка, на нейно място може да постъпи задача 3.

Като има предвид, че в асинхронните конвейери регистрите фиксатори се изграждат от тригери със структура *Latch*, едно циклическо тяло с m на брой степени може да бъде заредено безпроблемно максимум с $(m-1)$ на брой задачи. Това е така, защото m-степенното кръгово изместване на задачите от звено към звено в тялото на цикъла не може да се осъществи без наличие на свободно звено. Обратната връзка на цикъла не би могла да върне във входното звено задача, идваща от изходното звено, ако входното не е свободно.

Процесът на зареждане и превъртане на задачи в примерната структура е илюстриран на фиг.3. Първото връщане на задача 1 по обратната връзка във входното звено е изобразено в такт T_{k+8}. Преходът на задача 1 от звено 6 в звено 3 се дължи на нулевата стойност на условието *CEC*, която звено 6 изчислява за нея. Тази логическа стойност определя данновите връзки в изходната и входната точки, както беше пояснено в началото.



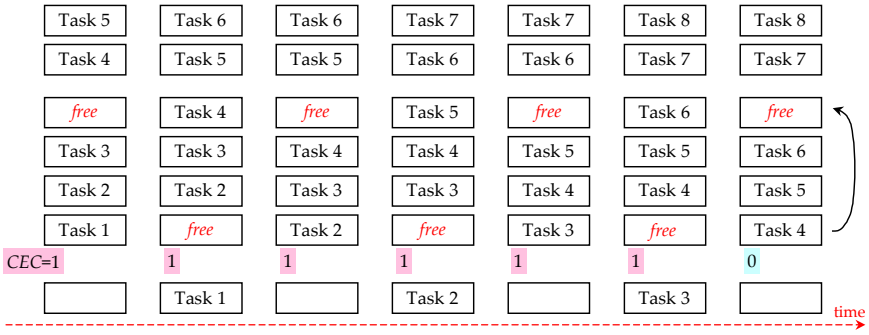
Фиг. 3. *Навлизане на много задачи в цикъла и превъртане*

Казаното до момента все още не изчерпва изясняването на организационния аспект на функционирането на конвейеризираната циклическа структура. За целите на по-нататъшния анализ ще въведем допълнителна конкретност.

4. Цикъл с предварително известен брой повторения

Когато реализираният цикъл е от вида с предварително известен брой повторения, заредените в неговото тяло задачи се превъртват еднакъв брой пъти. От това следва, че

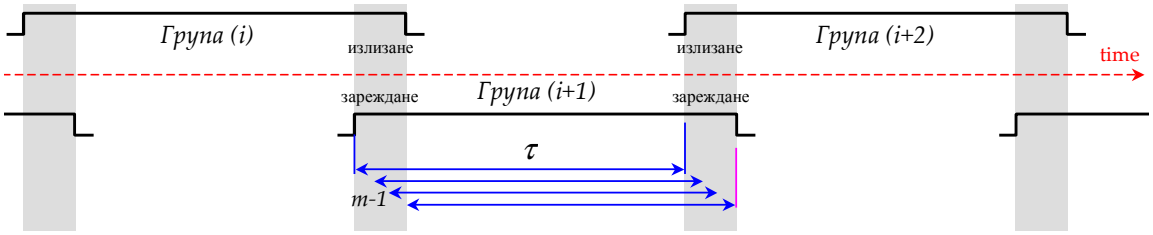
тяхната последователност ще се запази и до момента на последното повторение. Условието за край на цикъла *CEC* ще се изпълни за първи път най-напред за задача 1 и тогава тя трябва да излезе от цикъла. След нея по същата причина последователно ще напуснат цикъла и останалите задачи от групата. След като звеното във входната точка бъде освободено, тъй като обратната връзка по това време ще бъде изключена (*CEC*=1), в него може да постъпи нова задача (втората колонка в рисунката от фиг.4). Така процесът на последователно излизане от цикъла на трите изпълнени задачи №(1,2,3) ще бъде съпроводен с последователно зареждане на 3 нови задачи №(4,5,6). Фигура 4 илюстрира тези процеси.



Фиг. 3.8.4. *Процес на излизане на задачи от цикъла*

Вижда се, че в последния изобразен такт за задача 4 се изчислява условието *CEC*=0. С това започва превъртането в тялото на цикъла на втората група задачи.

На кратко може да се каже, че ако тялото на цикъла е конвейерно реализирано чрез *m* на брой степени, структурата от фиг.2 може да работи по едновременното изпълнение на (*m-1*) на брой задачи. В процеса на излизане от цикъла на завършилите задачи, в него се зареждат също толкова нови задачи. Ако се приеме, че броят на циклическите повторения е *n* на брой, то латентността на цяла група от (*m-1*) задачи е почти колкото за една единствена задача. Процесът на излизане на задачи от цикъла, заедно с процеса на зареждане, може да бъде представен чрез времедиаграмата от фиг.5.



Фиг. 5. *Циклическа активност на 3 групи от по (m-1) задачи*

На времедиаграмата активните изчисления по задачите от дадена група се застъпват във времето, когато те излизат от цикъла, с навлизането на следващата група задачи. Латентността τ на всяка задача се представя със следния израз

$$\tau = n.\sum_{k=1}^{m} t_k \quad , \tag{1}$$

където с t_k е означена латентността на k-тото звено.

Ако се приеме (в най-лошия случай), че латентността t_k на всяко звено е константа, равна на максимално възможната за съответното звено, следва че и оценката (1) е константа. Последното дава правото да се твърди, че при тази организация, конвейерната циклическа структура е най-малко (*m-1*) пъти по-производителна по сравнение с еднозадачната организация.

5. Място и време на събитията

За да бъдат изявени настъпващите събития, ще бъде анализиран по-подробно процеса на преминаване на задачите през изходната и входната точки. Трансферът на данните в тези точки зависи от стойността на условието *CEC*. Следва да се помни, че условието *CEC* в текущия такт се отнася само за задачата, достигнала изходната точка, и че в следващия такт в същата точка ще бъде изчислено условието на следващата задача. Анализът на функционирането на циклическата структура откроява три различни ситуации във времето:

1. Когато в тялото на цикъла навлизат новите задачи;
2. Когато тези задачи се превъртват последователно и многократно;
3. И когато задачите започват да излизат от цикъла.

След изпълнение на циклическите изчисления за няколко групи задачи, както е показано на фиг.5, първият и третият случаи се обединяват. Между тях може да се видят отличия само при първоначално зареждане на първата група или когато при излизане на дадена група, се окаже че тя е последната.

В изходно състояние всеки конвейер се характеризира с всеобща готовност на микроконвейерните звена. Заявки няма и всички конвейерни автомати връщат един към друг сигнали, потвърждаващи готовността. В това състояние, навлизането на много задачи в тялото на цикъла, както е показано на фиг.4, изисква данновите връзки във входната му точка да са такива, че да осигурят зареждане на циклическата конвейерна структура с максимално възможния брой задачи. Докато това не се постигне, обратната връзка трябва да бъде трайно изключена. От посоченото следва, че в конвейерната структура трябва да се въведе нов елемент, чиято задача ще бъде да покаже момента, в който циклическото тяло е заредено с необходимия брой задачи. Синтезът на този елемент е следваща, трета задача. Тъй като при начално зареждане на задачи в конвейера все още не е била изчислявана стойност на условието за край на цикъла *CEC*, то управлението на данновите връзки във входната точка на цикъла по това време ще се осъществява именно от този елемент. В момента, в който задача 1 достигне звено 6 и за нея бъде изчислена стойност за условието *CEC*, управлението на данновите връзки ще поеме тази именно стойност.

По-детайлният анализ на последните два такта (k+7) и (k+8) от фиг.3 показва, че след като задача 1 в такт (k+5) попадне в звено 6, започва изчисляването на стойността на условието *CEC*. Едновременно с това останалите задачи слизат в тялото, а свободното звено *free* се премества в обратна посока. В такт (k+6) или (k+7) стойността на условието е вече изчислена и тя е *CEC*=0. Тази стойност определя данновите връзки в изходната и във входната точки, като образува обратната връзка. С това в такт (k+8) става възможен преходът на задача 1 от звено 6 в звено 3.

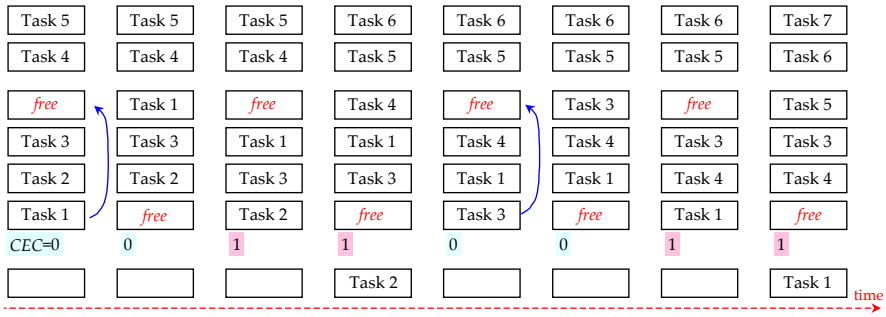
В началото на фиг.4 е изобразен обратният случай, когато *CEC*=1. При тази стойност данновите връзки в изходната и входната точки на цикъла трябва да се определят така, че обратната връзка да бъде изключена. Така задача 1 ще излезе от цикъла и ще слезе надолу в конвейера в звено 7, а в същото време в звено 3 ще бъде заредена новата задача 4. В края на процеса, изобразен на фиг.4, в циклическата структура е заредена новата група задачи №(4,5,6). И тъй като за задача 4, която е в звено 6, е изчислена стойността *CEC*=0, обратната връзка отново е включена и тази нова група започва своето *n*-кратното превъртване в тялото на цикъла.

6. Цикъл с предварително неизвестен брой повторения

Дълготрайността на стойностите на условието за край на цикъла е характерна особеност на функционирането на разгледаната по-горе разновидност. Значително по-динамично е то когато структурата съответства на алгоритъм с неизвестен брой повторения. В този случай представата за известна статичност, че задачите ще навлязат последователно, че ще се превъртват заедно и ще излязат заедно, в реда, в който са влезли, като цялостна и постоянно задържаща се в цикъла група, губи смисъл. Конвейерното изпълнение на няколко задачи в цикъл с неизвестен брой повторения не може да запази първоначалния

състав на групата от (*m-1*) на брой задачи. Тъй като броят на повторенията за всяка задача ще бъде различен, групата ще има променлив състав във времето. Последният ще се променя вследствие на излизане на задачи от цикъла и влизане на други, които ще ги заместват.

Ходът на изпълнението на всяка отделна задача зависи от текущата стойност на условието за край *CEC*, която тя генерира. Ето защо е важно да се анализират събитията в двата случая на това условие, както по отношение на изходната, така и по отношение на входната точки. Приема се, че влезлите в тялото на цикъла задачи 1, 2 и 3 са изпълнили няколко повторения и се намират в положението, показано на фиг.6, в най-лявата колонка.



Фиг. 6. *Процес на излизане от цикъла*

Нека за задача 1 звено 6 (фиг.2) е генерирало условието *CEC*=0. В резултат на това тя преминава по обратната връзка в звено 3. В последствие задачите последователно слизат от звено на звено, като в третата колонка се вижда, че звено 3 отново е свободно. В звено 6 се намира задача 2, за която *CEC*=1. Задача 2 излиза от цикъла, а в него влиза нова задача 4. Задача 3 се превърта, а задача 1 излиза от цикъла. Така в края на рисунката в тялото на цикъла остава да се превъртва групата от задачи (4, 3, 5). Изложеният току що анализ изявява още един нов проблем. Същността на този проблем се състои в реда на излизащите от цикъла задачи. Тъй като броят на повторенията не е известен и е индивидуален за всяка задача, не може да се очаква редът при зареждането им да се повтори при излизането им от цикъла. Така в отделните моменти съставът на групата в цикъла се променя. В заключение на казаното следва да бъде формулирана за този раздел нова четвърта задача, състояща се във възстановяване (при необходимост) на реда на слизащите от конвейера задачи. Тази задача вече има решение [4] и няма да бъде дискутирана тук. По-долу са представени решенията на формулираните задачи, които позволяват създаването на метод за асинхронна микроконвейерна реализация на алгоритмични структури от вида цикъл.

7. Конвейерен автомат на звеното с условие за край на цикъл

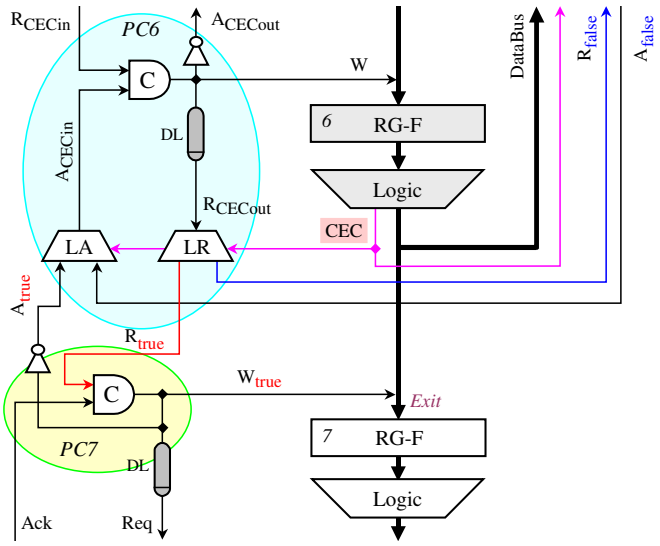
Конвейерният автомат на звеното, което генерира условието за край на цикъла *CEC* (изходното звено), по същество е свързан са два други конвейерни автомата. Единият автомат управлява звеното, което се намира по изхода от цикъла (звено 7, фиг.2). В това звено попадат резултатите, получени от циклическите изчисления, когато условието за край бъде изпълнено *CEC*=1. По същество, това е изходът "*истина*" (*CEC=true*). Другият автомат е този, който управлява звеното във входната точка на цикъла. Връзката с него се осъществява по изхода "*лъжа*" на условния преход (*CEC=false*).

7.1. Конвейерен автомат с 2-фазов протокол за трансфер на данни

Синтезираната логическа структура на конвейерен автомат *PC6* с 2-фазов протокол, управляващ звеното, което генерира условието за край на цикъла, е представена на фиг.7.

Разглеждано като част от логическата структура на конвейера, микроконвейерното звено 6, което генерира условието за край на цикъла *CEC*, независимо какъв е той по вид, има аналогичните проблеми на всяко друго звено, генериращо условие за преход. По тази

причина структурното решение за автомата на разглежданото тук звено наподобява решението, което е изложено в [2].



Фиг. 7. *Логическа структура на конвейерен автомат с условие за край на цикъл*

Както се вижда от фигурата, сигналът потвърждение A_{CECin} е функция, реализирана от схемата LA (*Logic Acknowledgement*) от потвържденията A_{true} и A_{false}, а двете заявки R_{true} и R_{false} се формират от сигнала R_{CECout} в схемата LR (*Logic Request*).

7.2. Синтез на сигнал потвърждение

Превключването на конвейерния автомат в последното звено от тялото на цикъла е функция от два входни сигнала: A_{true} и A_{false}. Последните оповестяват готовността на всеки от клоновете по отделно, т.е. те са "родители" на сигнала A_{CECin}. Поради значително по-голямата сумарна латентност на циклическата структура като цяло, в сравнение с тази на микроконвейерното звено, намиращо се след изхода от цикъла, може да приемем, че по време на циклическите изчисления, потвърждението A_{true} е постоянно подадено и очаква момента, когато ще бъде използвано. Потвърждението A_{true} идва от звено 7, което се намира след циклическата структура, което поради закъснението в последната се предполага, че вече е завършило своето изчисление. До този момент обаче актуалното потвърждение идва от звеното в клона "лъжа", а то е A_{false}. Това е така по причина на многократните завъртания на цикъла по обратната връзка. В тези условия, актуалното потвърждение пристига в отговор на заявката R_{false} в момент, когато логическата стойност на условието *CEC* е стабилно установена във времето. От тук следва, че логическата схема LA ще осъществява само мултиплексиране на двете входни потвърждения, според следната логика.

$$A_{CECin} = A_{true} \cap CEC \quad \cup \quad A_{false} \cap \overline{CEC} \ . \tag{2}$$

Сигналът A_{CECin} ще има стойността на идващото от съответния клон потвърждение, в следствие на което не е необходимо конвертиране на тази стойност.

Следва да бъде отчетен и събитийния характер на новите стойности. Моментът на поява на новата логическа стойност на условието за преход *CEC* и моментът на поява на заявката R_{CECout}, се състезават във времето. Резултатът от това състезание е или равен или се печели от *CEC*. Мултиплексорът на потвържденията LA, който ще се управлява от условието *CEC*, трябва да се превключи не в момента на възникване на новата стойност на условието *CEC*, а в момента на появата на заявката R_{CECout}. Това налага състезателността да бъде отстранена, което можем да постигнем единствено с помощта на тригер. В момента на възникване на заявката R_{CECout} този тригер трябва да фиксира стойността на условието *CEC*

и да я поддържа до следващия такт. Така той ще има още една положителна роля – ще предпазва мултиплексора LA от нежелани превключвания по време на самото изчисление на следващата нова стойност на *CEC*.

7.3. Генериране на заявки към разклоненията

Конвейерният автомат, управляващ звеното с условен преход, разпространява към следващите звена заявка, отбелязана като R_{CECout}. Заявката не може да се подаде директно към входовете на конвейерните автомати в началото на всеки от алтернативните клонове. Съответните запитвания, които следва да получат конвейерните автомати, са означени с R_{true} за клона "истина" и с R_{false} за клона "лъжа" и са функция на логическата схема LR (фиг.7). Непосредственото подключване на заявката R_{CECout} не е възможно, защото в точката на разклонението тя влиза в сложна функционална връзка с логическата стойност на условието *CEC* от една страна, и с текущото (завареното) състояние на конвейерния автомат в началото на всеки клон, от друга страна. Това пък се налага от вида на самите автомати, които както вече беше споменато, използват 2-фазов протокол за управление на трансфера. Последното означава, че всяко тяхно превключване (0→1 и 1→0) причинява запис в регистрите фиксатори и старт на изчисленията в звената. Ако се разглежда превключване в началото на клон от микроконвейера, то освен логическата стойност на условието за преход *CEC* (0 или 1) следва да се отчете и завареното състояние на съответния автомат. Ще отбележим, че С-елементът в схемата на автомата се превключва както при съчетаване на две входни единици, така и на две нули. С други думи сигналите R_{true} и R_{false} са функции не само от превключването на сигнала R_{CECout}, но и от стойността на условието *CEC*, а така също и от състоянието на конвейерните автомати на входа на разклоненията. Например, ако стойността на условието за преход е нула (*CEC*=0), това означава, че изчисленията трябва да продължат в клона "лъжа", т.е. в звено 3, като се затвори за пореден път обратната връзка на цикъла. Единственото, което различава този тип разклонение от обикновеното разклонение, е точно тази няколкократна повторяемост на ситуацията, представляваща същността на структурата цикъл. Ако състоянието на автомата *PC3* в този клон, в който ни насочва условието, е единица, което се поддържа във времето до този момент от стойността R_{false}=1, то той трябва да се превключи в ново състояние нула, за да стартира чрез заден фронт на сигнала *W* тези изчисления. За да стане това, на входа на този С-елемент трябва да се съчетаят две нули. Не трябва да се забравя и факта, че всяка нова стойност на сигнала R_{CECout} (и 0 и 1) представлява по същество нова заявка, излизаща от автомата *PC6* на звено 6.

Логиката, която току що беше пояснена, е изразена чрез таблиците на истинност 1 и 2, в които сигналите W_{true} и *W* представят състоянието на съответните С-елементи. Тук следва допълнително пояснение. За разлика от случая на обикновено разклонение, където се използваха два сигнала W_{true} и W_{false}, тук вторият сигнал е означен само с *W*, т.е. без подсказката *fasle*. Това е така, защото звеното във входната точка не е обикновено начално звено на клон в алгоритъм. В нашият случай входното звено е звено в обща точка. То записва нови данни както в качеството си на начално звено в клон "лъжа", така и като обикновено звено при първо навлизане на задачи в циклическата структура, ето защо сигналът за запис на това звено е означен само *W*.

Въз основа на таблиците на истинност са синтезирани следните логически функции:

$$R_{true} = CEC \oplus W_{true} \quad . \tag{3}$$

$$R_{false} = \overline{CEC \oplus W} \quad . \tag{4}$$

Както се вижда, логиката на заявките R_{true} и R_{false} не зависи от заявката R_{CECout}, което се очакваше. Зависимостта на заявките R_{true} и R_{false} не е от стойността на сигнала R_{CECout}, а от времето, т.е. от моментите на неговото превключване. Следва да се разбира, че смяната

на стойността на R_{CECout}, т.е. появата на негов фронт, маркира момента във времето, в който операционната логика на това звено завършва изчисленията си. Този момент не е задължително да съвпада с появата на истинната стойност на условието *CEC*. В зависимост от сложността на изчислението на *CEC*, в общия случай следва да се приема, че истинната стойност на *CEC* може да се появи и по-рано във времето спрямо новия фронт на сигнала R_{CECout} или най-късно едновременно с него и никога по-късно от него. В една непосредствена реализация на (3) и (4), по-ранното явяване на *CEC* ще доведе до по-ранно формиране на стойностите на заявките R_{true} или R_{false}, а от там и до по-ранно стартиране на съответния клон на конвейера. Това стартиране ще започне със запис на данни в регистъра фиксатор, а те в общият случай все още няма да са достигнали във времето истинните си стойности. Така най-вероятно изчисленията могат да стартират с грешни данни.

Таблица 1. Заявка към конвейерен автомат в клон “лъжа”

CEC	*W*	R_{CECout}	R_{false}
0	0	появява се заден фронт 0	1 , превключва се
0	0	появява се преден фронт 1	1 , превключва се
0	1	появява се заден фронт 0	0 , превключва се
0	1	появява се преден фронт 1	0 , превключва се
1	0	появява се заден фронт 0	0, не се превключва
1	0	появява се преден фронт 1	0, не се превключва
1	1	появява се заден фронт 0	1, не се превключва
1	1	появява се преден фронт 1	1, не се превключва

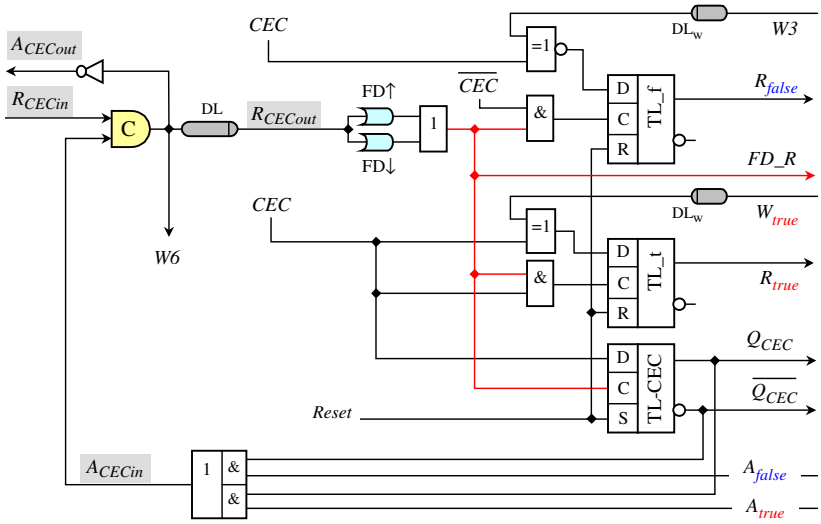
Таблица 2. Заявка към конвейерен автомат в клон “истина”

CEC	W_{true}	R_{CECout}	R_{true}
0	0	появява се заден фронт 0	0, не се превключва
0	0	появява се преден фронт 1	0, не се превключва
0	1	появява се заден фронт 0	1, не се превключва
0	1	появява се преден фронт 1	1, не се превключва
1	0	появява се заден фронт 0	1 , превключва се
1	0	появява се преден фронт 1	1 , превключва се
1	1	появява се заден фронт 0	0 , превключва се
1	1	появява се преден фронт 1	0 , превключва се

Основният извод, който се налага от така изложените съображения е, че формулите (3) и (4) определят стойностите на заявките, но моментът, в който те следва да се появят и да подействат, се определя от момента на превключване на сигнала R_{CECout}. Или казано по друг начин, новите стойности на заявките R_{true} или R_{false} трябва да се явят в отговор на фронт в сигнала R_{CECout}. Последното означава, че формирането на заявките R_{true} и R_{false}, не е възможно да бъде постигнато единствено с комбинационна логика.

Изложените разсъждения доказват, че времевата зависимост на изчислените стойности (3) и (4) може да се реализира само с помощта на запомнящ елемент – тригер. Тъй като запис трябва да се извършва при всяко превключване на С-елемента, синхронизиращият тригер трябва да бъде от тип DEDTFF (D-тригер, работещ и по двата

фронта). В окончателната логическа схема на конвейерния автомат *PC6* (фиг.8) е представено едно предпочетено тук решение, основаващо се на обикновен D-*Latch* тригер и два детектора на фронт – FD↑ за преден и FD↓ за заден. Логическата схема, която беше отбелязана с LR във фиг.7, съдържа С-елемента на звеното с условен преход, схемата LA, обединяваща потвържденията A_{true} и A_{false} , както и двете тригерни схеми, генериращи заявките R_{true} и R_{false}.



Фиг. 8. *Логическа схема на 2-фазов конвейерен автомат за звено с условие за край на цикъл*

Както се вижда от схемата, пулс-генераторите се обединяват в схема ИЛИ, чийто изход е означен *FD_R*. Този сигнал реализират запис по входа С на съответния тригер – в тригер TL-CEC винаги, а в другите два тригера, според стойността на условието за край. Изчислената според (3) логическа стойност на заявката R_{true} постъпва по входа D и се съхранява в тригера до следващия път, когато ще бъде избран същия клон. Сигналът *Reset* е необходим в началния момент, в който всички конвейерни автомати принудително се установяват в изходно състояние. Аналогична логическа схема формира според (4) заявката R_{false} към конвейерния автомат в клон "лъжа".

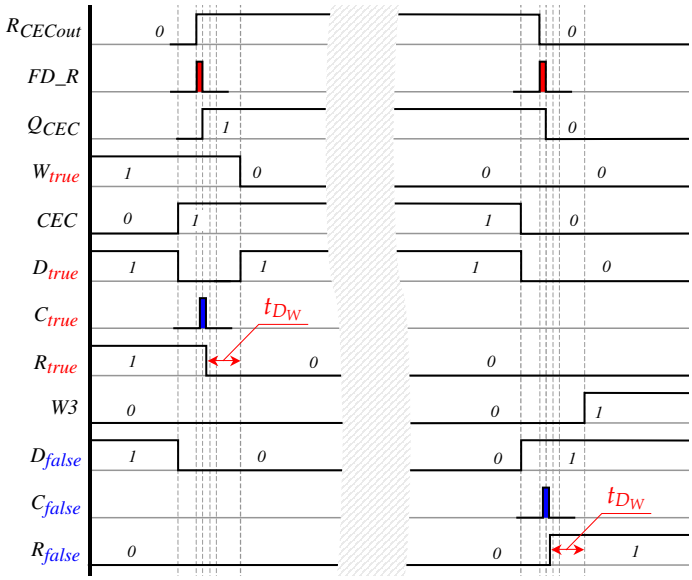
Тъй като в отговор на изпратената заявка R_{true} или R_{false} съответният конвейерен автомат ще се превключи и ще върне по обратната връзка нова стойност на сигнала *W*, която застрашава надеждността на записа в D-тригера TL_t или TL_f, то това налага задържане във времето, докато изчезне импулсът за запис от входа С. Закъснението се осигурява от елемента DL_W, което се осигурява конструктивно от неравенството $t_{DLw}>t_{FD}$.

В схемата е включен тригер TL_CEC, който има за задача да запомни логическата стойност на условието за край на цикъла *CEC* в момента, когато звеното завърши своите изчисления, т.е. когато се появи заявката R_{CECout}. Така той няма да допусне новата стойност на условието *CEC* да манипулира преждевременно мултиплексора на потвържденията, поддържайки стабилна текущата стойност до края на следващото изчисление.

Изходното състояние на тригерите TL_t и TL_f е без значение, но е прието те да се нулират от сигнал *Reset*. В същото време, принудителното състояние на тригер TL_CEC е единица, което се изисква от конвейерния автомат на звеното във входната точка. Това положение е пояснено чрез времедиаграмата от фиг.9.

Времедиаграмата изобразява две превключвания на автомата на изходното звено. Първото е при стойност на условието за край *CEC*=1. Изобразените стойности до този момент на сигналите W_{true} и *W* са приети за единица и нула съответно, като възможни. Вижда се, че стойността на заявката R_{false} и състоянието на автомата *W3* в клона "лъжа" не се променят. В същото време заявката към клон "истина" се превключва. В резултат на това

се превключва и съответният автомат – сигналът W_{true}=0. Новото му състояние е отразено със закъснение t_{DW}.



Фиг. 9. *Две възможни превключвания на автомата на изходното звено*

Второто превключване е при стойност на условието за край *CEC*=0 и следователно актуалната заявка ще бъде насочена в клона "лъжа". Така заявката R_{false} се превключва в единица. В резултат на това автоматът на звено 3 също се превключва (*W3*=1), което е отразено във времедиаграмата със същото закъснение t_{DW}.

8. Конвейерен автомат на звеното във входната точка

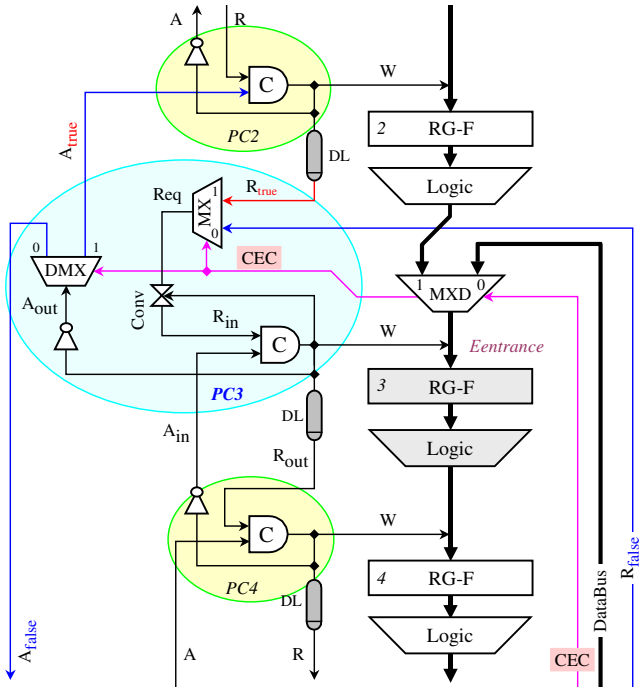
Общата характеристика на звеното във входната точка на цикъла беше изложена в началото. Данновите входове на регистъра фиксатор RG-F в това звено са два. Те трябва да се мултиплексират в зависимост от логическата стойност на условието за край на цикъла *CEC*. Като се има предвид, че тази стойност се съхранява в тригер TL_CEC, който е част от схемата на автомата, управляващ изходното звено, то неговото състояние може да се използва и в схемата на автомата на звеното във входната точка. Това е така, защото съответното зареждане с данни на това звено става по времето на актуалната стойност на условието *CEC*.

8.1. Конвейерен автомат с 2-фазов протокол за трансфер

От фиг.2 се вижда, че автоматът, който управлява звеното в тази точка, е свързан със следващото звено традиционно чрез сигналите *Request* (R_{out}) и *Acknowledgement* (A_{in}). Двустранна е обаче връзката му с автоматите на предходни звена. Фиг.10 представя синтезираната за този автомат логическа структура.

Заявката, която трябва да получи автоматът на звеното във входната точка на цикъла, е означена R_{in}. Схемата на конвейерния автомат *PC3*, който управлява звеното във входната точка на цикъла, съдържа мултиплексора MX. Той обединява двете заявки към това звено, означени R_{true} и R_{false}. Мултиплексорът MX се управлява от условието за край на цикъла *CEC*, което се поддържа стабилно във времето от тригер TL-CEC (фигура 3.8.8). Това означава, че мултиплексорът ще бъде правилно превключен както при първо влизане в цикъла, така и при всяко завъртане на изчислителния процес по обратната връзка. Това е така, защото след всяко излизане от цикъла, което се реализира при стойност на условието за преход *CEC*=1, тригер TL-CEC остава в единично състояние. Именно това състояние държи конвейерния автомат на звеното във входната точка включен към предходния

автомат *PC2*, т.е. изгражда връзката $Req=R_{true}$. Това състояние обяснява и защо сигнал *Reset* е подаден към S входа на тригер TL-CEC. Алтернативната връзка $Req=R_{false}$ се поддържа по време на циклическите повторения, когато *CEC*=0. Същото важи и за мултиплексора на данновите шини MXD.



Фиг. 10. *Логическа структура на конвейерен автомат във входна точка на цикъл*

Генериране на заявка

Логическата стойност на заявката *Req*, такава каквато излиза от мултиплексора MX, не винаги е тази, която е актуална за Мюлер С-елемента, на чийто вход R_{in} следва да бъде подадена. Това неудобство се дължи на 2-фазовия протокол на автомата, който разглеждаме. Според него всяко състояние на С-елемента е работно и то трябва да се изменя на противоположното при всеки старт на микроконвейерното звено. Това означава, че ако С-елементът е в състояние нула (*W*=0), то при нова заявка той трябва да се превключи в състояние единица. Това е възможно само ако тази заявка се появи като единица. Аналогично е положението за обратното превключване.

Възможното несъответствие между логическата стойност на новата заявка R_{out} и очакваната стойност R_{in} налага нейното преобразуване. Така тази стойност ще бъде съобразена със завареното състояние на С-елемента в автомат *PC3*. За целта съставяме таблица на истинност.

Таблица 3. Логическа стойност на заявката R_{in} и потвърждението A_{out}

Заявката *Req* пристига с нова стойност:	С-елементът е заварен в състояние:	С-елементът трябва да се превключи в състояние:	Следователно R_{in} трябва да има стойността:	Трябва да бъде върнато потвърждение A_{out} със стойност:
0↓	0	↑1	1 ($\overline{Req}$)	1
0↓	1	0↓	0 (Req)	1
↑1	0	↑1	1 (Req)	0
↑1	1	0↓	0 ($\overline{Req}$)	0

От таблицата се вижда, че когато логическата стойност на новата заявка съвпада със състоянието на С-елемента (първи и четвърти ред), необходима е инверсната стойност, а когато са различни – необходимата стойност е същата. Тази логика

$$R_{in} = \overline{(Req \oplus W)} \cap \overline{Req} \ \cup \ (Req \oplus W) \cap Req = \overline{W} \qquad (5)$$

изразява формално схемата на конвертора Conv (фиг.10). Според (5) входната заявка има пряка връзка с изходното състояние, но тя следва да се осъществява и актуализира само в момента, в който се появява стойността на условието *CEC*. Този момент се бележи от фронтовете на превключване на заявката R_{CECout} (фиг.8), които са реализирани от сигнала, означен *FD_R*. Казаното означава, че връзката (5) трябва да се осъществи чрез тригер, т.е. заявката R_{in} трябва да се подаде към С-елемента в нужния момент.

Синтез на сигнал потвърждение

След като конвейерният автомат на звеното във входната точка се превключи, той трябва да върне сигнал за потвърждение A_{out} към звеното, от което е получил заявката. Звеното е било определено от стойността на условието *CEC*, следователно пак то определя направлението на актуалното потвърждение. За целта в схемата е включен демултиплексорът DMX. Така, когато *CEC*=0, стойността A_{out} ще бъде насочена към автомата в клона "лъжа", т.е. $A_{false} \equiv A_{out}$. В обратния случай, когато *CEC*=1, стойността A_{out} ще бъде насочена към автомата в клона "истина", т.е. $A_{true} \equiv A_{out}$.

При чистото демултипликсиране на сигнал A_{out} обаче, към неактуалния клон винаги ще се изпраща нова стойност нула. От една страна тази нула може да предизвика неправомерно превключване на конвейерния автомат в този неактуален клон, а от друга страна, тъй като клонът е неактуален, то стойността на потвърждението към него трябва да остава неизменна се във времето, докато той чака обръщение. Последното означава, че стойността на върнатото преди това актуално потвърждение следва да бъде запомнена и поддържана във времето от тригер. Този тригер трябва да променя състоянието си в съответствие с новата стойност на сигнала потвърждение, само когато клонът за който той работи, бъде определен за актуален. С други думи тригерите с това предназначение трябва да са на брой два.

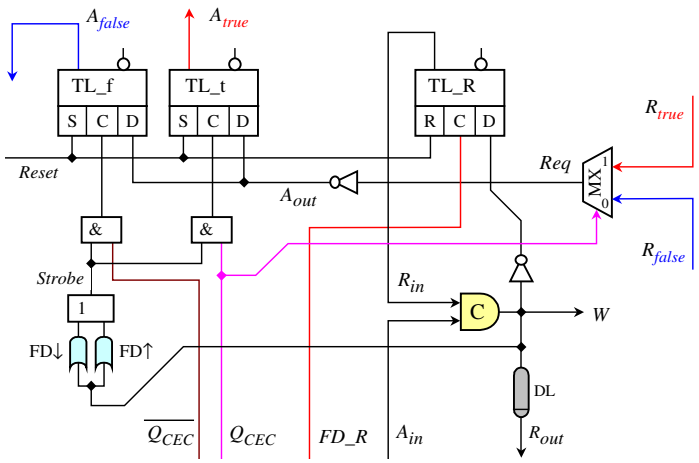
Следващата задача, която трябва да бъде решена, е каква трябва да бъде стойността на сигнала потвърждение. Тъй като конвейерният автомат *PC3* на звеното във входната точка би могъл преди този такт да се е превключвал неизвестен брой пъти, то връщаната от него в текущия такт стойност A_{out} може да не съответства на очакваната от автомат *PC2* или от автомат *PC6* (фиг.2). Налага се преобразуване на стойността на сигнала A_{out} в необходимата, което се дължи на 2-фазовия протокол за управление на данновия трансфер.

Логиката за определяне на актуалната стойност на потвърждението се основава на логиката, по която се превключва Мюлер С-елементът, а именно (вижте например първи ред в таблица 3), когато автоматът *PC3* се е превключил в резултат на получена заявка със стойност нула, към автомата, подал тази заявка трябва да бъде върнато потвърждение със стойност единица. Изказаната логика се съдържа в таблица 3, а решението е

$$A_{out} = \overline{Req} \qquad (6)$$

Окончателната принципна логическа схема на конвейерния автомат на звеното във входната точка е представена на фиг.11.

Схемата реализира функциите (5) и (6) като отчита времевата им зависимост от събитията, които ги формират. Така стойностите на потвържденията A_{false} и A_{true} се записват в съответния тригер, когато се превключи С-елементът на автомата, а заявката R_{in}, която той получава, се записва в тригер TL_R от сигнал *FD_R*. Изходното състояние на автомата определя сигнал *Reset*.

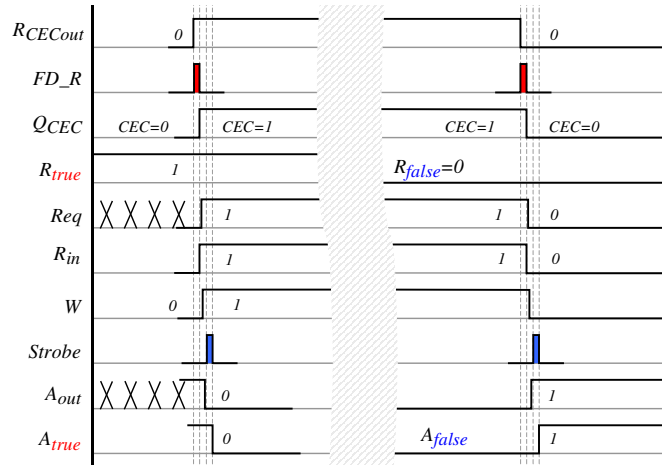


Фиг. 11. *Логическа схема на 2-фазов конвейерен автомат за входно звено на цикъл*

Функционирането на автомата на звеното във входната точка е илюстрирано с времедиаграмата от фиг.12, при следните начални условия: нека при предходното превключване автоматът да е записал данни по обратната връзка. Това означава, че условието за край е било *CEC*=0, откъдето следва, че в тригер TL_R има записана нула, т.е. R_{in}=0. Следва, че С-елементът е в състояние 0 (*W*=0).

Прието е, че при новото превключване автоматът ще запише данни по правата връзка, тъй като ще приемем още, че условието за край ще бъде *CEC*=1. Прието е също, че заявката от предходното звено 2 има стойност R_{true}=1. Последното означава, че тригер TL_t поддържа стойност 1, т.е. A_{true}=1. Новото превключване започва с превключването на заявката R_{CECout} в изходното звено, което ще приемем за единица.

На времедиаграмата, след превключването на заявката R_{CECout} в единица, е изобразено следващото й превключване, което е в нула. При това превключване е прието, че стойността на условието за край на цикъла е *CEC*=0, което означава, че този път входното звено ще приеме данни по обратната връзка. Вижда се, че превключването зависи от стойността на заявката R_{false}, която е приета да е нула (R_{in}=R_{false}=0). Тази стойност е поддържана във времето до този момент от стойността A_{false}=0. След превключването на автомата (*W*=0), в клона "лъжа" към автомата на изходното звено се връща потвърждението A_{false}=1.

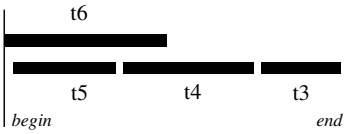


Фиг. 12 *Две последователни превключвания на автомата на входното звено*

9. Задача трета

По-горе в раздел 5 беше изявена задача, която получи пореден номер три. Тя изисква, в началните тактове, при първоначално зареждане на конвейера със задачи, обратната

връзка на циклическото тяло да бъде изключена. При това, данновите връзки на входното звено на тялото на цикъла трябва да се поддържат включени към предходното звено, с което се осигурява зареждане на многостепенното тяло с максимално допустимия брой задачи. Беше определено, че ако степените в конвейеризираното тяло са *m* на брой, то задачите трябва да бъдат (*m-1*) на брой. Имат се предвид регистри фиксатори от тип *Latch*. Обратната връзка трябва да бъде включена веднага след зареждане на този брой задачи. От този момент нататък данновите връзки в изходната и входната точки трябва да се управляват от условието за край на цикъла *CEC*. В началния момент, за който важи трета задача, протича процес на зареждане на циклическото тяло със задачи, който е илюстриран с фиг.3. Както е показано на фигурата, обратната връзка трябва да бъде включена точно в такт (k+7). Заедно с това управлението на входния мултиплексор в този момент трябва да се преотстъпи на условието за карай на цикъла *CEC*. Процесът на зареждане на циклическото тяло със задачи е съпроводен с изчислителен процес, който всяко звено провежда върху заредените в него данни. Анализира се такт (k+5) от същата фигура. В този момент задача 1 постъпва в изходното звено 6 и стартира своите изчисления. Интересното за тях е това, че те ще получат първата стойност на условието за край на цикъла *CEC*. Докато това условие се изчислява, в посока на входната точка се изкачва свободното звено, което в момент (k+7) се оказва в звено 3. Изчислението в звено 6 има латентност t6. В същото време, входното звено ще бъде свободно след време (t5+t4+t3), което е сумарната латентност на останалите предходни звена. След като звено 6 приеме данни от звено 5 и му върне сигнал потвърждение, последното прави същото, т.е. може да се приеме с първо приближение, че тези две съседни звена (6 и 5) стартират своите изчисления почти едновременно. Това ще позволи да се илюстрира казаното по-горе със следната рисунка, позволяваща да се сравнят възможните времеви закъснения



Тъй като в общия случай латентностите на звената при конкретните изчисления не са известни, а така също и броят на степените в конвейера не е известен, както и структурата му не е известна, то са възможни следните отношения на латентността на изходното звено t_m (с m беше означен броя на звената в тялото на цикъла)

$$\begin{cases} t_m < \sum_{k=1}^{m-1} t_k \ ; \\ t_m \geq \sum_{k=1}^{m-1} t_k \ . \end{cases} \qquad (7)$$

Първото отношение съответства на горната рисунка. За изчислителния процес това означава, че изходното звено е завършило своите изчисления и стойността на условието *CEC* е готова. Предхождащата структура от звена обаче не е завършила, а може би дори не е заредена до край със задачи. Това означава, че изходното звено ще остане в състояние на очакване, докато настъпи момента, в който ще се свърже с входното звено, от което ще получи сигнал потвърждение и където ще може да управлява входния даннов мултиплексор. Разрешение за това ще му даде решението на трета задача, което предстои да представим.

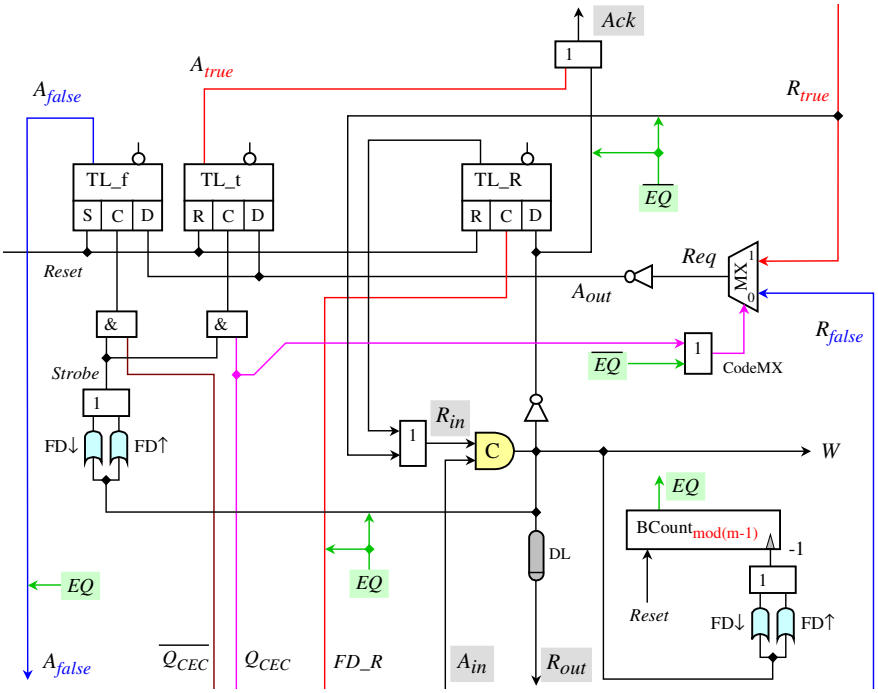
Второто отношение на (7) в общия случай е малко вероятно. Според него предходната структура от звена е заредена и е завършила изчисленията си, като очаква управление на обратната връзка от изходното звено. С други думи принудителното управление на входната точка е завършило и е прекратено.

Направеният анализ показва, че във всички случаи, при първоначално зареждане на

конвейерното тяло на цикъла, изходното звено все още не е в състояние да управлява автомата на входното звено и процесът на зареждане трябва да се контролира от друг елемент. Този елемент трябва да бъде двоичен брояч по модул (*m-1*), за да може да отброи необходимия брой заредени в тялото на цикъла задачи. Върху логическата схема от фиг.11 е проведен допълнителен синтез, в резултат на което е получена окончателната принципна логическа схема на конвейерния автомат, който управлява звеното във входната точка на цикъла, представена на фиг.13. В схемата се вижда споменатият брояч BCount, който е декрементен. Началното му съдържание е числото (*m-1*), което се установява по сигнал *Reset*. Нулевото съдържание на брояча се разпознава от дешифратор, който генерира единица на изхода *EQ*. Стойността на този сигнал се използва за управление на посочените в схемата връзки. Правата фаза на сигнала *EQ* забранява разпространението на сигналите A_{false} , *FD_R* и *W* в показаните направления. Докато броячът не се нулира, сигналът има стойност нула (EQ=0). Инверсната фаза на сигнала *EQ* забранява разпространението на сигналите R_{true} и $\overline{W}$. Мултиплексорът MX се управлява от функцията *CodeMX*

$$CodeMX = CEC \cup \overline{EQ} \ . \tag{8}$$

Тази логическа функция изчислява стойност единица винаги, когато съдържанието на брояча е различно от нула, т.е. по време на зареждането на тялото на цикъла със задачи. Докато тази функция има стойност единица, мултиплексорът MX поддържа вход 1. Така заявката R_{true} достига входа R_{in} на Мюлер С-елемента. Същата функция трябва да управлява и данновия мултиплексор MXD (фиг.10). Докато броячът не се нулира, автоматът на звеното във входната точка комуникира с предходното звено 2 (сигнали *Ack* и R_{true}) и със следващото звено 4 (сигнали A_{in} и R_{out}) от тялото на цикъла. Тригерите TL_f, TL_t и TL_R не се превключват и остават в изходно състояние, определено от сигнал *Reset*. Така, тъй като автоматът на изходното звено 6 ще получава стойност A_{false}=0, то ще остава в изходно състояние. В момента, в който броячът се нулира, сигналът *EQ* получава стойност единица. Забранените до момента връзки се разрешават, а разрешените се забраняват. След достигане на съдържание нула броячът не трябва да функционира. Това означава, че той не трябва да бъде кръгов брояч.



Фиг. 13. *Окончателна принципна логическа схема на 2-фазов ковейерен автомат за входно звено на цикъл*

След началното зареждане на задачи в тялото на цикъла, когато задача 1 завърши своите изчисления в звено 5, тя ще поиска да се прехвърли в последното звено 6, подавайки заявка към автомата му. Този автомат няма да може да удовлетвори тази заявка, тъй като достъпът на потвърждението A_{false} е блокиран от сигнала EQ=0, въпреки, че тригер TL_f е в единично състояние.

Докато не бъде заредено звеното във входната точка с последната (m-1)-ва за тялото на цикъла задача, забраната от сигнала EQ ще е в сила. Тази забрана ще бъде снета в момента, в който звеното бъде заредено с последната задача, тъй като тогава броячът ще се нулира. С появата на сигнал EQ=1 задача 1 ще премине в последното звено и ще започне изчислението на първата стойност на условието CEC. През това време следващите задачи ще слязат надолу и когато звеното във входната точка се освободи, в него ще постъпи задача, която ще определи условието CEC. От този момент нататък схемата от фиг.13 ще функционира аналогично на схемата от фиг.11.

10. Заключение

В смисъла на това изследване не са открити сходни публикации. Многотактовите циклически структури, изследвани тук, не винаги могат да бъдат разгънати и така тяхната реализация не може да бъде да бъде по-проста. Ето защо това изследване е актуално.

Както беше посочено по-горе, представеното комплексно решение не зависи от вида на цикъла. В хода на анализа бяха изявени няколко нови задачи, решението на една от които е представено в предходна публикация. Получените решения на останалите задачи са приложими в микроконвейери, използващи конвейерни автомати както с 2-фазов протокол за трансфер на данни, така и с 4-фазов.

Приложението на асинхронната конвейерна реализация върху циклически структури води до значително повишаване на производителността на подобни изчислители. В циклическото тялото могат да съществуват достатъчно сложни и общи по вид алгоритмични последователности.

Наличието на решения за конвейерните автомати в изходната и входната точки на циклическото тяло в конвейера, които не зависят от броя на степените в циклическото тяло, както и от вида на микроконвейерните звена, по същество създава метод за хардуерна реализация на такива общи алгоритмични структури.

От друга гледна точка, получените решения създават възможност за изграждане на универсална структура. Конвейерното тяло на тази структура може лесно да бъде подменяно. Това е възможно дори в процеса на реално функциониране. Възможността структурата да бъде реконфигуруема, представлява едно изключително полезно следствие, което предстои да бъде изследвано.

Литература:

[1]. Tyanev, D.S., *Four-phase micro-pipeline with one-cycle and multi-cycle micro-pipeline sections*, Computer Science and Technologies, Publication of Computing and Automation Faculty Technical University of Varna, Bulgaria, ISSN 1312-3335, VII 1/2009, pp. 3-12.

[2]. Tyanev, D.S., *Branch management in asynchronous micro-pipeline,* Computer Science and Technologies, Publication of Computing and Automation Faculty Technical University of Varna, Bulgaria, ISSN 1312-3335, VII 2/2009, pp. 3-12.

[3]. Tyanev, D.S., S.I. Kolev, D.V. Yanev, *Race Condition free Asynchronous Micro-Pipeline Units*, International Conference on Computer Systems and Technologies – CompSysTech'10, 17-18 June 2010, Sofia, Bulgaria.

[4]. Tyanev, D.S., V.T. Bozhikova, S.K. Gerganov, P.G. Georgiev, *Algorithm for micropipeline buffer control,* Applied Technologies and Innovations, ISSN 1804-1191, vol. 4, Issue 1, April 2011, pp.12-21.