

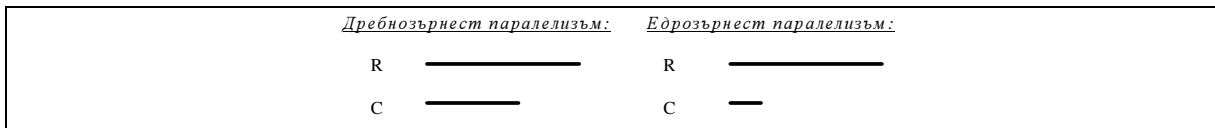
Лабораторно упражнение No 3

Изследване на организацията и системните характеристики на базови конвейерни архитектури

1. Теоретична постановка

1.1. Основни характеристики на паралелни КА и дефиниционна област

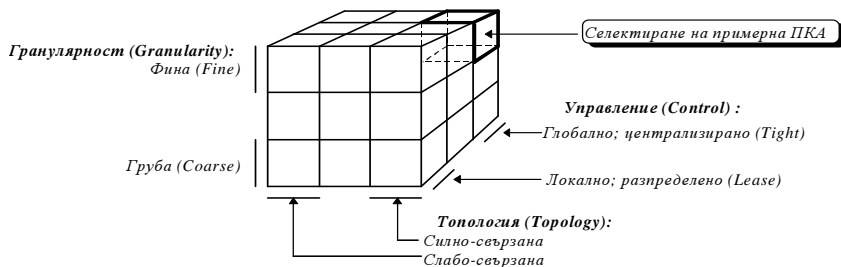
♦ **Гранулярност (G)** - мярка за атомарността на обработваните информационно-програмни единици (задачи, подпрограми и функции, векторни данни, инструкции и микроинструкции). Оценява се чрез отношението между времето R за цялостната реализация на процесите и частта C, която е необходима за осъществяване на комуникациите. Това определя базовите нива: фина (Fine-grain) гранулярност (R/C е с малка стойност поради завишените комуникации) и груба (Coarse-grain) гранулярност (високи стойности за R/C).



♦ **Междупроцесна комуникация** - определя се от **топологията (T)** на свързващата мрежа (Interconnection Network - IN) между основните компоненти на КА - процесори, памети, периферия. Дефинират се гранични нива: силно-свързана топология (Heavily interconnected topology), която се отнася главно за наличие на общи ресурси и слабо-свързана топология (Lightly interconnected topology) при разпределени и самостоятелни архитектурни компоненти.

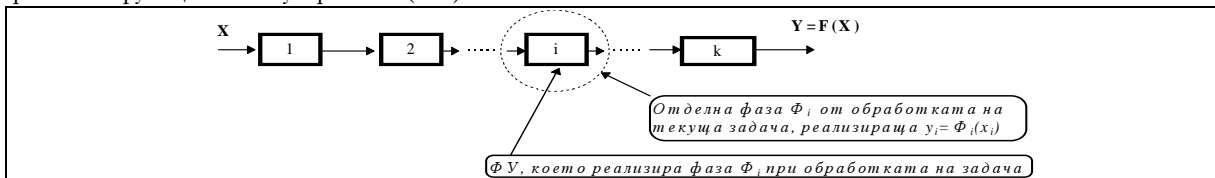
♦ **Междупроцесна синхронизация**, която се свързва с принципа на **управление (C)** в КА. Възможностите са от централизирано (глобално) управление (“Tight”) до разпределено (локално) управление (“Lease”).

На базата на тези три характеристики се формира **дефиниционна област** за селектиране на базовите паралелни КА - определено сечение (подобласт) в тримерния куб:

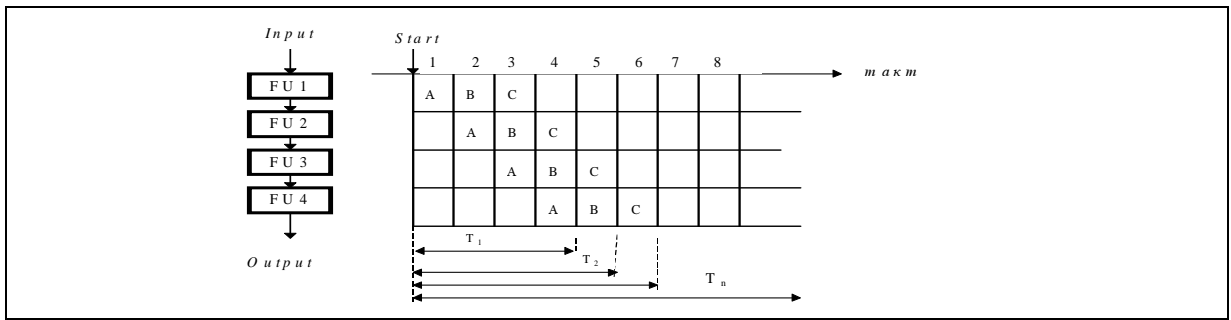


1.2. Принцип на конвейерната обработка (Pipeline Processing)

Конвейерната обработка е типична за системи, поддържащи паралелизъм на ниско ниво и се състои в декомпозиране на задачите на отделни информационно-независими фази, изпълнявани последователно в различни функционални устройства (ФУ).



Конвейерната структура изчислява глобалната функция $F(X)$ чрез функционална декомпозиция: $F(X) = f_k[f_{k-1}[\dots f_2(f_1(X))]\dots] = f_1 * f_2 * \dots * f_{k-1} * f_k$. Всяка подфункция се реализира в отделна фаза, като обработката на отделна задача е последователна във времето. Паралелизмът се осигурява при множество от задачи A, B, C, ..., преминаващи през конвейера.



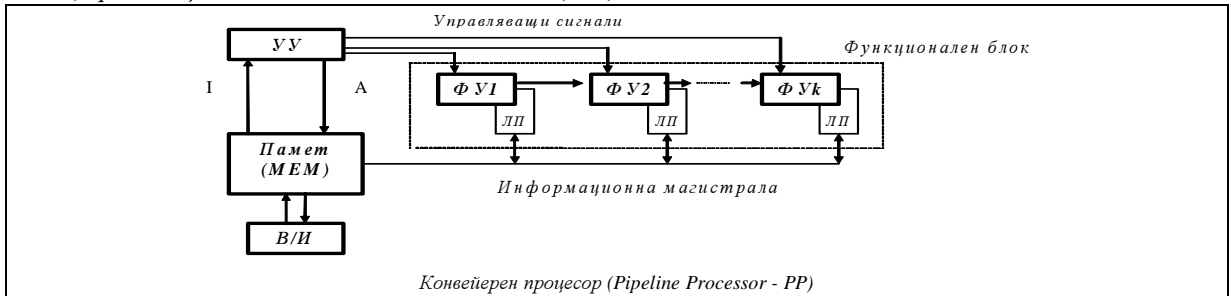
Конвейерната обработка е синхронизирана, обикновено с фиксиран такт на конвейера $\tau = const$ за изчислението във фаза "i" ($i=1, \dots, k$). Общото време за изчисляване на една задача е $T_1 = k \cdot \tau$. Всяка следваща задача ще бъде готова на изхода на конвейера след време τ , т.е. $T_2 = T_1 + \tau$ и $T_{i+1} = T_i + \tau$. В този смисъл, общото време за обработка на n задачи ще бъде: $T_n = T_{n-1} + \tau = k \cdot \tau + (n-1)\tau = (k+n-1)\tau$.

При конвейерното изпълнение на определени задачи се налага използване на обратна връзка. Обикновено това е при рекурсивно изчисляване на функция $F(X)$, така че резултатът y_i във фаза Φ_i да зависи от няколко предходни фази. При такава организация на обработката се прилага метода на съпровождащата функция: $\exists g$, такава че $f[a, f(b, x)] = f[g(a, b), x]$.

1.3. Базова структура на конвейерен процесор

♦ **Базовата структура** включва: функционален блок, изграден от ФУ и локални памети за буферизиране на междинни данни; обща памет (MEM); управляващо устройство (УУ). Управляващите сигнали се формират от УУ на базата на инструкциите I, извличани съгласно адресите A от паметта за програми. При организацията на конвейерния процесор се решават следните проблеми:

- а) **конвейеризация на обработката** - декомпозиране на задачите, формиране на фазите и евентуалните обратни връзки, дефиниране на обобщен алгоритъм на конвейера;
- б) **организация на паметта** - обособяване на локалните памети (ЛП) и общата памет (MEM), разпределяне на данните между фазите, формиране и съхраняване на резултатите;
- в) **организация на управлението** - диспечериране на задачите и синхронизация на фазите;
- г) **организация на входно-изходната система (В/И)**.



♦ **Дефиниционна област.** Съгласно архитектурните форми на Flynn, PP се класифицира като MISD. Поддържа паралелизъм от ниско ниво, реализиран чрез функционалната декомпозиция и обработката във ФУ. Дефиниционната област се определя на базата на:

- а) Гранулярност - предимно фина (от много-фина до средна) и зависи от конкретното приложение;
- б) Топология - средно свързана, поради променливата комуникация на данните в ограничен смисъл;
- в) Управление - глобално, осъществявано от централизирано устройство за обща синхронизация.

♦ **Ефективност на конвейерната обработка.** Оценява се чрез коефициент на ускоряване, където T_s е времето за последователно (SISD) изпълнение на n задачи, а T_k е времето за конвейерно (MISD) изпълнение на n задачи в k фазен конвейер:

$$\sigma_k = \frac{T_s}{T_k} = \frac{n \cdot (k \cdot \tau)}{(k+n-1) \cdot \tau} = \frac{n \cdot k}{k+n-1};$$

Асимптотичната оценка на σ_k се определя при $n \rightarrow \infty$:

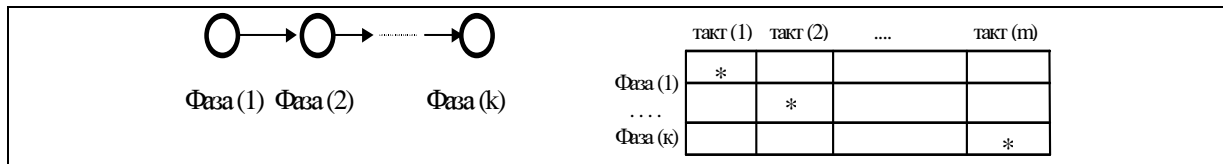
$$\lim_{n \rightarrow \infty} \left(\frac{T_s}{T_k} \right) = k;$$

Тогава ефективността може да се оцени чрез:

$$\eta = \frac{\sigma_k}{k} = \frac{1}{k} \cdot \frac{n \cdot k}{k+n-1} = \frac{n}{k+n-1}; \text{ и ако } n \rightarrow \infty \Rightarrow \eta \rightarrow 1.$$

1.4. Диспечериране, синхронизация, характеристики

♦ Диспечериране - определяне на реда на отделните атоми за изпълнение в конвейера и моментите за подаване на входните данни към ФУ. Осъществява се на базата на таблица на заетост за отделните ФУ през тактовете на конвейера. Ред в таблицата показва използването на дадено ФУ последователно във времето, а колона - кои ФУ са заети през даден такт. Изпълнението на дадена конвейерна задача (или операция) става чрез инициализация на съответстващата ѝ таблица на заетост. Целта на диспечерирането е безконфликтни инициализации на една таблица многократно или на няколко различни таблици едновременно.



♦ Синхронизация - регламентиране на достъпа до данните в паметта през отделните фази и заемането на ФУ от последователни инициализации на таблици на заетост за конвейера. Постигане на глобална синхронизация при наличие на доминираща фази (например с квант 2τ) е възможно чрез разделяне на доминиращата фаза на две подфази с квант τ или чрез нейното разпаралелване (дублиране).

♦ Характеристики - основни характеристики за конвейера се следните:

- а) време за изчисление (T) - общ брой единици време, необходими за еднократна реализация на таблицата на заетост (определя се от броя на колоните); общото време за N последователни инициализации е T_N ;
- б) латентност (λ) - време (брой тактове) между две последователни инициализации ($\lambda \geq 0$); средната латентност $\lambda_{av} = T_N / N$ е усредненият брой тактове на синхронизация между две инициализации;
- в) темп на инициализация (Θ) - среден брой инициализации на последователни задачи за време T_N , т.е. $\Theta = 1/\lambda$;
- г) използваемост на i -то ФУ - относителна част от времето T_N , през която е използвано ФУ от последователни инициализации - $\gamma_i = (t_i / T_N) \leq 1$.

1.5. Видове конвейерни архитектури и възможности за симулация

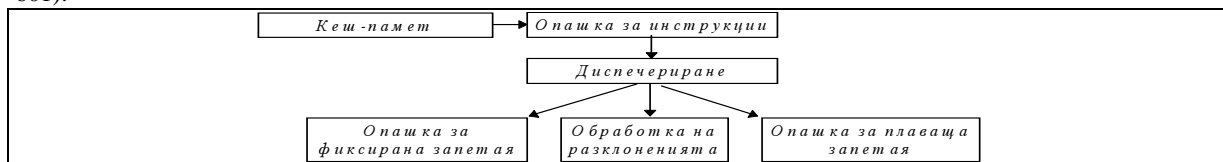
♦ В зависимост от обекта на конвейеризация - дефинират се следните два типа:

- а) аритметичен конвейер - организация на k -фазна магистрала за аритметична обработка на данни; обикновено блокът за скаларни изчисления (скаларен процесор) се разделя на блок за целочислени изчисления (Fixed-point execution unit) и на блок за изчисления с плаваща запетая (Floating-point execution unit) - напр. при IBM 360/91:



Примерно конвейерно изпълнение на операцията “сумиране на числа с плаваща запетая” включва фазите: (1) нормализация на операндите; (2) сравняване на порядъците; (3) коригиране на мантисата; (4) събиране на мантисите; (5) нормализация на резултата.

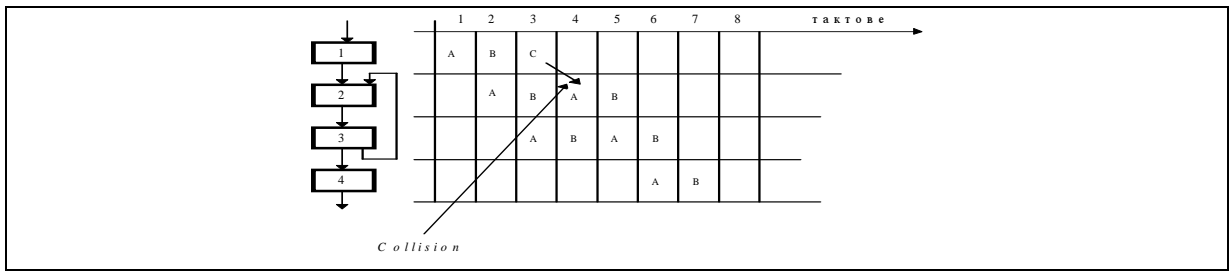
- б) инструкционен конвейер - конвейеризация на изпълнението на инструкциите, характерно най-вече за RISC-архитектурите (напр., инструкционен конвейер при Pentium и при суперскаларния процесор Power PC 601):



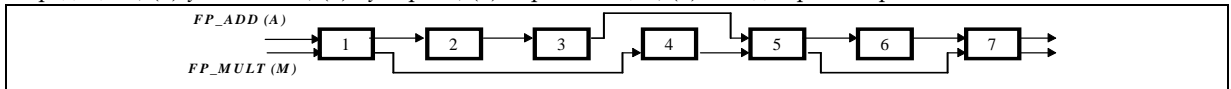
Примерна конвейеризация при изпълнение на инструкцията се основава на фазите извличане, декодиране и изпълнение. Един инструкционен конвейер действа правилно при условие, че никоя инструкция не е принудена да изчака резултата от друга инструкция в процеса на обработката.

♦ В зависимост от функционалните възможности:

- а) Еднофункционален (фиксиран) конвейер (ЕФК) - специализиран за изпълнение на една операция над потока от данни. Допуска се циклично използване на някои фази (обратна връзка), при което се получава наслагване на последователни инициализации:



а) *Многофункционален конвейер (МФК)* - позволява инициализация на две или повече таблици на заетост. Възможни варианти: *статичен* - сравнително рядко се сменя изпълняваната функция; *динамичен* - честа смяна на изпълняваната функция, което налага прецизна синхронизация между фазите. Примерен динамичен МФК е показан по-долу, изпълняващ операциите сумиране (FP_ADD) и умножение (FP_MULT) на числа с плаваща запетая. Фазите са следните: (1) входен регистър; (2) изчисляване на порядъците; (3) изравняване на порядъците; (4) умножение; (5) сумиране; (6) нормализация; (7) изходен регистър.



При *симулация* на конвейерна архитектура всяка фаза може да се представи като едноканална СМО с фиксиран квант за обслужване на постъпила заявка, определен от такта на конвейера и буфер (опашка) за изчакване. Поради строгата синхронизация между фазите е необходимо да се вземат мерки за съхраняване реда на последователните фази от дадена задача. Това е важно при ЕФК с обратна връзка и при МФК за различни операции. За избягване на конфликти се допуска дублиране на критична фаза (или област). При МФК е удачно различните операции да се идентифицират по тип (напр., чрез стойност на параметър за всеки транзакт) и да се обслужват съгласно съответната таблица на заетост.

2. Задание за лабораторна работа

1. Да се разучат организацията, особеностите при диспечериране и системните характеристики за различни конвейерни структури.
2. Да се проведе симулационно изследване на *линеен еднофункционален конвейер* с постоянен такт, еднакъв за всички фази и да се анализират получените оценки за основните системни характеристики и ефективността.
3. Да се разработи базова структура и симулационен модел за изследване на системните характеристики и ефективността на *еднофункционален конвейер* с обратна връзка (по допълнително задание). Да се обобщат и анализират получените експериментални резултати.
4. Да се разработи базова структура и симулационен модел за изследване на системните характеристики и ефективността на *многофункционален конвейер* (по допълнително задание). Да се обобщат и анализират получените експериментални резултати.

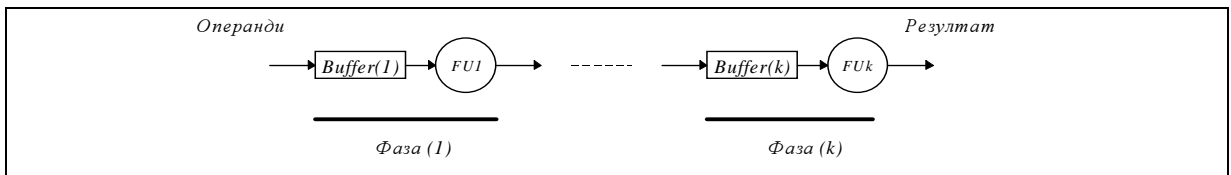
3. Лабораторни експерименти

По точка (1)

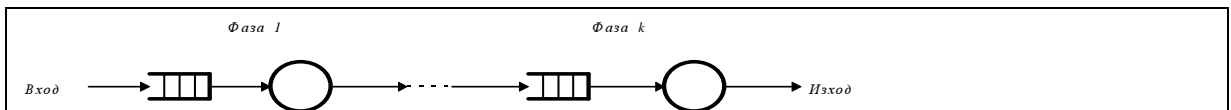
Запознаване с текстовия материал от теоретичната постановка.

По точка (2)

Изследването се провежда на базата на следната *структура* на линеен ЕФК с постоянен такт, еднакъв за всички фази:



Абстрактното описание чрез СМО е линеен многофазен модел:



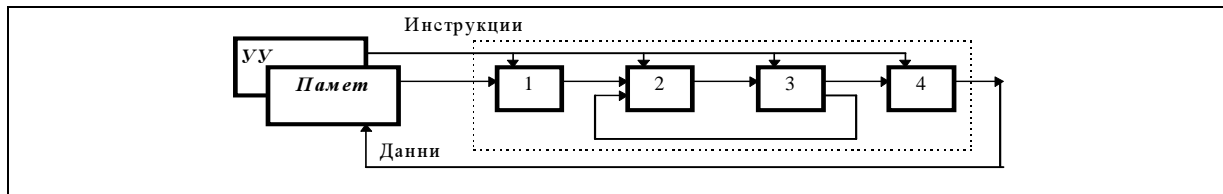
По-долу е показан примерен *симуляционен модел* за линеен конвейер с $k=2$ фази, такт на конвейера 5 ЕМВ и експоненциален входен поток от задачи. Изпълнява се за 30 задачи.

EXPO	FUNCTION	RN1,C24
		0,0/1,,104/2,,222/3,,355/4,,509/5,,69/6,,915/ .7,1.2/75,1.38/8,1.6/84,1.83/88,2.12/9,2.3/ .92,2.52/94,2.81/95,2.99/96,3.2/97,3.5/ .98,3.9/99,4.6/995,5.3/998,6.2/999,7/9998,8
	GENERATE	10,FNSEXPO
	QUEUE	1
	SEIZE	1
	DEPART	1
	ADVANCE	5
	RELEASE	1
	QUEUE	2
	SEIZE	2
	DEPART	2
	ADVANCE	5
	RELEASE	2
	TERMINATE	1
	START	30
	END	

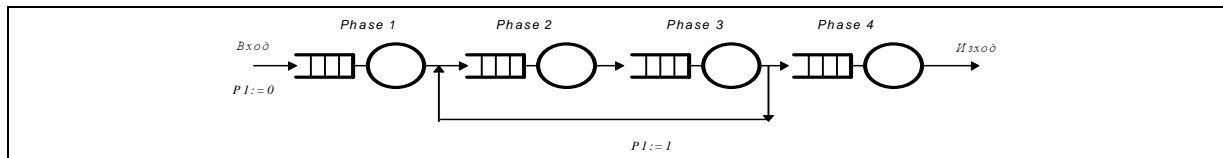
По точки (3) и (4)

Изпълнението на задачите е аналогично на предходната задача и включва разработване на примерна базова структура, нейното описание чрез СМО, създаване на симуляционния модел и провеждане на експерименти. Примерни елементи по работата са показани по-долу.

◆ *Еднофункционален конвейер с обратна връзка* (примерна базова структура):



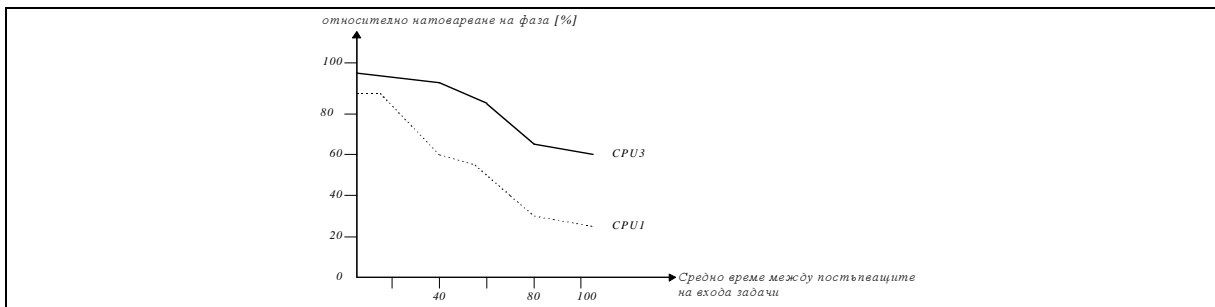
Представяне на ЕФК чрез СМО:



Програмен симуляционен модел:

	GENERATE	30,10	-- P1=0 (стандартна стойност)
	QUEUE	QCPU1	
	SEIZE	CPU1	
	DEPART	QCPU1	
	ADVANCE	20	
	RELEASE	CPU1	-- задача напуска Фаза (1)
BACK	QUEUE	QCPU2	-- и постъпва във входния буфер на Фаза (2)
	SEIZE	CPU2	
	DEPART	QCPU2	
	ADVANCE	20	
	RELEASE	CPU2	
	QUEUE	QCPU3	
	SEIZE	CPU3	
	DEPART	QCPU3	
	ADVANCE	20	
	RELEASE	CPU3	
	TEST E	P1,0,FOUR	-- проверка "P1=0" и преход към FOUR при "FALSE"
	ASSIGN	1,1	-- ако ["P1=0" е TRUE], тогава се присвоява P1=1
FOUR	TRANSFER	,BACK	-- и задачата се връща към Фаза (2)
	QUEUE	QCPU4	
	SEIZE	CPU4	
	DEPART	QCPU4	
	ADVANCE	20	
	RELEASE	CPU4	
	TERMINATE		
	GENERATE	1000	
	TERMINATE	1	

Симулацията се изпълнява за 1000 ЕМВ, след което се снемат статистическите оценки за основните характеристики. Примерна графична интерпретация на оценките за натоварването на отделни функционални устройства (фази) е представена на следващата



♦ Многофункционален конвейер за N операции

Базовата конфигурация на примерен МФК за $N=2$ и разпределението на фазите по операциите е показана в т.1.5 от теоретичната постановка. За диспечерирането на различни по тип задачи (операции), постъпващи последователно на входа на конвейера, е важно кой тип е изпреварващ. Това се определя от факта, че операциите от тип FP_ADD (A) използват повече функционални устройства от конвейера спрямо операциите от тип FP_MULT (M).

В този смисъл, при последователност "M предхожда A" не се получава конфликт при заемане на ФУ. При обратната последователност обаче, е възможен конфликт при опит за заемане на ФУ(5) от различни инициализации. Аналогичен проблем възниква и за ФУ(7).

За разрешаване на колизията може да се използва дублиране на фаза или индексиране (чрез параметър на транзакта) на последователните задачи още при генерирането и въвеждане на приоритет по постъпване при опит за множествен достъп до ФУ.

Управлението на симулацията може да се направи както по часовник (таймер), така и по зададен брой задачи.

4. Съдържание на отчета

В отчета, оформен стандартно (име, факултетен номер, група, курс, дисциплина), се включва:

- номер и тема на упражнението;
- задание за лабораторна работа;
- резултати от лабораторните експерименти.