Хиерархиски PLC системи

Милошевски Александар

Ментор: проф. Миле Станковски

Хиерархиски PLC системи

Падот на цената на компјутерските компоненти, големиот степен на интеграција и брзини на процесирање и проток на податоците, водени од технолошкиот напредок целосно ги преплави старите оператор-ПЛЦ системи и доведе до забрзан развој на употреба на сложени хирерахиско управувани ПЛЦ системи во една дистурибутивна средина овозможена со примена на компјутерски мрежи. Овие системи овозможуваат помали почетни трошоци за изведба, поголема сигурност, проширливост, брз одзив и итн. Се состојат од голем број процесорски, операторски и аквизициски единици разместени по целиот објект меѓусобно поврзани. Посебните функции и компоненти од различни произведувачи, кои ги задоволуваат комуникационите протоколи и комуникацискити мрежи хиерархиски, се организирани во повеќе нивоа.

Оператор-ПЛЦ систем е прикажан на сликата.

Поврзувањето на компонентите на хиерархиските системи на може да биде:

- точка до точка
- мрежно
- комбинација

Поврзување (точка до точка)

Соодржи повеќе процесорски единици сместени во близина на надзираните уреди и операторски станици. Поцесорските единици меѓусебе комуницираат по пат на меѓусебни независни комуникациски канали. Трошоците за ожичување се релативно исплатливи со оглед на тоа што процесорските единици се поврзани со сензорите и актуаторите со кратки водови. Електромагнетните зрачења се битно намалени со што е зголемена отпорноста на шумови и локалните процесорски единици можат да работат без обзир на случувањата во други делови на системот (дефект, одржување...) со што е зголемена зголемена сигурноста.

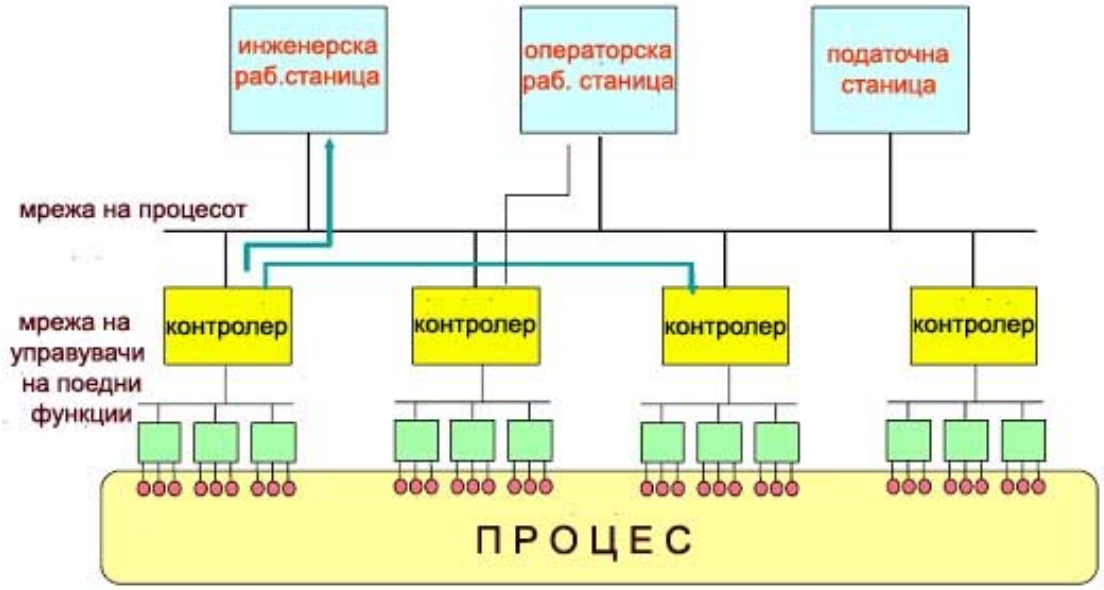
Мрежно поврзување

Процесорските единици не се поврзани со поединечни комуникациски врски туку во заедничка комуникациска мрежа. На овај начин се постигнува заштеда во ожичување и битно се поедноставува надоградбата на системот, додека недостатокот поради сложеноста на комуникацијата се губи благодарејќи на готовите хардверски и софтверски комуникациски модули. Во современите системи за управување и надзор се користат мрежно поврзување, додека поврзување точка до точка се употребува за поврзување на специјални, нестандардни уреди.

Начинот на управување на хиерархиските систем можи да биди централизиран или децентрлизиран.

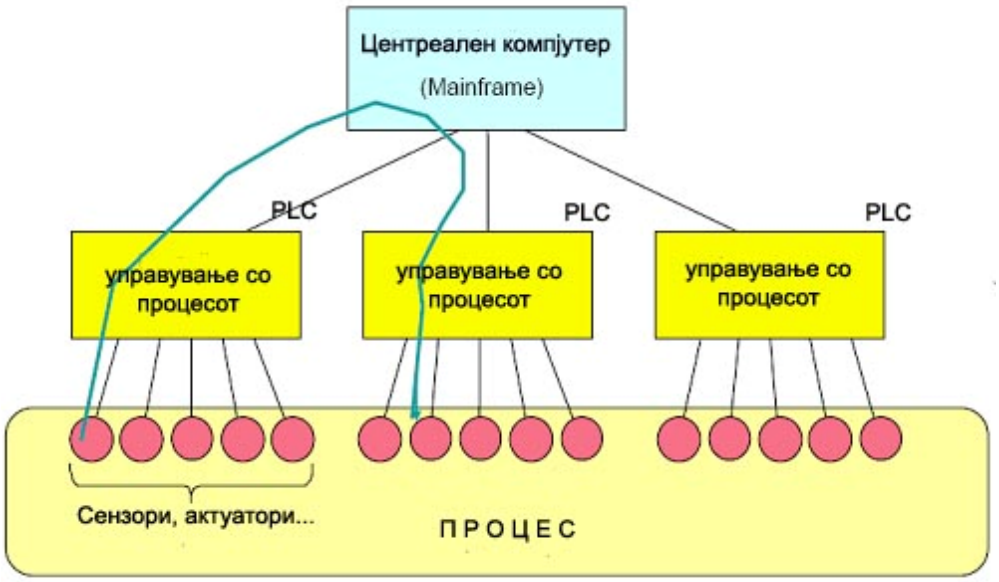
Децентрализирана архитектура

Соодржи само мрежи од моќни компјутери со одредени права и привилегии за надгледување и управување на поединечните системи. Ваквите системи ги користат предностите размена на податоци, ресурси и компоненти, како и комуникација и контрола и на компонентите дури и на исто ниво во хиерархијата.

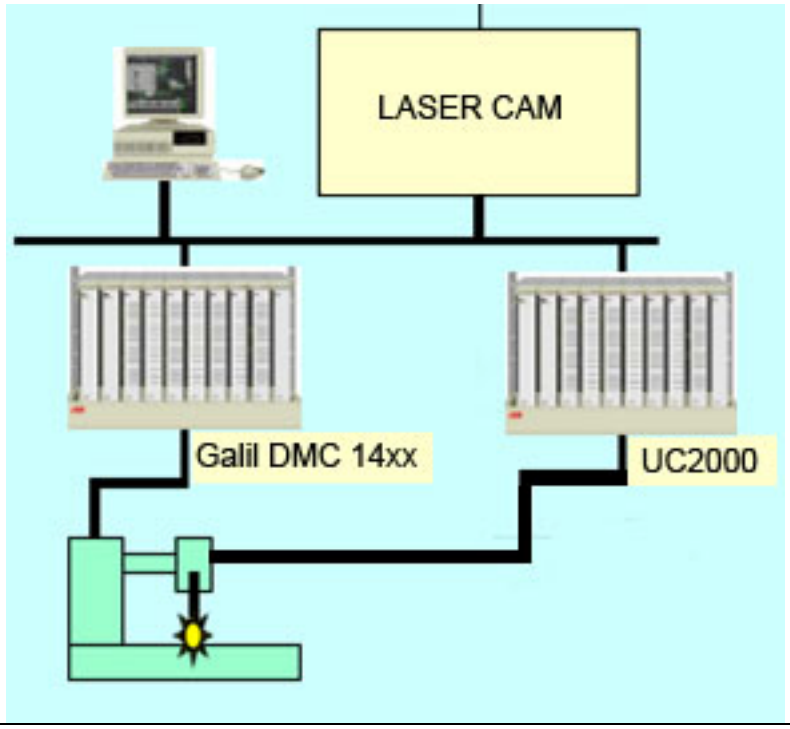


Централизирана архитектура

Соодржи само една централен компјутер кој ги надгледува и ги проследува наредбите до секој од ПЛЦ –та. Предностите на ова архитектура се ниска цена и едноставност. Со падот на цените и се поедноставната инсталација на сложени управувачки архитектури ова архитектура се смета за застарена. Нејзината примена ограничена е главно на мали подсистеми и производни процеси со строго локално управување. Сите примери при имплементацијата и објаснувањето на техниките и технилогиите на хиерархистките ПЛЦ системи ке бидат поврзани со практичните искуства од еден ваков мал централизиран систем со напомена дека еден сложен хиерариски систем можи да се разгледува како збир од поедноставни подсистеми.

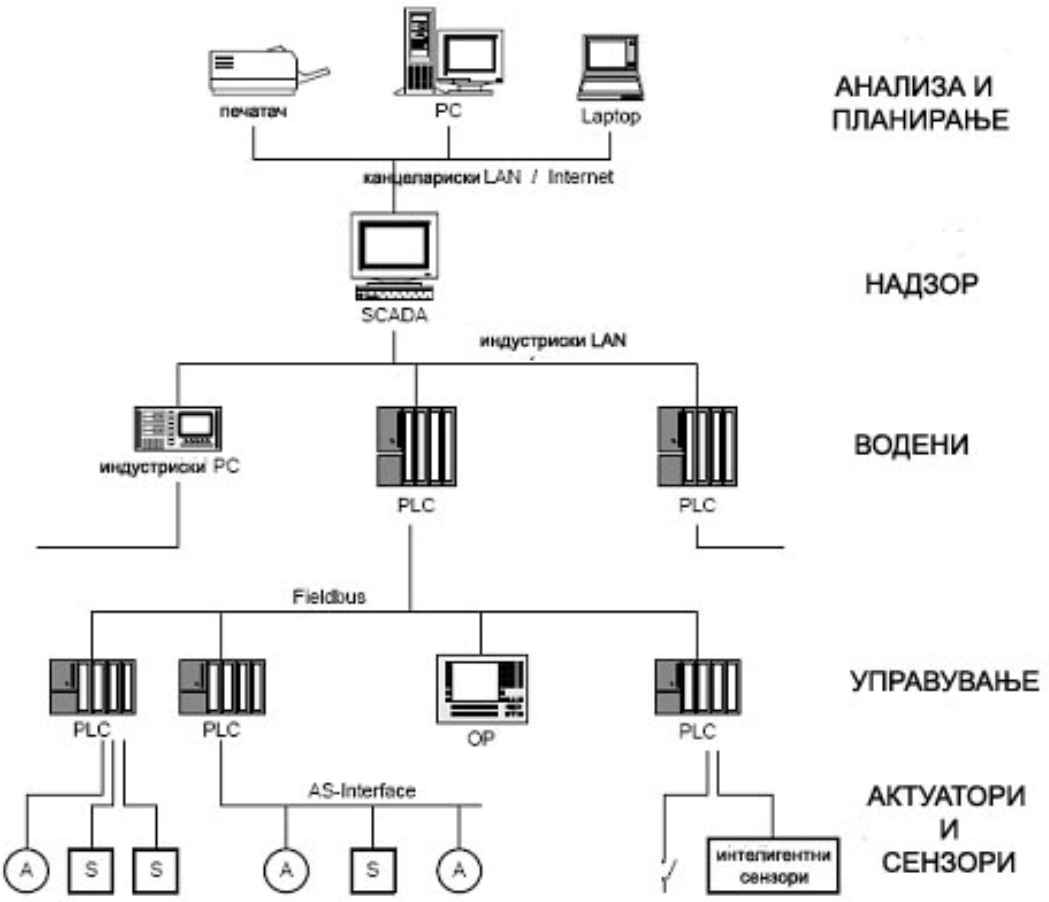


Егзампларниот реален систем кој ќе се користи за опис на техниките и технологиите соодржани во денешните хиерархиски системи е типичен пример на централизирана структура и е преставен на слика

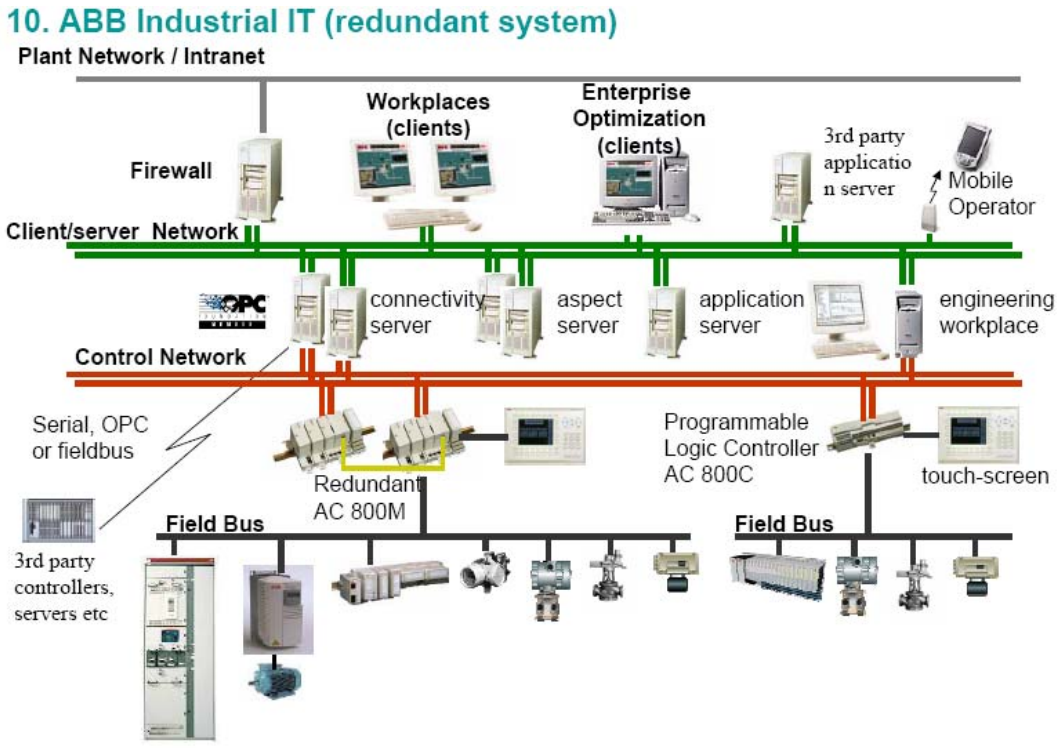


Глобално во производствените системи можат да се воочат 5 нивоа на контрола и управување:

1. Ниво на планирање и анализа
Информацискиот систем на ова ниво се состои од класична канцелариска компјутерска опрема поврзана поврзана со канцелариски LAN и врска со интернет.
2. Ниво на надзор и аквизиција
Опфаќа интегриран систем за аквизиција и надзор кој ги покрива сите важни системи.
3. Ниво на управување на процеси
Опфаќа ПЛЦ-а и индустриски ПЦ-а кои управуваат на цел подсистем или функционална група. Комуникациски врски се изведени со помош на мрежи со голем пропусен опсег...
4. Ниво на управување со поединечните функции, уреди и регулациони уреди
Опфаќа ПЛЦ-а, микроконтролери, индустриски ПЦ-ја. Комуникациски врски се изведени со помош на fieldbus,..., profibus
5. Ниво на управувани уреди
Ги опфаќа сензорите и актуатори поврзани на управувачкиот уред



Пример за еден ваков систем е.



Хиерархиските ПЛЦ системи ги покрива долните 3 нивоа па излагањето ќе биди насочено во описот на составните компонентите и меѓусебната комуникацијатана тие нивоа.

1. **Компјутерско водење**

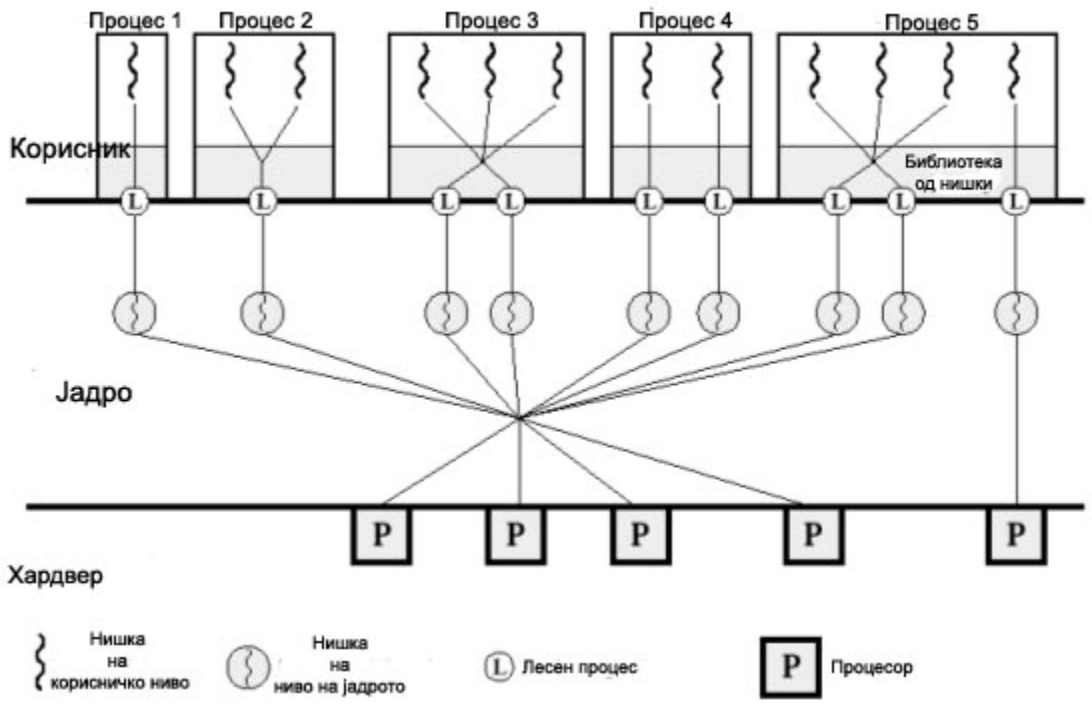
2. **ПЛЦ управување**
   2.1 Контрола на процес
   2.2 PID
   2.3 Програмирање

3. Актуатори и Сензори
   3.1 PID PLC контрола на актуатори
   3.2 Подесување (*Tuning*)
   3.3 Контрола на движење (Motion Control)

4.Комуникација
   4.1 Controlnet
   4.2 DeviceNet (CAN Bus)
   4.3 Sercos
   4.4а Синхроно мултиплексирање со поделба на кодот Synchronous CDMA
   4.4б Асинхроно мултиплексирање со поделба на кодот Asynchronous CDMA

1.Компјутерско водено управување

Иако се сметало дека компјутерите ке ги истиснат PLC-и во пракса се покажало тека тие одлично се надополнуваат....Денес компјутерските системи со кои се водат подсистемите или функционалните групи можат да бидат составени од обичен PC компјутер до специјални индустриски мултипроцесорски компјутери. Оперативни системи под кој работат компјутерските ситеми се *MSDOS*, *UNIX*, *SOLARIS*, *WINDOWS* серија, *VxWorks*, *QNX*... Шематски приказ на оперативен систем при управувањето со процеси на компјутерски систем со повеќе процесори е даден на сл.



Компјутерски водење во егзампларниот систем го извршува индустриски PC компјутер под WINDOWS XP Embedded

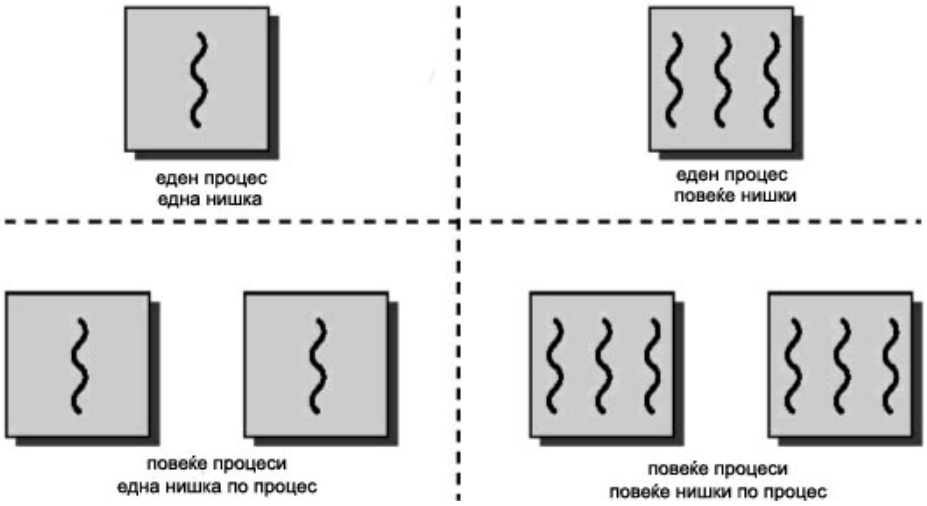
Еден компјутерски системи треба да ги има следните к-ки за да овозможи квалитетно управување.

Мултизадачност (Multithreading) и мултипроцесирање

Способност да изведува повеќе нишки (задачи) истовремено во оквир на адресниот простор на еден процес и повеќе процеси на ниво на систем без меѓусебни судирања.

Оперативниот систем на компјутерот гледано од системска страна можи да дозволува извршување на (види слика):

- еден процес со една нишка (*MSDOS*)
- еден процес со повеќе нишки
- повеќе процеси со по една нишка (*UNIX*)
- повеќе процеси, секој со повеќе нишки (*SOLARIS, WINDOWS*)



,додека гледано од хардверска страна треба да се забележи дека различни оперативни систем подржуваат и различен број на процесори.

За имплементација на мултизадачноста се користат следните алгоритми за доделување одреден задачи на процесорот:

- *First In First Out (FIFO): Прв барал прв услужен.*
- *Shortest Job First (SJF): Најкратката задача прва.*
- *Time-slicing: Времето во периодот е поделено на временски парчиња и секој задача добива по едно временско парче. Овој алгоритам е познат и како (Round-robin).*
- *Earliest Deadline First (EDF): Приоритет добива онаа задача која треба да се изврши што поскоро*
- *Fixed priority scheduling (FPS): Секоја задача има фиксен приоритет врз база на кој се донесува одлуката. Овој систем е доминантен во* RTOS
- *Off-line scheduling: Во зададен временски момент задачата е доделена на процесорот. Природен за периодични процеси*.
- *"Hydrid" Хибриден пристап како комбинација на претходните*

На сликата е прикажана функционална шема на мултизадачност на оперативен систем.

Прекини (Interrupt mode)

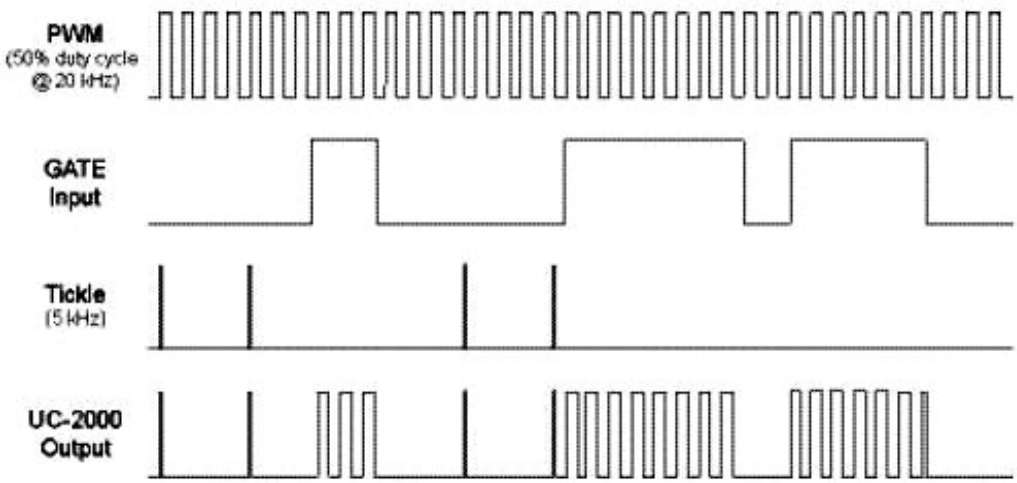
Способност на системот да го суспензира тековното извршување на програмот или нишката и да изведи друг програм или нишка како одговор на сигнал на процесот кој означува случување со поголем приоритет.

За време на извршувањето на гравирачкиот процес, системот добива интерапт за почеток и крај на линија за гравирање при што го овозможува/оневозможува пукањето на ласерот.

Временски иницирани процеси
Способност на системот да изведи акција (стартување, паузирање или гасење) на одреден процес во зададен временски момент.

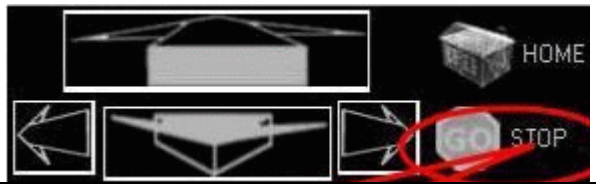
Управувачки сигнални
Способност на системот да пресметува и да испраќа управувачки сигнали на процесот

Софтверот што работи на компјутерот го чита моменталниот пиксел и во зависност од вредноста на скалата на сивата боја испраќа GATE управувачки сигнал со различна широчина на тој начин контролирајќи ја снагата на ласерот.



Операторско-иницирани случувања
Способност да ги прифаќа командите од операторот (интерфејс човек-машина) HMI и да одговара на нив

Преку софтверскиот интерфејс операторот можи да изврши команди на манулено или автоматско движење на ласерот по *x* и *y* оска, стартување на гравирачкиот процес или *home* секвенца, или нивно стопирање.



Аквизиција (Polling)
Способност периодички да прави аквизивија на податоци кои ја означуваат моменталната состојба на процесот.

Меѓусебно исклучување (Mutual exclusion)
Системот треба да располага со механизми за кординација на акцијата на два или повеќе уреди или задачи при пристапување на критични региони (заеднички податоци или ресурси)

	Нишка А	**Нишка В**
Изворен код:	*var = Y +1*	*X = f(var)*

Механизми за меѓусебно искучување:

- Временско секвенцирање

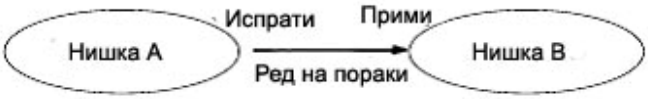
Времето е поделено на мајорен циклус дефиниран од Најмалиот заеднички производ и минорен циклус дефиниран од времтраењето и периодите на трење на нишките со најголем приоритет. Пример на временско секвенцирање е дадено на сликата.

Нишка {траење=0.2 временски дел, период=5 временски делови}

Нишка$_i$ {*0.2, 2*}
Нишка$_{i+1}$ {*2, 5*}
Нишка$_i$ има повисок приоритет од Нишка$_{i+1}$
Најмал заеднички производ (LCM) за Нишка$_i$ и Нишка$_{i+1}$ е 5.

- Редови со пораки

Нишките меѓусебе комуницираат преку пораки во кружни редови каде оперативниот систем обезбедува меѓусебно исклучување и други сервиси.



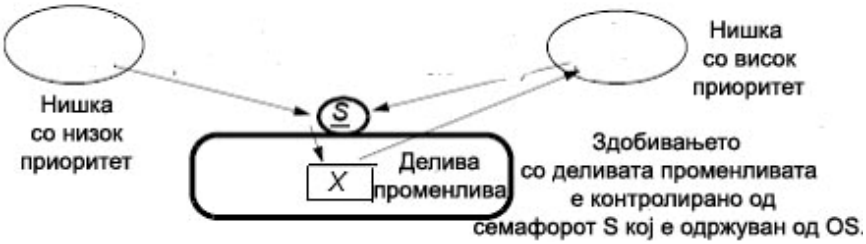
Изворен код:

Нишка А
```
LocalVarA= LocalVarA+1;
Send(LocalVarA, X) ...
```

Нишка В
```
 Read(LocalVarB, X)
use of LocalVarB ...
```

- Семафори

Семафорите се измислени во 1965 од старана на Дијкстра и се редовни во оперативните системи. Можат да се гледаат како знаменца(flag) кои можат да се користат како катанци за заштита на критичните региони.



Изворен код:

Нишка со низок приоритен
```
Aqure(S);
X= ...
Release(S);
```

Нишка со висок приоритет
```
Aqure(S);
Y=f(X)
Release(S);
```

Справување со исклучителни ситуации (Exception Handling).
Способност системот да се справи со случки (лош квалитет на производот, неостаматок на суровина или ел.енергија, опасни услови, несиправност на ситемот ...) надвор од нормалниот и посакуван режим на работа на работа на процесот или компјутерскиот систем.

Системот при престанок на електрична енергија користи UPS за да ги забележи параметрите во дадениот момент и да го угаси ситемот и на тој начин обезбедува сигурност и продолжување на работата на она место кај што е прекината. Исто така обработува исклучоци како движење надвор од работната површина, прекин во комуникација ...

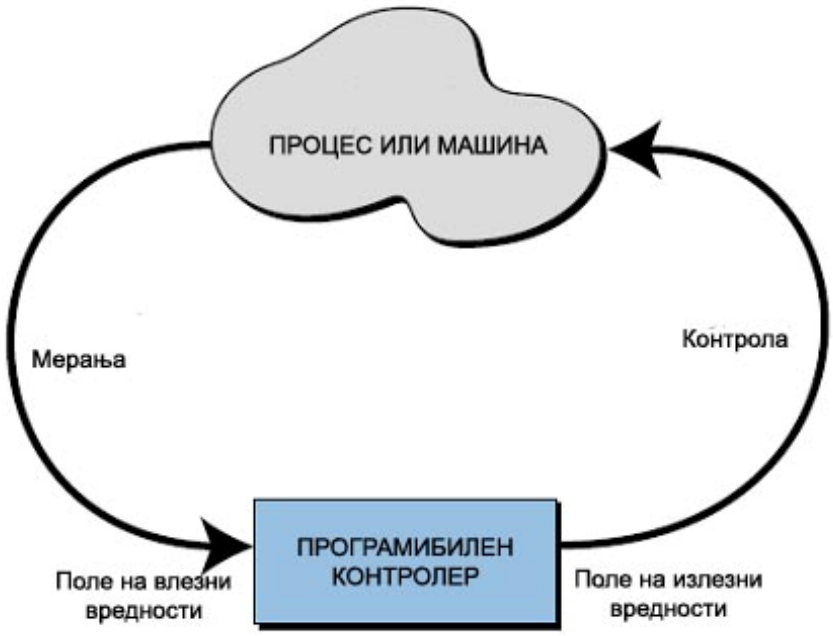
Комуникација
Способност да комуницира со други компјутери со помош на дирекни или мрежни протоколи, со различни типови на програмабилни логички контролери (PLC) и интелегентни уреди од различни произведувачи ради формирање на хиерархиски систем.

Системот комуницира:
- **со ласерскиот контролер преку сериска RS232 (сетирање и отчитување) и паралелна LPT (за гејт управувачки сигнал) комуникација**
- **со контролерот за движење преку сериска USB (сетирање, отчитување и контрола) и парелна LPT (за процесни интерапти) комуникација**
- **со контролерот на индустриската копчарница преку сериска комуникација со прилагоден протокол**

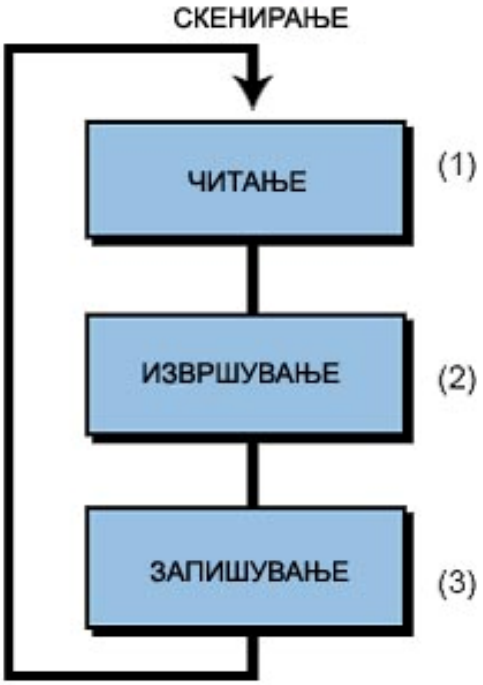
Дијагностика
Способност на системот да овозможи лесна и брза дијагностика и лоцирање на дефектот.

2. ПЛЦ управување

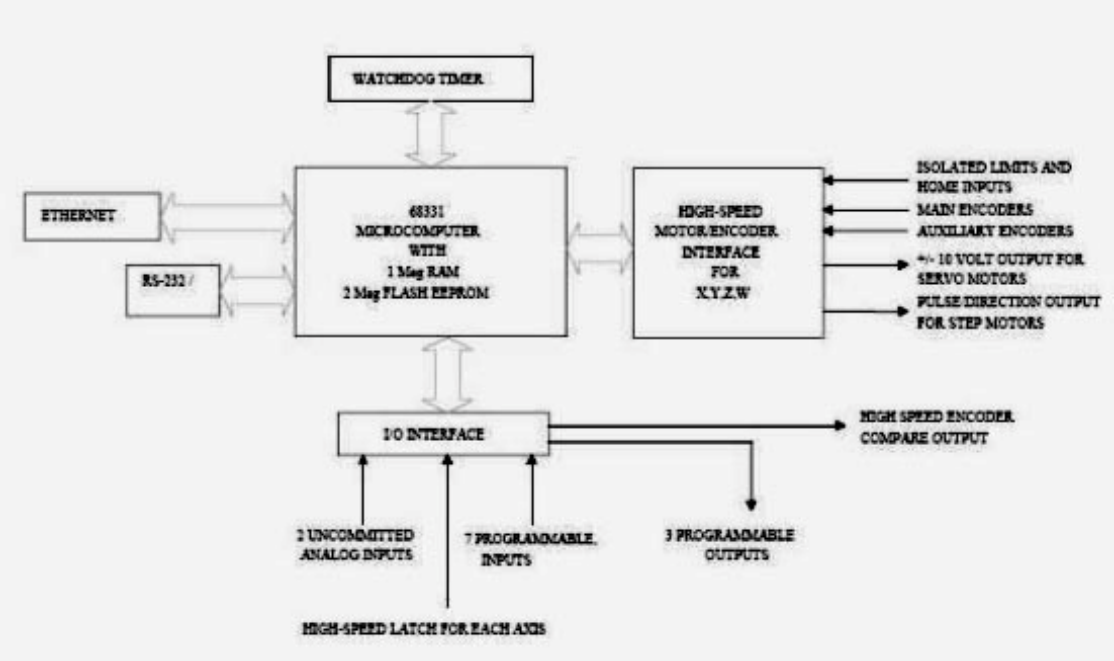
Програмибилни логички управувачи, истото така наречени програмибилни управувачи се членови на компјутерската фамилија, користат интегрирани кола наместо електромеханички уреди за имплементација на управувачка функција. Тие се способни да чуваат инструкции, како секвенцирање, тајминг, броење, аритметички, манипулација на податоци, и комуникација, за да ги контролираат индустриските машини и процеси. На сл. е илустриран концептуален дијаграм на ПЛЦ апликација.



Операциите на програмибилниот управувач се релативно едноставни. Влезно/Излезниот ситстем е физички поврзан со уредите што се среќаваат во машините или се користат во контрола на процесите. Овие уреди можат да бидат дискретни или аналогни влезно/излезни уреди, како лимит прекинувачи, трасдуктери за притисок, копчиња, стартери на мотори, соленоиди, итн. И/В интерфејс обезбедува врска меѓу *ЦПЕ* (Централно процесорска единица) и провајдерите на информации (влезовите) и контролираните уреди (излези). За време на опрерацијата *ЦПЕ* извршува три процеси: (1) **чита**, или прифаќа влезни податоци од надворешните уреди преку влезниот интерфејс, (2) **извршува**, или изведува контролен програм зачуван во меморијата, и (3) **запишува**, или ги обновува излезните уреди преку излезниот интерфејс. Овој процес периодично читање на влезовите, извршување на програмот во меморијата и обновување на излезите е наречен **скенирање**. Сликата ги илустрира графичка репрезентација на скенирањето.



Во нашиот систем е употребен е DMC14xx PLC управувач чија структура е прикажана на слика



ЦПЕ
ЦПЕ и специјализиран 32-битена Motorola 68331 со 1MB RAM и 2 MB EEPROM
RAM меморијата кој обезбедува меморија за променливи, поле на елементи и апликациски програми. EEPROM обезбедува постојана меморија за променливи, програми и полиња како и DMC-14XX основниот фабрички програм(firmware).

Интерфејс за мотор
Галил субмикронско влезно поле за квадратно декодирање за секој енкодер 12 MHz. За стандардно серво операција, контролерот генерира +/-10 Volt аналоген сигнал (16 Bit Дигитално аналоген конвертор). За синусуидална комутациска опреација контролерот користи 2 +/-10Volt
аналогни сигнали. За чекорни мотори, контролерот генерира чекорен и сигнал за насока

Комуникација
Содржи 2 комуникациски модули: сериски RS-232 (19.2 kbaud) за точка до точка поврзување и 10base-ETHERNET за мрежно поврзување.

Влезно/Излезни уреди
7 TTL влезови and 3 TTL излези + 12-битен аналогени влези. Влезовите можат да се користат за тригер позицирачките кола за задршка со голема брзина во секоја оска.

Процесна контрола

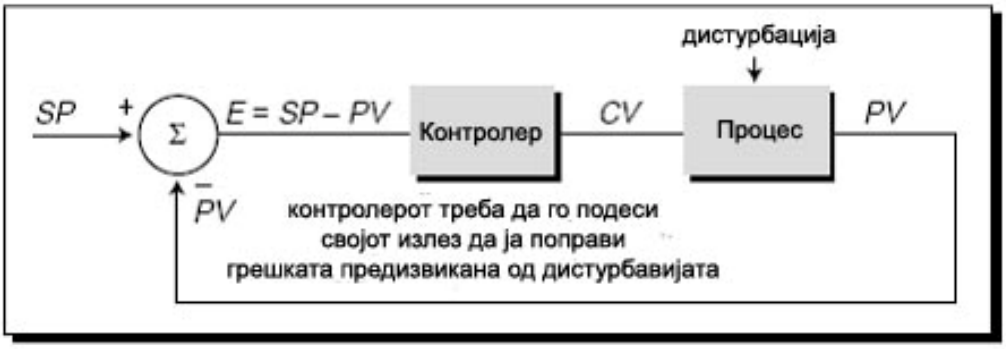
Процесна контрола е регулација на параметрите на процесот во определен опсег или поставување на командна вредност. Процесна контрола најчесто се користи во фабричкото производството, бидејќи многу фактори како боја, композиција, густина, облик, траекторија мора да бидат прецизни за производот да биде добро направен. Затоа, за да се добие квалитетен производ, се користи процесната контрола за надзор и корегирање на параметрите со анализа на состојбата на динамичките променливи. Динамички променливи се карактеристики на процесот, како температура, проток, притисок, позиција, снага и итн. Низ неговиот влезен интерфејс PLC можи да ги регулира динамичите променливи за постигнување на командната вредност на тој начин имплементирајќи ја процесната контрола.

Во нашиот случај контролерот за движење DMC-14XX ги контролира моторите за постигување на бараната позиција додека ласер контролерот UC-2000 ја контролира снагата и тајмингот на испукување на ласерот.

Девијацијата на контролата или грешката помеѓу командаата вредност и процесната променлива е дадена со равенакта.

$$E = SP - PV$$

каде:
E - грешка
SP- командна вредност
PV- процесна променлива
CV-контролна променлива



За време на контрола на процесот контролерот ја пресметува вредноста на грешката и ја подесува контролната променлива така да ја доведи грешката на нула. Контролерот произведува инверзија од 180^o на повратниот сигнал за да произведе негативна повратна врска и да ја намалува системската грешка преку контролната променлива.

За да се види колкава е грешката во однос на командната вредност се користи следната равенка:

$$E = \frac{SP - PV}{SP}$$

Така за зададена командна вредност SP=125^o на моторот и измерена вредност на процесната променлива PV=120^o имаме грешка од 4% во однос на командната вредност.

Изразување на грешката како процент од опсегот (max-min) на процесната променилива е дадена со:

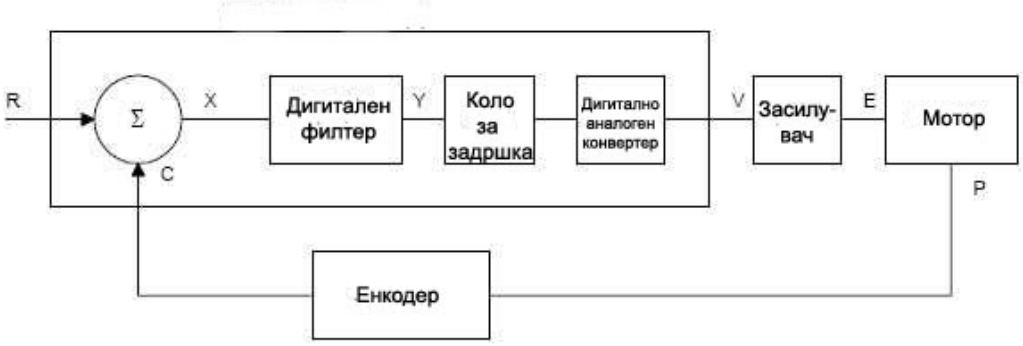
$$E\% = \frac{SP - PV}{PV_{min} - PV_{max}}$$

каде:

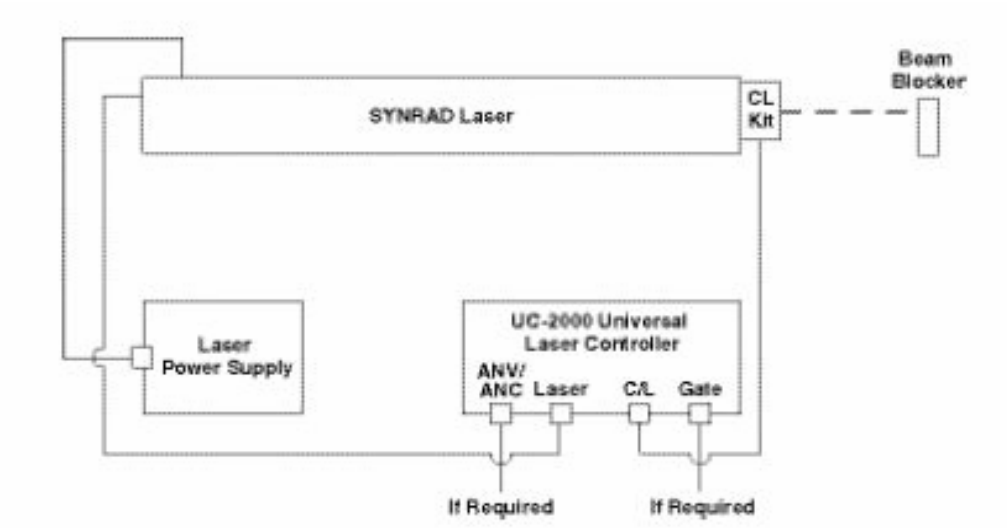
E% = грешка како процент од опсегот на вредности на PV
SP =командна вредност
PV =процесна променлива
PV_{min} = максимална вредност PV
PV_{max} =минимална вредност PV

Така за зададена командна вредност SP=180^o на моторот и измерена вредност на процесната променлива PV=168^o во опсег од (100^O-200^O) имаме грешка од -12% (негативниот знак значи дека вредноста на PV<SP)

Пример за наша процесна контрола на двата контолери во нашиот системи е дадена на следнате слики



Блок шема на конторла на процес на движење



Блок шема на контрола на процес на ласерско испукување

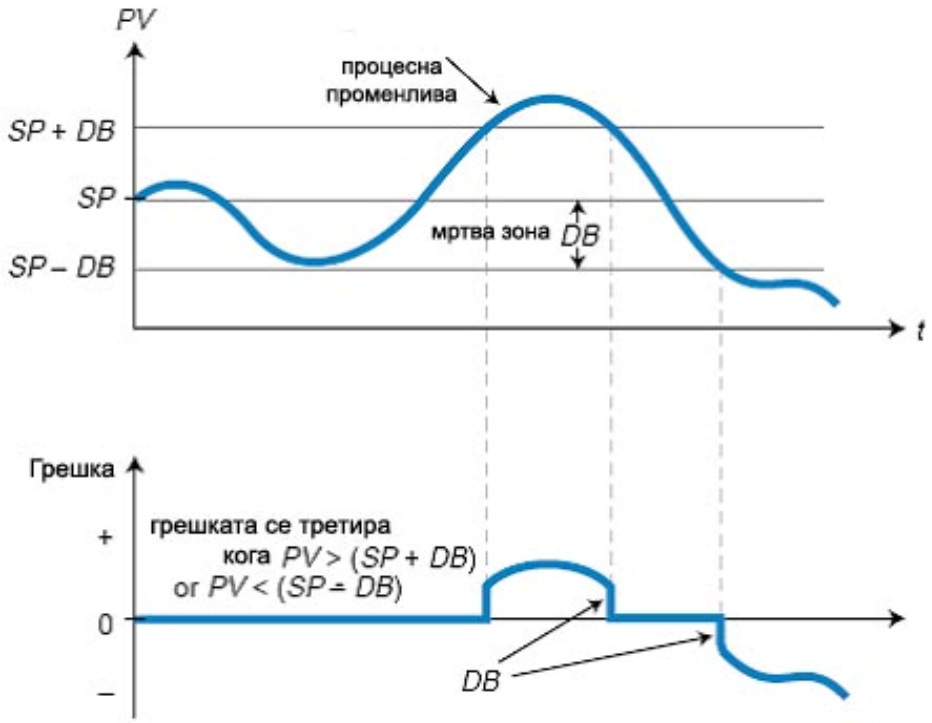
Како и грешката така и контролната променлива можи да се изрази во проценти од опсегот.

$$CV\% = \frac{CV_{actual} - CV_{min}}{CV_{max} - CV_{min}}$$

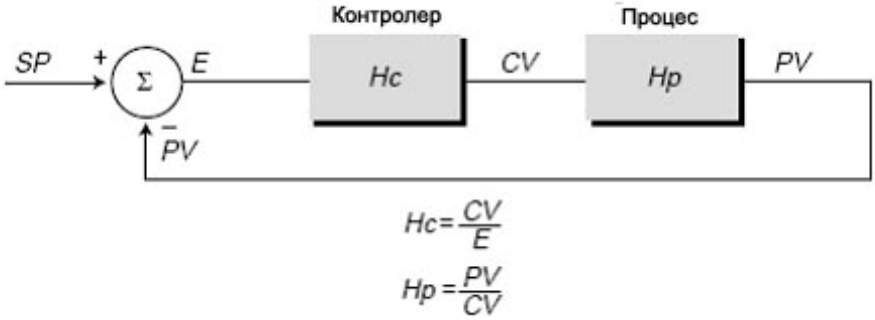
CV% = грешка како процент од опсегот на вредности на PV
CV_{actual} = моментална вредност на CV
CV_{min} = максимална вредност CV
CV_{max} =минимална вредност CV

МРТВА ЗОНА НА ГРЕШКАТА

Во основа сите системи имаат дозволена флуктуација на грешката, што значи дека грешката можи да варира од нула до одредена вредност без да го наруши финалниот продукт. Тоа дозволена флуктуација е прикажана на слика и се вика **мртва зона на грешката** (*dead band* - DB).
Во внатрешноста на мртвата зона, контролерот ги третира грешките како да се нула, што значи дека не превзема корективни акции врз контролната променлива. Ако грешката има девијација поголема од мртвата зона, контролерот ќе превземе корективна акција со измена на неговиот излез (*control variable* - CV)

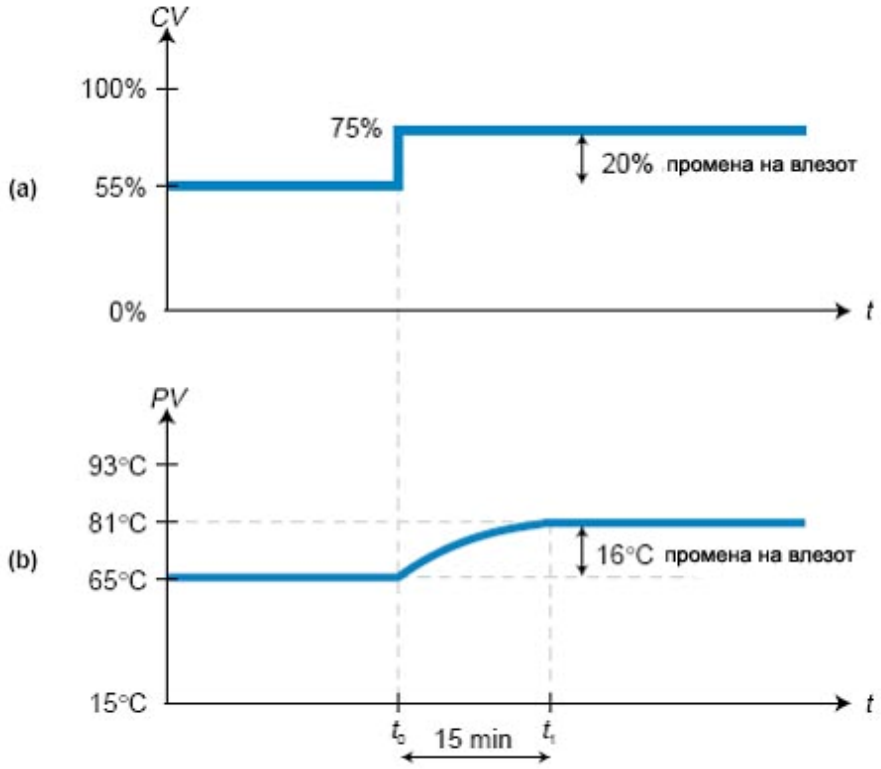


Терминот динамички, каков што се користи во процесната контрола, се однесува на промените што настануваат во процесот заради присутната дистурбација или промена на командната вредност. Процесот се одзива низ процесната променлива на динамичките промените на контролната променлива во сооднос на карактеристиката на процесот. Оваа карактеристика на процесот, што вклучуваат фактори како каснење и наследен физички одзив на процесот е дефиниран како **преносна функција,** презентирана со термот *Ht*. (види слика)



Преносната функција е равенка која го опишува одзивот на процесот во времето, а воедно ја пресметува процесната променлива. Затоа, вредноста на *Ht* еднаква е на процесната променлива при одредена контролна променлива и време, дадена со карактеристиката на процесот. Секој процес има своја уникатна преносна функција базирана на неговата одредена карактеристика, и за повеќето од процесите, преносната функција е позната. Најважен аспект на преносната функција не е е толку во композицијата или формата, туку во одзивот на изненадните процесни промени предизвикани од дистурбација или промена на командната вредност. Овој особински одзив на процесот е наречена **трансцедентен одзив** и го вклучува времето потребно излезот да го достигне стабилна финална состојба дадена со изненадна промена на влезот. Трансцидентниот одзив обезбедува повеќе информации за динамиката на процесот од преносната функција.

15 мин. период на трансцедентниот одзив на преносната функција при промена на контролната променлива е дадена на сликата



ПРОЦЕСНО ЗАСИЛУВАЊЕ

Процесниот **коефициент на засилување**, преставено со *K* го дефинира односот меѓу процесниот излез и влез. Коефициентот на засилување е друг динамички елемент кој е набљудуван во трансцедентниот одзив. Се пресметува со делење на промената на процесниот излез во времето со кореспондентната промена на процесниот влез односно однос меѓу промената на процесната променлива и промената на контролната променлива.

$$K = \frac{PV_{\text{final}} - PV_{\text{initial}}}{CV_{\text{final}} - CV_{\text{initial}}}$$

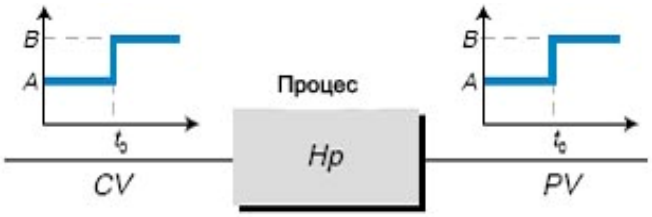
Во случај на промена на контролната променлива предизвикана од промена на командната вредност од операторот од 65 W снага на 81 W снага, *K* би било промената на снагата во однос на промената на излезот под дејство на таа промена со стабилизационо време од 15с.

$$K = \frac{81W - 65W}{75\% - 55\%} = 0,8W/\%$$

Засилување од 0.8W/% значи дека процесната променлива (снагата) се менува 0.8W со секој еден процент промена во контролната променлива

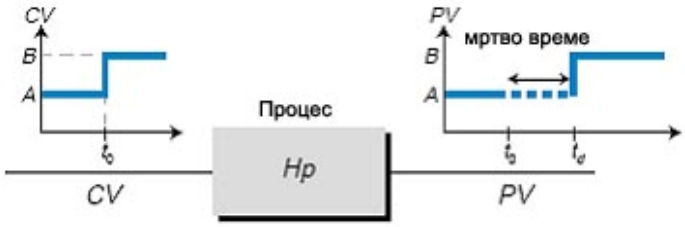
МРТВО ВРЕМЕ (Dead Time)

Перфектен одзив на процесната променлива на отскочна промена во контролната променлива е прикажан на слика и процесната преносна карактеристика е еднаква на 1, (види слика)



$$Hp = \frac{PV}{CV} = 1$$

што значи дека контролна променлива на влезот веднаш резултира со еднаква процесна променлива на излез .
Во реалноста тоа не е случај и еден од карактеристиките на процесите е каснењето асоцирано со одзивот на излезот на влезен сигнал. Ова каснење се вика мртво време.



$$Hp_{(s)} = e^{-t_d s}$$

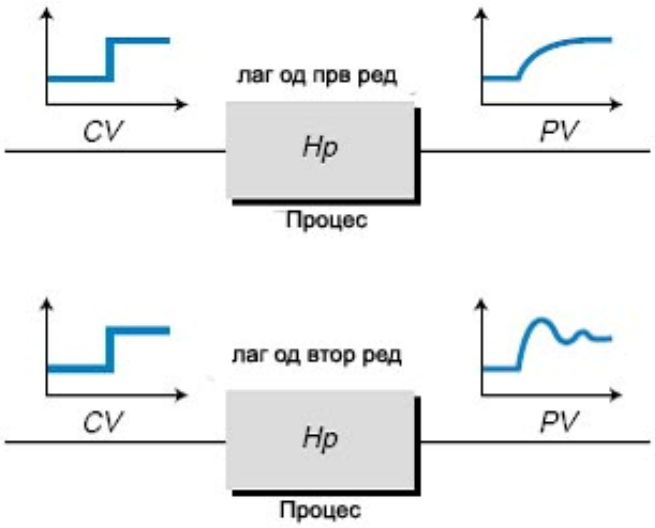
И покрај тоа дека каснењето на сензорите, во најголем дел од случаеви, е мало во однос на мртвото време на процесната променлива, сепак е важен дел и треба да се земи во обзир на брзореагирачките системи со повратна врска, како серво моторите и другите апликации за позиционирање.

ЛАГ ВРЕМЕ

За разлика на мртво време каснењето, кое е каснење помеѓу промената на влезот и иницијалниот одзив на процесната променива на влезната промена, лаг време е времето од иницијалниот одзив на процесната променлива до постигнување на оптималниот одзив. Лаг настанува ради карактеристиката на процесот и е дел од преносната функција.

- лаг од прв ред
- лаг од втор ред

Лаг од прв ред е лаг на процесната променлива и преставува одзив на брзи промени на контролната променлива. *Лаг од втор ред* е осцилирачки одзив на процесната променлива со смирување кон стабилна состојба после отскочена промена на влез.



Повеќето процеси во индустријата можи да се апроксимираат со диференцијални равенки од прв и втор ред.
Преносната функција на процес со лаг од прв ред може да се апроксимира со диференцијална р-ка од прв ред :

$$y_{(t)} = \frac{dx}{dt} + x_{(t)}$$

чија Лапласова трансформација е:

$$Y_{(s)} = sX_{(s)} - x_{(t=0)} + X_{(s)}$$

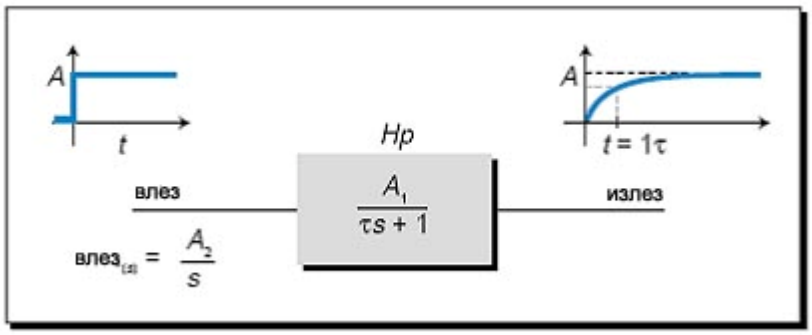
Ако $X_{(s)}$ Лапласов излез на процесот и $Y_{(s)}$ преставува влез, со претпоставка на дека $x_{(t=0)}$ е нула и делење на влезот со излезот се добива преносната функција

$$\frac{X_{(s)}}{Y_{(s)}} = \frac{1}{s+1}$$

Преносната функција на систем со лаг од прв ред во Лапласова форма е :

$$Hp_{(s)} = \frac{A_1}{\tau s + 1}$$

каде :
A_1 = *процесно засилување*
τ = *системски лаг*



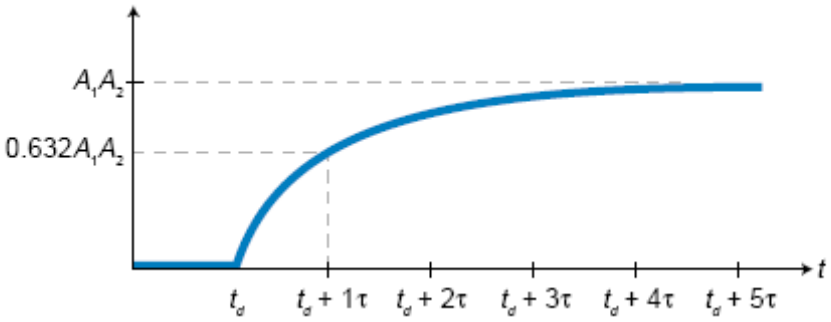
Одзив на системот на отскочна побуда е даден со Лапласовата р-ка:

$$\begin{aligned}\mathrm{Out}_{(s)} &= \left(\mathrm{In}_{(s)}\right)\left(Hp_{(s)}\right)\\ &= \left(\frac{A_2}{s}\right)\left(\frac{A_1}{\tau s+1}\right)\\ &= \frac{A_1A_2}{s(\tau s+1)}\end{aligned}$$

Со додавање на мртвото време во Лапласов облик e^{-tds} на одзивот со лаг од прв ред генерира Лапласова равенка:

$$\mathrm{Out}_{(s)} = \left(\frac{A_1A_2}{s(\tau s+1)}\right)e^{-t_d s}$$

и облик на преносна функција прикажан на слика



Преносна функција си лаг од втор ред може да се апроксимира со диференцијална р-ка од втор ред :

$$y_{(t)} = \frac{d^2x}{dt^2} + \frac{dx}{dt} + x_{(t)}$$

чија Лапласова трансформација е:

$$Y_{(s)} = \left[s^2X_{(s)} - s\frac{dx_{(t=0)}}{dt} - x_{(t=0)}\right] + \left[sX_{(s)} - x_{(t=0)}\right] + X_{(s)}$$

Ако $X_{(s)}$ Лапласов излез на процесот и $Y_{(s)}$ преставува влез, со претпоставка на дека $x_{(t=0)}$ е нула и делење на влезот со излезот се добива преносната функција

$$\frac{X_{(s)}}{Y_{(s)}} = \frac{1}{\left(s^2+s+1\right)}$$

Преносната функција на систем со лаг од прв ред во Лапласова форма е :

$$Hp_{(s)} = \frac{\mathrm{Out}}{\mathrm{In}} = \frac{A\omega_n^2}{\left(s^2+2\zeta\omega_n s+\omega_n^2\right)}$$

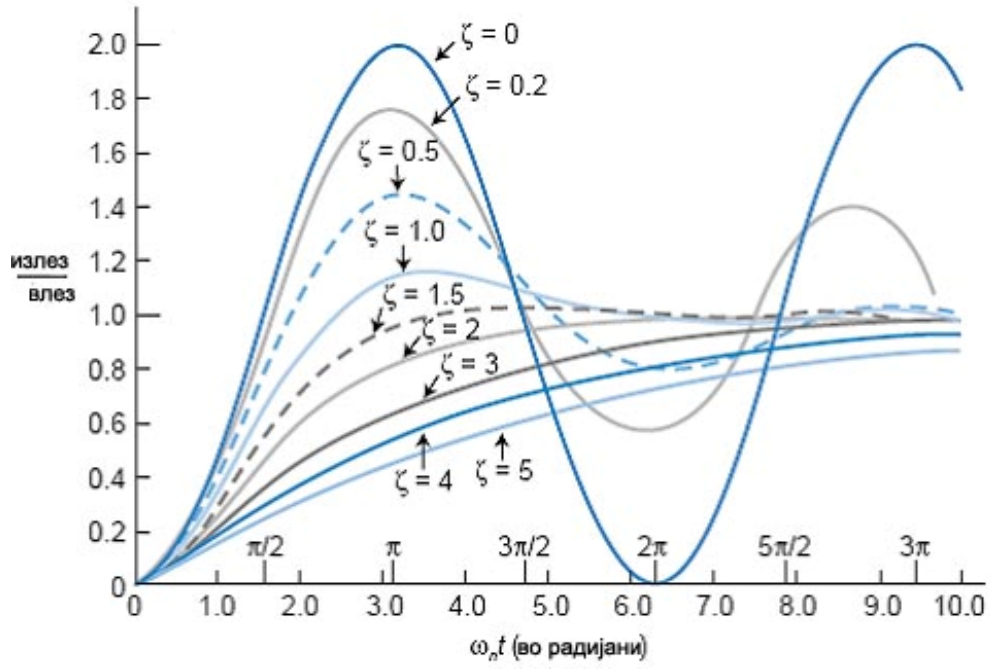
A = *процесно засилување*

ζ = *фактор на придушување*

ω_n= *резонантна фреквенција*

$K_{sys}= A\omega_n^2$ = *засилување на системот*

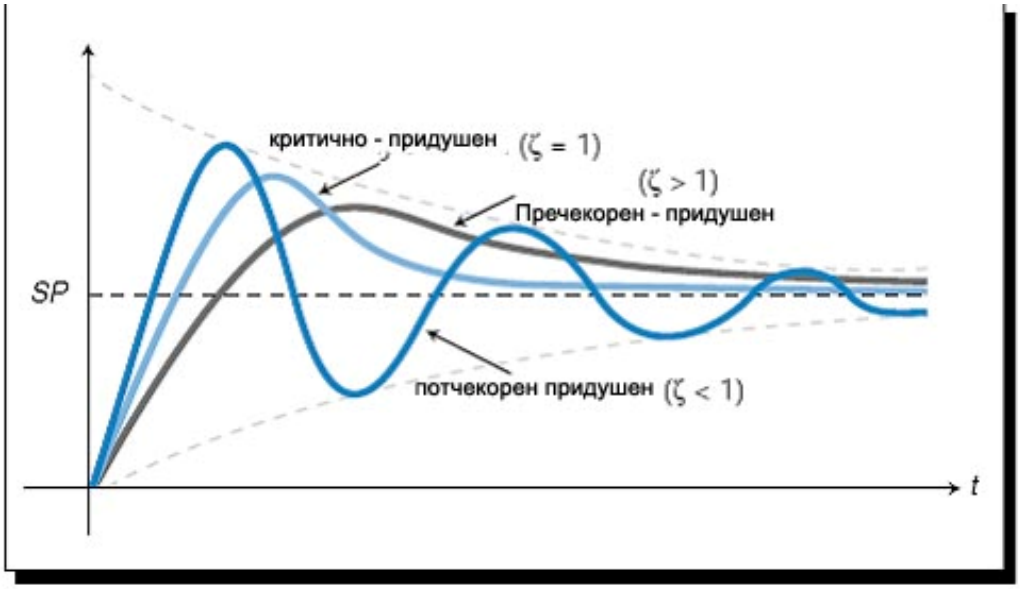
Фрекфенцијата ω_n е фактор кој одредува колку брзо одзивот осцилира над и под посакуваната вредност. Фактор на придушување ζ е фактор кој ги гаси осцилациите со времето, така да одзивот на крајот се нивелира со саканата излезна вредност. Амплитудата на осцилациите се гаси експоненцијално со факторот на пригушување изразен во инверзна Лапласова трансформација $e^{-\zeta\omega_n t}$. Ако факторот на придушување (ζ) е 0 од што јасно се гледа дека $e^{-\zeta\omega_n t}$ ке биде 1 што значи одзивот ке преставува бесконачна синусуида. Обликот на одзивот во зависност од различни вредноста на факторот на придушување е даден на слика



Процес од втор ред може исполнува еден од трите типови на одзив:

- *Пречекорен - придушен (critically damped ζ >1)*
- *Критичпо – придушен (overdamped ζ =1)*
- *Подчекорен - придушен (underdamped ζ <1)*

Овие одзиви се разликуваат во тоа како ја постигнуваат крајната стабилна вредност или командна вредност со текот на времето и се во дирекна зависност од факторите на придушување (види слика) .



За разлика од процес од прв ред, процес од втор ред има две лаг времиња $(\tau_1$ и $\tau_2)$ кои се поврзани со фрекфенцијата на осцилации (ω_n) и преносна функција:

$$Hp_{(s)} = \left(\frac{A_1}{\tau_1 s + 1}\right)\left(\frac{A_2}{\tau_2 s + 1}\right)$$

ПРЕЧЕКОРЕН ПРИДУШЕН ОДЗИВ

Пречекорен придушените одзиви се одзиви од втор ред каде ζ >1

$$Hp_{(s)} = \left(\frac{A_1}{\tau_1 s + 1}\right)\left(\frac{A_2}{\tau_2 s + 1}\right)$$

а ако се спореди со:

$$Hp_{(s)} = \frac{A\omega_n^2}{\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right)}$$

$$Hp_{(s)} = \frac{\overbrace{\left(\frac{AA_2}{\tau_1\tau_2}\right)}^{K_{sys}}}{\left(s^2 + \underbrace{\frac{(\tau_1+\tau_2)}{\tau_1\tau_2}}_{2\zeta\omega_n} s + \underbrace{\frac{1}{\tau_1\tau_2}}_{\omega_n^2}\right)}$$

$$\omega_n = \sqrt{\frac{1}{\tau_1 \tau_2}}$$
$$= \frac{1}{\sqrt{\tau_1 \tau_2}}$$

$$K_{sys} = A\omega_n^2 = \frac{A_1 A_2}{\tau_1 \tau_2} = \frac{K_{OD}}{\tau_1 \tau_2}$$

$$2\zeta\omega_n = \frac{(\tau_1 + \tau_2)}{\tau_1 \tau_2}$$

Преносната функција на процес од втор ред во временски домен е преставен со инверзна Лапласова трансформација:

$$H_{(t)} = \frac{K_{OD}}{\tau_1 - \tau_2}\left(e^{\frac{-t}{\tau_1}} - e^{\frac{-t}{\tau_2}}\right)$$

КРИТЧНО ПРИДУШЕН ОДЗИВ

Критично придушените одзиви се одзиви од втор ред каде ζ =1 и $\tau = \tau_1 = \tau_2$:

$$H_{(s)} = \frac{A}{(\tau_1 s + 1)(\tau_2 s + 1)}$$
$$= \frac{A}{(\tau s + 1)^2}$$

а ако се спореди со:

$$Hp_{(s)} = \frac{A\omega_n^2}{\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right)}$$

$$= \frac{\left(\frac{A\omega_n^2}{\omega_n^2}\right)}{\left(\frac{s^2}{\omega_n^2} + \frac{2s\omega_n}{\omega_n^2} + \frac{\omega_n^2}{\omega_n^2}\right)}$$

следи:

$$\tau_{sys} = \sqrt{\tau_1 \tau_2}$$

$$\zeta = \frac{\tau_1 + \tau_2}{2\sqrt{\tau_1 \tau_2}}$$

$$\tau_{sys} = \sqrt{\tau_1^2} = \tau_1$$

or

$$\tau_{sys} = \sqrt{\tau_2^2} = \tau_2$$

Бидејќи $\tau_1 = \tau_2$ во овој тип на одзиви следи:

$$\zeta = \frac{2\tau}{2\sqrt{\tau^2}}$$

$$= \frac{2\tau}{2\tau}$$

$$= 1$$

Преносната функција на процес од втор ред во временски домен е преставен со инверзна Лапласова трансформација:

$$\mathcal{L}^{-1}\left[\frac{K_{sys}}{s(\tau s + 1)^2}\right] = K_{sys}\left[1 - \left(\frac{\tau - t}{\tau}\right)e^{\frac{-t}{\tau}}\right]$$

ПОДКОРАЧЕН ПРИДУШЕН ОДЗИВ

Подкорачен - придушен одзив исполнува и прчекорување и подчекорување т.е преставува осцилирачки одзив со резонантна фрекфенција (ω_n) rad/s. Причина за осцилациите е факторот на придушување кој е помал од 1. Ова значи дека наместо да го факторира деноминаторот во полиномот $(s^2 + 2\zeta\omega_n s + \omega_n^2)$ на р-ката на преносната функција на процес од втор ред, деноминаторот станувам комплексен корен на квадратна р-ка, па преносната функција на отскочен одзив во временски домен е дадена преку инверзна Лапласова трансформација.

$$\mathcal{L}^{-1}\left[\left(\frac{1}{s}\right)\left(\frac{A\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}\right)\right] = A\left[1+\frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}}\sin\left(\omega_n\sqrt{1-\zeta^2}\,t-\psi\right)\right]$$

За мали вредности на ζ одзивот на откочен влез може да се апроксимира со:

$$\underset{\text{unit step}}{\text{Out}_{(t)}} = A\left[1+e^{-\zeta\omega_n t}\sin(\omega_n t)\right]$$

$$H_{(t)} \approx e^{-\zeta\omega_n t}\sin(\omega_n t) \approx \sin(\omega_n t)\Big|_{\zeta=0}$$

Временската константа на процесот τ_{sys} е:

$$\tau_{sys} = \frac{1}{\zeta\omega_n}$$

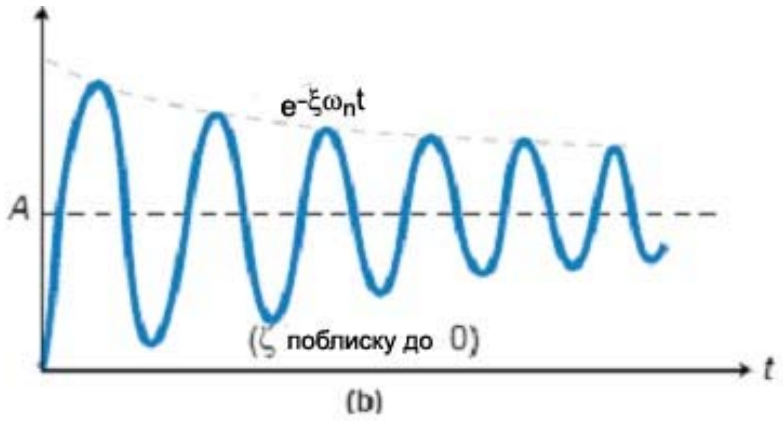
or

$$\zeta\omega_n = \frac{1}{\tau_{sys}}$$

па $e^{-\zeta\omega_n t}\sin(\omega_n t)$

$$e^{\frac{-t}{\tau_{sys}}}\sin(\omega_n t)$$

Што е поблиску факторот на придушување на 1, помала е фрекфенцијата на осцилирање е помала и поскоро ќе дојде до нивелирње на одзивот. Што е поблиску до 0 се зголемува фрекфенцијата и осцилациите траат подолго за да при вредност еднаква на 0 се добие бесконечен синусоидален одзив.

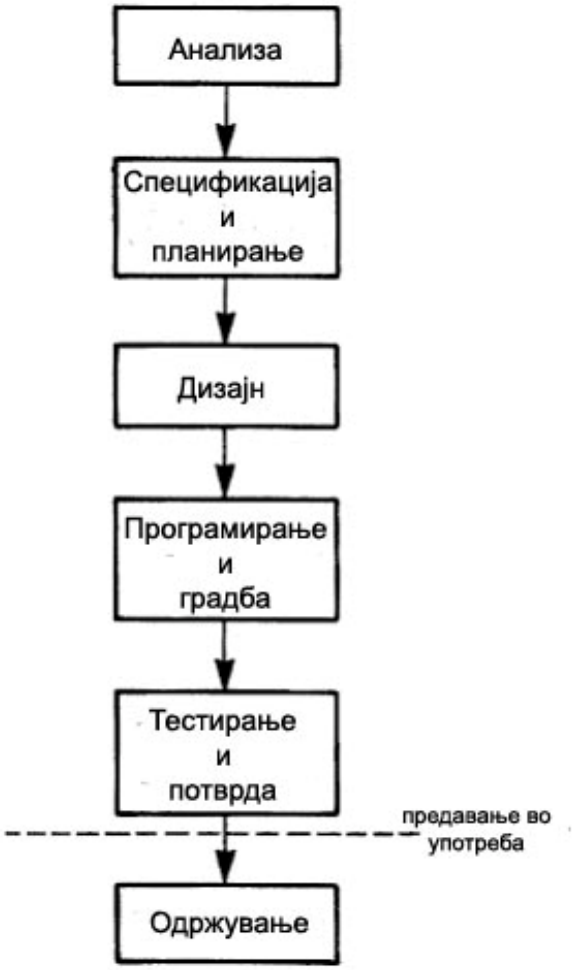


(b)

2.3 ПРОГРАМИРАЊЕ

Првата фаза на програмирањето е *анализа* на проблемот што треба да се реши. Програмерот на PLC системот мора да се сретни со договарачите и корисникот за да одреди каква контрола е потребна и како контролата ке биде обезбедена. Важни работи, како операторската контрола, треба да се земат во предвид и двосмислените описи треба да се разрешат.

Од сите фази, *анализата* е најтешка, затоа што крајниот корисник и другите договарачи вероватно не го земат во предвид контролната стратегија, односно немаат искуство да одлучат дали дел од фабриката е најдобро конторолирана од џојстик, копче и допирен екран.



Излезот од аналитичката фаза треба да биде опис на работата на фабриката, каква операторска станица и контороли се потребни (и како тоа ке биди имплементирано), каква помош и уреди за пронаоѓање на грешка/одржување ќе бидат вклучени и на крај(но не само тоа) комплетна листа на влезно/излезни сигнали со напонско-струјни спецификации и нивните локации во фабриката.
Тешкотиите (и важноста) на првата фаза неможат да се преувеличаат. Ако двосмисленостите и проблемите се решат на почеток, следните фази се лесни.

Ова фаза не трпи претпоставки и потребно е да се прашува/разрешува без разлика дали постои сомневање или не. Во оваа фаза треба да се дефинираат крајните барања за тестирање.

Следна фаза , познат како top-down, е *дизајн* на системот; коцки, делови и структура на програмот.
На крајот ќе може да се програмира, изградено врз основа на структурата поставена во фазата на дизајн. Ниеден програм не треба да се конструира *ad hoc* на тастатура (тн. Спагети програмирање). Комерцијалните програмери проценуваат дека ова фаза главно вклучува не повеќе од 10% од вкупниот напор.

Кога програмирањето ке биде завршено и фабриката изградена, може да почне *тестирањето и потврдата*. Операцијата треба да се провери во однос на преходната фаза. Проверувањето на сите делови и на наједноставен систем одзема многу време. На тоа време се додаваат и каснењата на сите претходни фази па важно е да се постави кое тестирање мора да се изврши а кое може да се тестира подоцна за време на работа. Тестирањето за време на работа, како и да е, е многу тешко и временски долго. Сигурносно поврзаните проверки не треба никогаш да се прескокнат

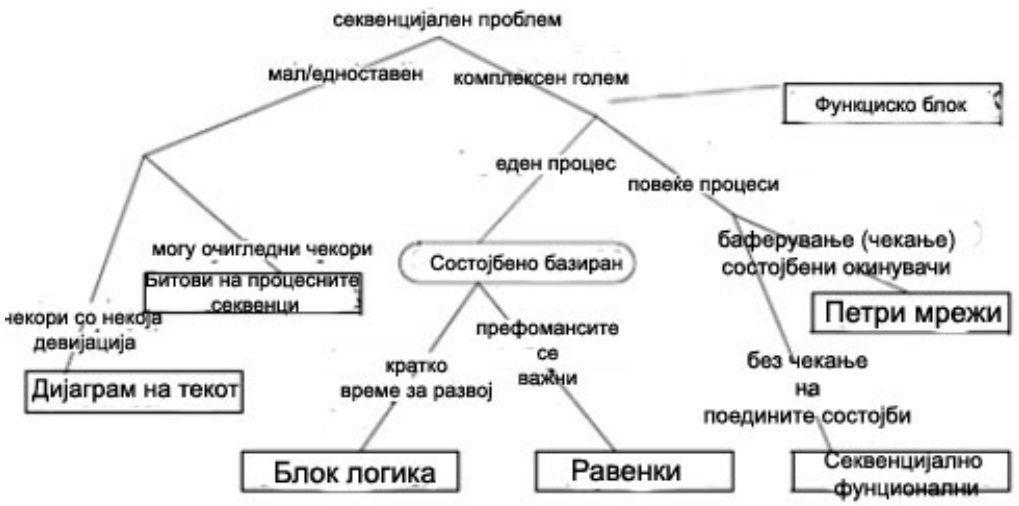
Финалната фаза најчесто е занемарена. Еднаш кога фабриката е предадена во употреба, контролниот систем треба да се одржува и тоа не сервисирање во механичка смисла, туку покривање на пронаоѓањето на грешки, разрешувањето на багови и (се надеваме мали) промени произлезени од модификациите на начинот на кој функционира фабриката. Ниедна фабрика не е непроменлива, се се менува во текот на нејзиниот животот како одговор на пазарот и технолошките промени, и тие модификации бараат промена на контролната стратегија.

Во комерцијалните програми одржувањето одзема преку 50% од напорот во животниот циклус на проектот. Затоа е од особена важност да се конструираат контролната стратегија и програм и да се документираат така да можат да бидат лесно подложни на промени и модификации во подоцнежните фази, најверојатно од луѓе кои не се вклучени во првите 5 фази.

Методи на програмирање

10% од времето во лош програмски дизајн отпаѓа на дизајн, 30% на пишување, 40% на дебагирање и тестестирање и 10% на документација. Високо квалитетниот програмски дизајн е со следно временско распределување и тоа 30% дизајн, 10% пишување на софтвер, 10% дебагирање и тестирање и 10% на документација. Од горенаведните статисички податоци јасно се гледа важноста на фазата за дизајнирање.

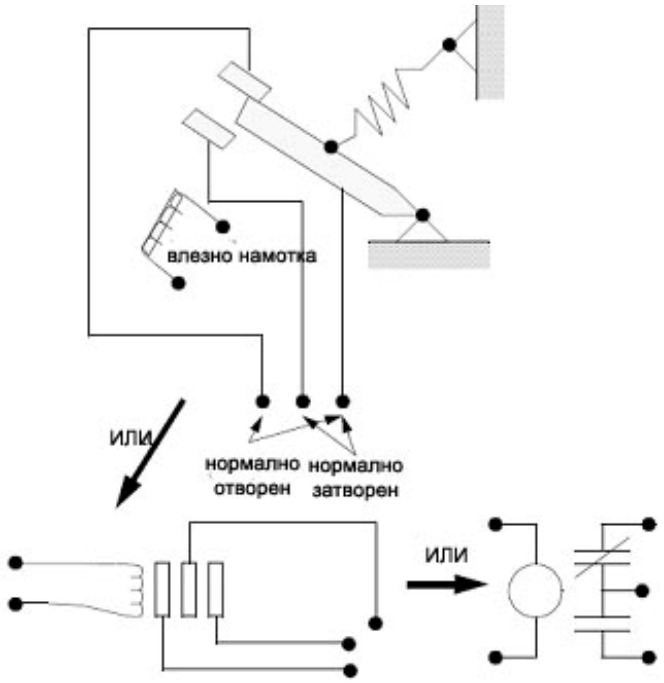
За дизајнирање се различни употребуваат различни методи на програмирање на PLC во зависност од комплексноста на проблемот, потребните перформанси, бројот на процеси кои треба да се контролираат, методите кои се подржани од производителот на PLC и др.....



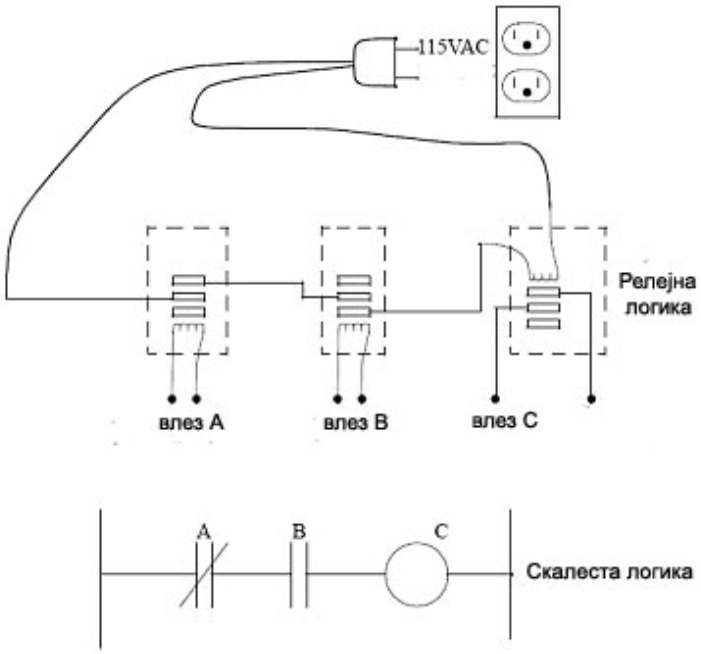
Скалеста логика (Ladder logic)

Скалеста логика е главен програмирачки метод кој се користи кај PLC –та. Овој метод е развиен да ја имитира релејна логика и да се намали бројот на инженери кои треба да се преквалифицираат.

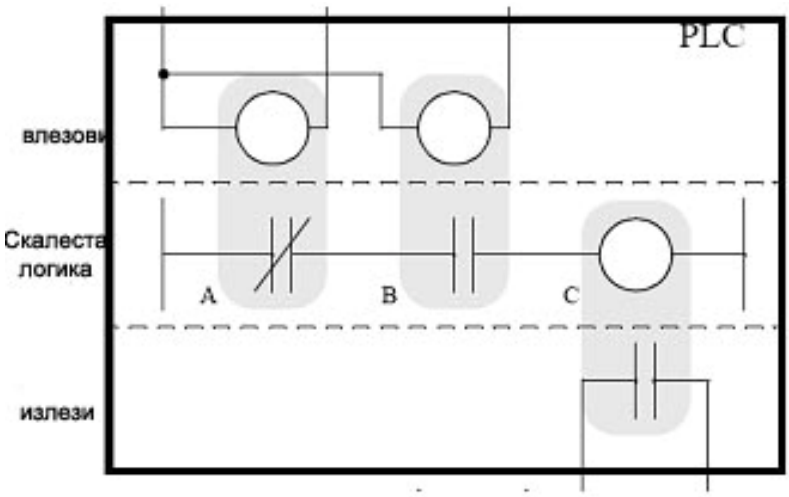
Модерните контролни системи сеуште соодржат релеи, но многу ретко за логички дизајн. Релеата нормално се цртаат шематски со круг кој ја преставува влезната намотка. Излезните контакти се прикажуваат со две паралелни линии. „Нормално отворени“ контакти се прикажани со две линии и ќе бидат отворени (не проведуваат) кога нема напон на влезната намотка. „Нормално затворените“ контакти се прикажани со две линии пресечени со дијагонална линија и ќе бидат затворени (ќе проведуваат) се додека нема напон на влезната намотка.



Пример на реле во едноставна контролна апликација е дадено на слика. Во системот првото реле се користи како „нормално затворено“ и ќе дозволи течење на струјата се додека не биде аплицирана волтажа на *влезот A*. Второто реле е „нормално отворено“ и нема да дозволи течење на струјата се додека не биде аплицирана волтажа на *влезот B*. Ако струјата течи низ првите две релеа тогаш таа ќе течи и низ третото реле, и ќе го затвори прекинувачот за *излезот C*.



Колото нормално се црта во скалестата логичка форма и можи да се прочита како „C ке биди уклучено односно на излезот на PLC ќе биди на 1, ако е A е исклучено и B уклучено (влезот на PLC е A=0 и B=1)

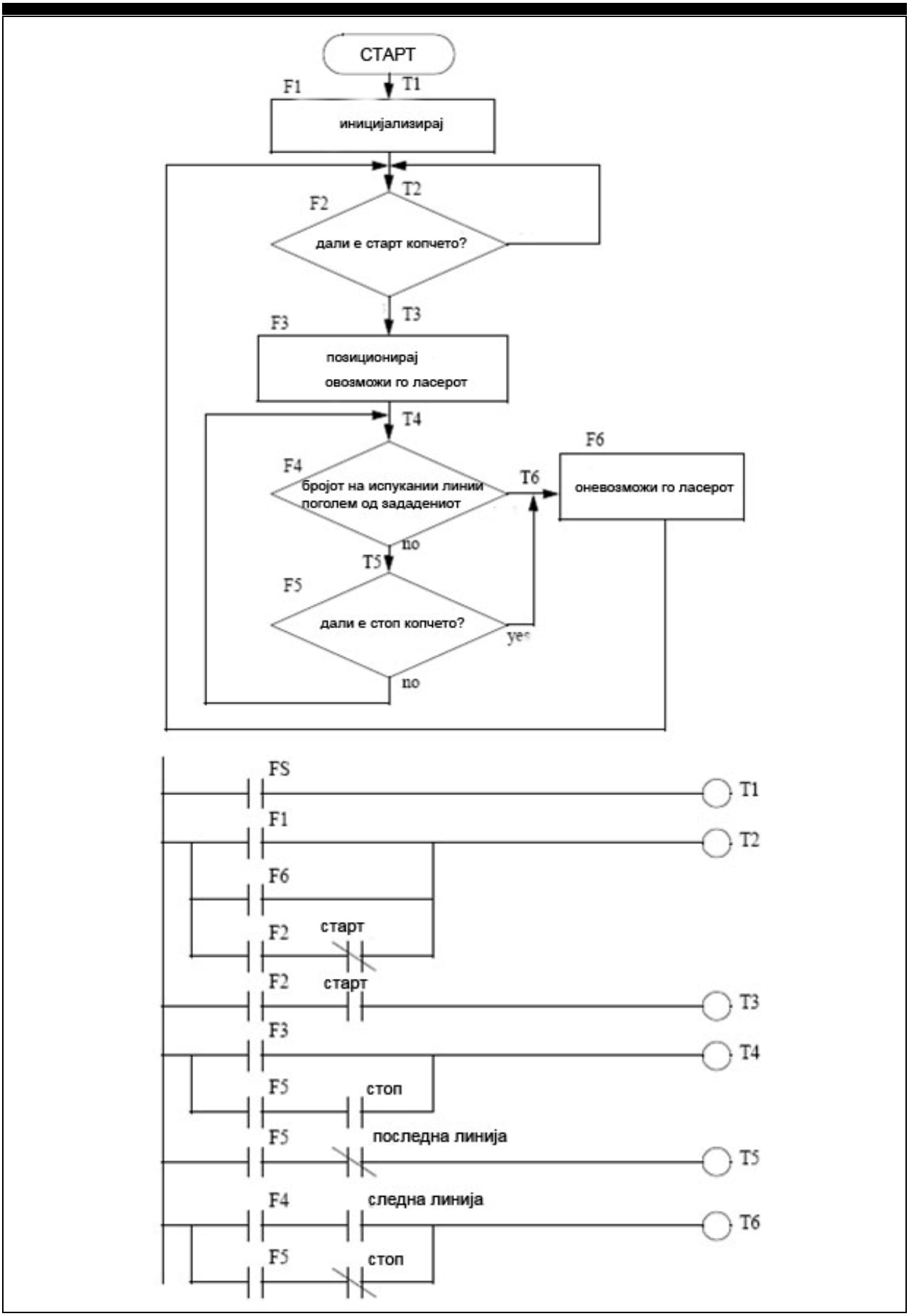


Дијаграм на текот(Flowchart)

Дијаграм на текот е идеален за процеси со секвенвијални(редоследни) чекори. Чекорите ќе бидат егзекутирани по едноставен редослед кој можи да се промени како резултат на некои едноставни одлуки. Симбоилите кои се користат во дијаграмите на текот се прикажани на слика() Овие блокови се поврзани со стрелки кои ја покажуваат секвенцата на чекори. Различните блокови имплицираат различни типови на програмска акција. На програмите секогаш им треба *start* блок, но PLC програмите ретко застануваат па *stop* блок ретко се користи. Другите важни блокови вклучуваат *операции* и *одлуки*. Можат да се користат и други функции но не се потребни во најголем број PLC апликации.

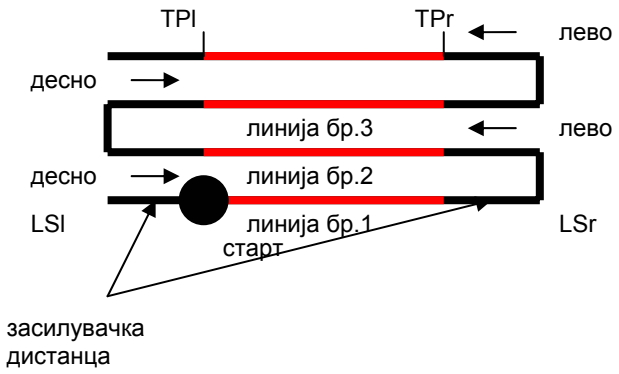
Чекорите што се употребуваат при креирањето на дијаграмот:

1. Разбирање на процесот
2. Одредување на главните акции, кои се цртат како блокови
3. Одредување на секвенците на операциите, кои се цртаат со стрелки
4. Секвенцата на извршување може да се промени со користење на условни блок за гранење


СТАРТ
F1
T1
иницијализирај
T2
F2
дали е старт копчето?
T3
F3
позиционирај
овозможи го ласерот
T4
F6
F4
бројот на испукании линии
поголем од зададениот
T6
оневозможи го ласерот
no
T5
F5
дали е стоп копчето?
yes
no
FS
T1
F1
T2
F6
F2
старт
F2
старт
T3
F3
T4
F5
стоп
F5
последна линија
T5
F4
следна линија
T6
F5
стоп

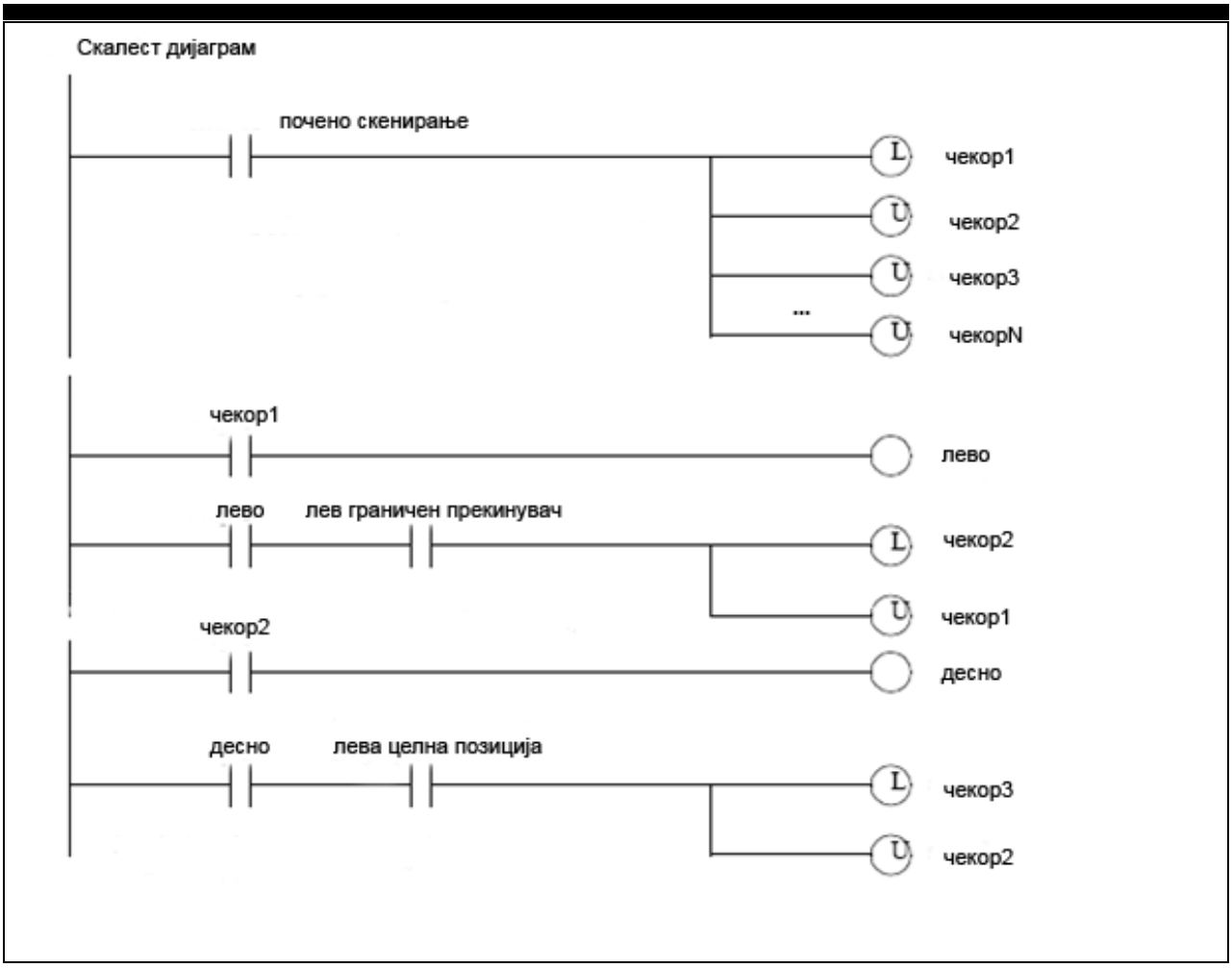
Битови на процесните секвенци (Process sequence bits)

Овој метод ги користи следните чекори на програмирање:
1. Разбирање на процесот
2. Запишување на чекорите на операцијата во секвенца и означување на секој чекор со број.
3. За секој чекор се доделува бит.
4. Се црта скалест дијаграм за уклучување/исклучување на битовите како процесот преминува низ состојбите.
5. Се црта скалест дијаграм за изведување на неговите машински функции за секој чекор
6. Ако процесот се повторува, направи последниот чекор да се врати на првиот.



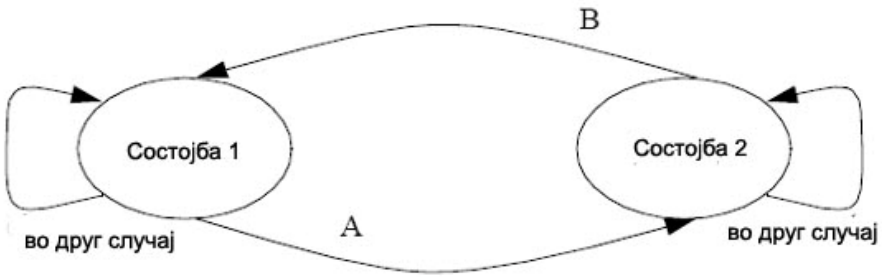
Ласерот е позициониран на почетната линија и се чека старт наредба. По давањето на старт наредбата се движи лево десно менувајќи ја насоката и се поместува на следната линија во зависност дали удрил во левиот или десниот граничен прекинувач и исцртувајќи ја линијата ограничена со левата и десната целна позиција. Процесот завршува со кога сите линии ќе бидат изгравирани.

Чекори:
1. Со стартот ласерот се движи лево
2. се додека не удри во левиот граничен прекинувач LS_l при што ја менува насоката на движење во десно
3. Во моментот на достигнување на левата целна позиција TP_l се стартува ласерскиот сноп
4. Во моментот на достигнување на десната целна позиција TP_r се стопира ласерскиот сноп
5. се додека не удри во десниот граничен прекинувач LS_l при што ја менува насоката на движење во лево
6. Во моментот на достигнување на левата целна позиција TP_l се стопира ласерскиот сноп
7. Во моментот на достигнување на десната целна позиција TP_r се стартува ласерскиот сноп
...



Состојбено базиран дизајн (State based design)

Состојбениот систем може да се опиши како системски состојби и транзиции помеѓу тие состојби. Состојбен дијаграм е прикажан на слика (). Дијаграмот има две состојби *Состојба 1* и *Состојба 2*. Ако системот е *Состојба 1* и ако се исполни е условот *A,* системот ќе оди во *Состојба* 2 , во друг случај ќе остане во *Состојба 1*. Слично ако системот е во *Состојба* 2, и ако се исполни условот *B* системот ќе се врати во состојба *Состојба 1*.

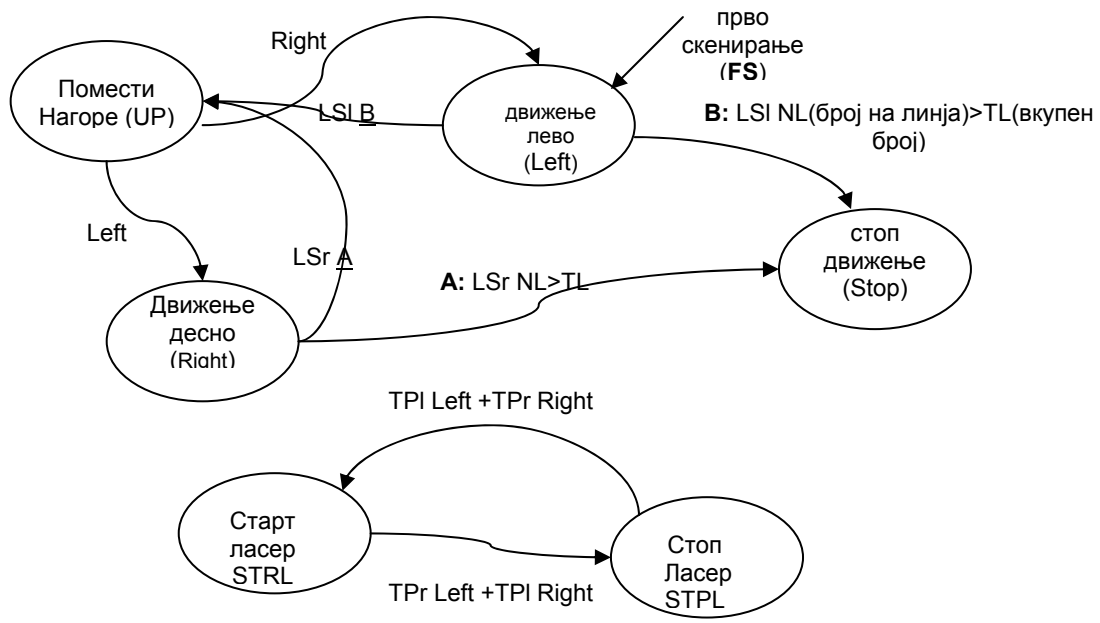


Формулата за испишување на равенките за секоја состојба кои се минимизираат со примена на Булова алгебра, Карнови карти, Деморганови закони... е дадена:

$$STATE_i = \left(STATE_i + \sum_{j=1}^{n} (T_{j,i} \bullet STATE_j) \right) \bullet \prod_{k=1}^{m} \overline{(T_{i,k} \bullet STATE_i)}$$

Каде, $STATE_i$ е

n – број на транзиции кои водат до $STATE_i$
m – број на транзиции кои водат од $STATE_i$
$T_{j,l}$ – логички услов за транзиција од j до i
$T_{i,k}$ – логички услов за транзиција од i до k



Пример за искорисување на состојбениот дизајн во егзапларниот систем е даден на сликата од кој е се изведени следните р-ки:

$$Left = (Left + Right \bullet UP) \bullet \overline{Left \bullet LSl \bullet \overline{B}} \bullet \overline{Left \bullet B} + FS$$

$$Right = (Right + Left \bullet UP) \bullet \overline{Right \bullet LSr \bullet \overline{A}} \bullet \overline{Right \bullet A}$$

$$UP = (UP + Right \bullet LSr \bullet \overline{A} + Left \bullet LSl \bullet \overline{B}) \bullet \overline{UP \bullet Right} \bullet \overline{UP \bullet Left}$$

$$Stop = (Stop + Right \bullet A + Left \bullet B)$$

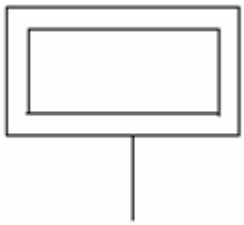
Секвенцијални функциски дијаграми (Sequential function - SFC)

Сите претходни методи се добро прилагодени на процеси кои имаат една активна состојба во текот на времето. Тие се адекватни за едноставни машини и процеси, но за покомплексни машини се дизајнирани за извршување симултани операции.

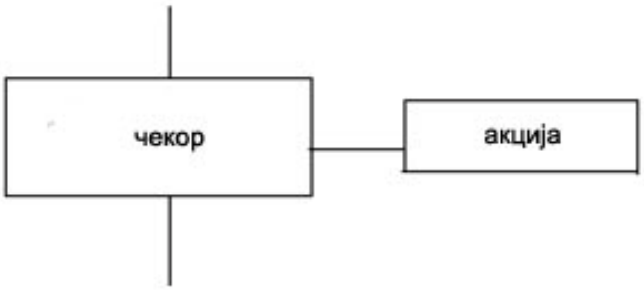
Секвенцијаните функциски дијаграми (SFC) позната уште како *Grafcet* или *IEC 848*, е графичка техника за пишување на конкурентрни контролни програми. SFC се подмножество на покомплексни *Petri net* техники. Елементите на оваа техника се следните:

линии на текот – ги поврзуваат чекорите и транзициите

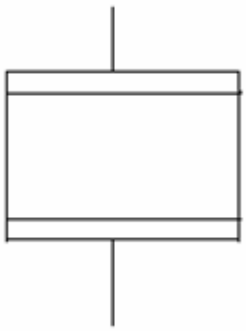
транзиција – предизвикува поместување помеѓу чекорите, се однесува како точка на координација. Дозволува на контролата да премини на следниот чекор кога условите се исполнети (најчесто условна или инструкција на чекање)



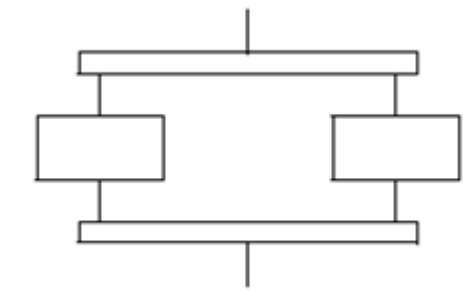
Инцијален чекор – прв чекор

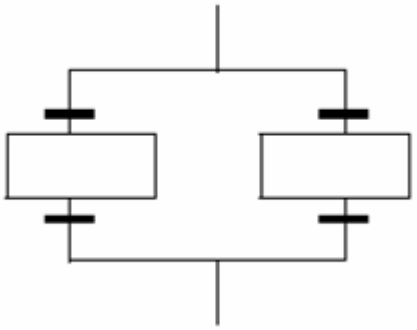


Чекор - основна состојба на операција и најчесто е поврзан со е некоја акција



Макрочекор – колекција на чекори најчесто цела потпрограма



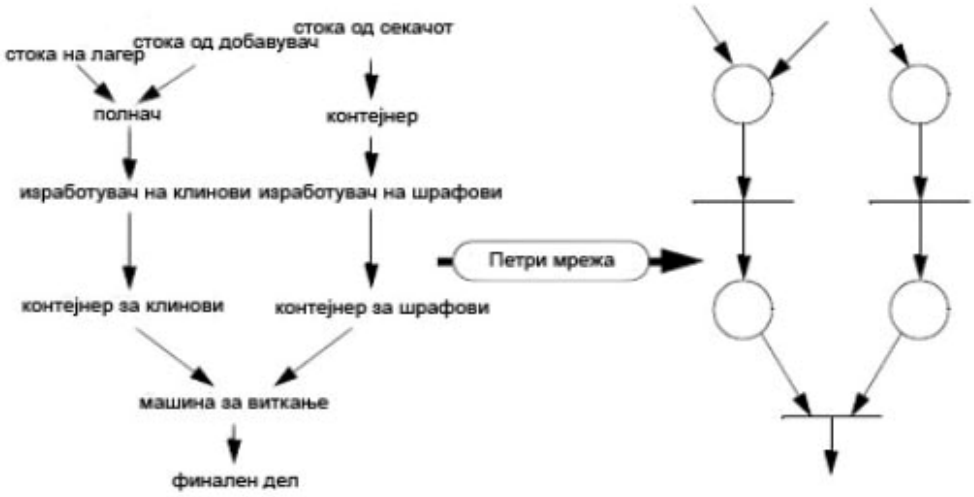


Симултани гранки – движење на токот по двата(или повеќето) патишта

Петри мрежи (PETRI NET)

Постојат 4 основни елементи во Петри мрежи: *место*, *транзиција*, *лак* и *симбол*. Ако се размислува со фабричка услови, *симболите* се еквивалентни на работни делови. *Лаците* се работните патишта низ фабриката. *Местата* се бафери каде се чуваат привремено деловите, и транзиции се еквивалентни на машините каде деловите кои се користат за правење нови делови.

Пример на еден систем опишан со Петри мрежа е даден на слика



Функционално блок програмирање (Function Block Programming)

Структурен Текст е многу силен високо нивовски јазик со корени во *Ada*, *Pascal* and *C*. Ги соодржи основните елементи на модерните програмски јазици, вклучувајки селекциски гранки It contains all the essential elements of a modern programming language, including selection branches (IF-THEN-ELSE и CASE OF) и итерациски јамки (FOR, WHILE и REPEAT). Овие елемент можат и да бидат вгнездени.

Пример во *Структурен Текст*:

```
I:=25;
WHILE J<5 DO
        Z:= F(I+J);
END_WHILE

IF B_1 THEN
        %QW100:= INT_TO_BCD(Display)
ENDIF

CASE    TW OF
        1,5:    TEMP := TEMP_1;
        2:      TEMP := 40;
        4:      TEMP := FTMP(TEMP_2);
ELSE
        TEMP := 0;
        B_ERROR :=1;
END_CASE
```

Контролерот на движење се програмира со помош на свој функционален блок јазик чиј изглед можи да се види во процедурата за проверка на “Emergency stop”

```
PR,,,-W1
BGW
AMW
N=0
#LOOP
PR ,,,WTOT
BGW
AD,,,W1DELAY
CB2
AD,,,W2DELAY
SB2
AMW
PR ,YDELTA,,0
BGY
AMY
N=N+1
JP#ENDLOOP, N=NL
PR ,0,,-WTOT
BGW
AD
,,,W1DELAY+COMP2
CB2
AD,,,W2DELAY
SB2
AMW
PR ,YDELTA,,0
BGY
AMY
N=N+1
JP#LOOP, N<NL
#ENDLOOP
```

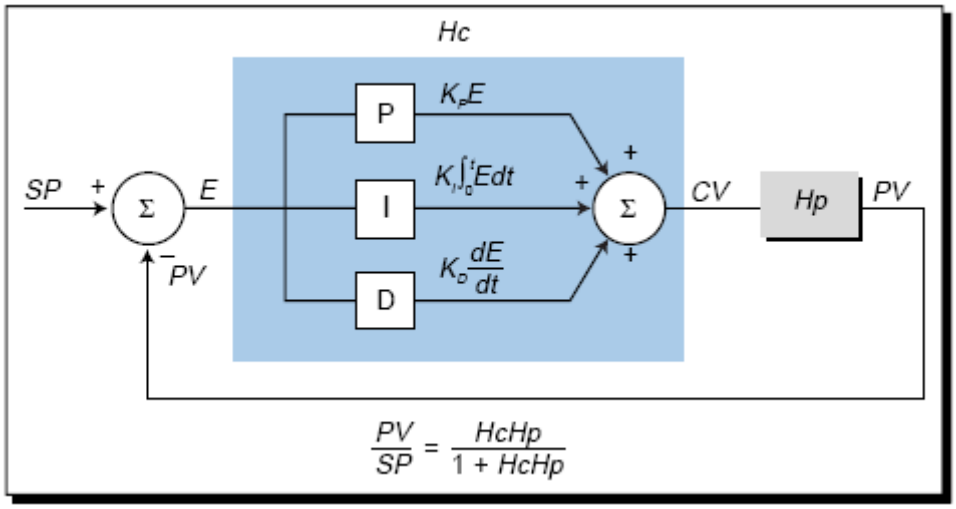
Производувачот обезбедува и интерфејсни библиотеки кои овозможуваат програмирање на контролерот во високо нивовските јазици *C++*, *C#* и *VB*.

```
Option Explicit On

Imports Galil   'Imports gives us access to the Galil Namespace in this module.

Module Module1
```

```
Sub Main()

    Try

        Dim myAPI As New DMCAPI  'Create an API object for communicating with a controller.
        Dim myReg As New DMCGalilRegistry   'Create a Registry object so a controller can be selected.
        Dim sarRspn() As String             'Dimension a string array.

        'Select a controller and open a communication session.
        myAPI.apiOpen(myReg.SelectControllerDlg(IntPtr.Zero), IntPtr.Zero)

        'Write some column headers to the console.
        Console.WriteLine("  X Actual   X Commanded   Y Actual   Y Commanded")
        Console.WriteLine("  Position   Position     Position   Position")

        'Set the X and Y axis speed, set the current positions to zero, and then
        'write the initial axis positions to the console.  Sending four commands
        '("TPX;RPX;TPY;RPY") means the sarRspn string array will have length four
        '(one string response for each command).
        sarRspn = myAPI.apiCmdDiscard("SP500,1000;DP0,0").sarCmdTrim("TPX;RPX;TPY;RPY")    'Note here that two methods are concatenated.
        …
```

2.2 Пропорционален – Интегративен – Диференцијативен Управувач (PID)

Пропорционален –интегративен – диференцијативен управувач ги комбинира сите три мода на работа на управувачот па затоа уште се вика три-моден управувач, кој може да се користи во скоро секој процес кој вклучува лаг и мртво време. Типична блок шема на PID Управувач е преставен на слика:



Функцијата на PID управувачот во временски домен е дадена со р-ката:

$$CV_{(t)} = K_I \int_0^t E dt + K_P E + K_D \frac{dE}{dt} + CV_{(t=0)}$$

каде:

E = грешка

SP =командна вредност

$CV_{(t=0)}$ *=контролна променлива во време t=0 (почетна вредност)*

$CV_{(t)}$ *=контролна променлива во време t*

K_I *=интегрален коефициент на засилување*

K_P *= пропроционален коефициент на засилување*

K_D *= диференцијален коефициент на засилување*

Ако ја поделиме горната равенка со грешката и извршиме Лапласова трансформација преносната функцијата на PID управувачот во s- домен ќе биди:

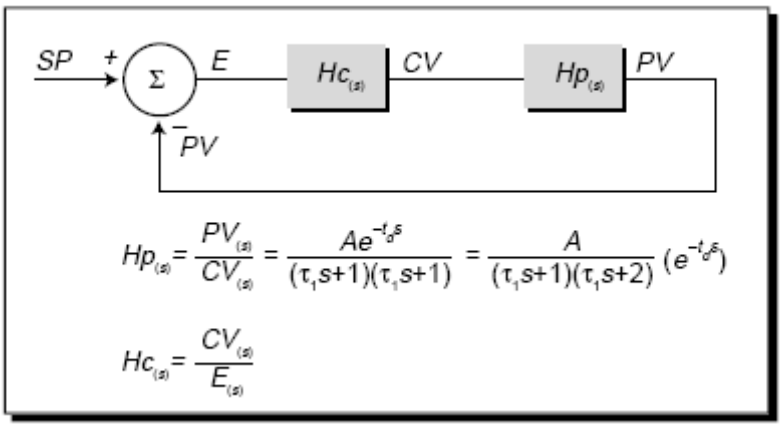
$$Hc_{(s)} = \frac{CV_{(s)}}{E_{(s)}}$$

$$= K_P + \frac{K_I}{s} + K_D s$$

Иако е тешко да се дефинира егзактна преносна функција на процеси од индустријата ($Hp_{(s)}$), можи да се апроксимира со преносна функција со лаг од втор степен и каснење од мртво време:

$$\frac{PV_{(s)}}{CV_{(s)}} = \frac{Ae^{-t_d s}}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

Блок шемата на поставениот систем преку преносните функции:



За да се добие перфектен одзив (излезот потополно да го следи влезот) значи дека Hp*Hc=1 па следи :

$$\begin{aligned} Hc_{(s)} &= \frac{1}{Hp_{(s)}} \\ &= \frac{1}{\left(\frac{A}{(\tau_1 s+1)(\tau_2 s+1)}\right)} \\ &= \left(\frac{1}{A}\right)(\tau_1 s + 1)(\tau_2 s + 1) \\ &= \left(\frac{1}{A}\right)(\tau_1 \tau_2 s^2 + \tau_1 s + \tau_2 s + 1) \\ &= \left(\frac{1}{A}\right)\left[\tau_1 \tau_2 s^2 + (\tau_1 + \tau_2)s + 1\right] \end{aligned}$$

Членот 1/*A* е константа па се преименува во A_1

$$Hc_{(s)} = A_1\left[\tau_1 \tau_2 s^2 + (\tau_1 + \tau_2)s + 1\right]$$

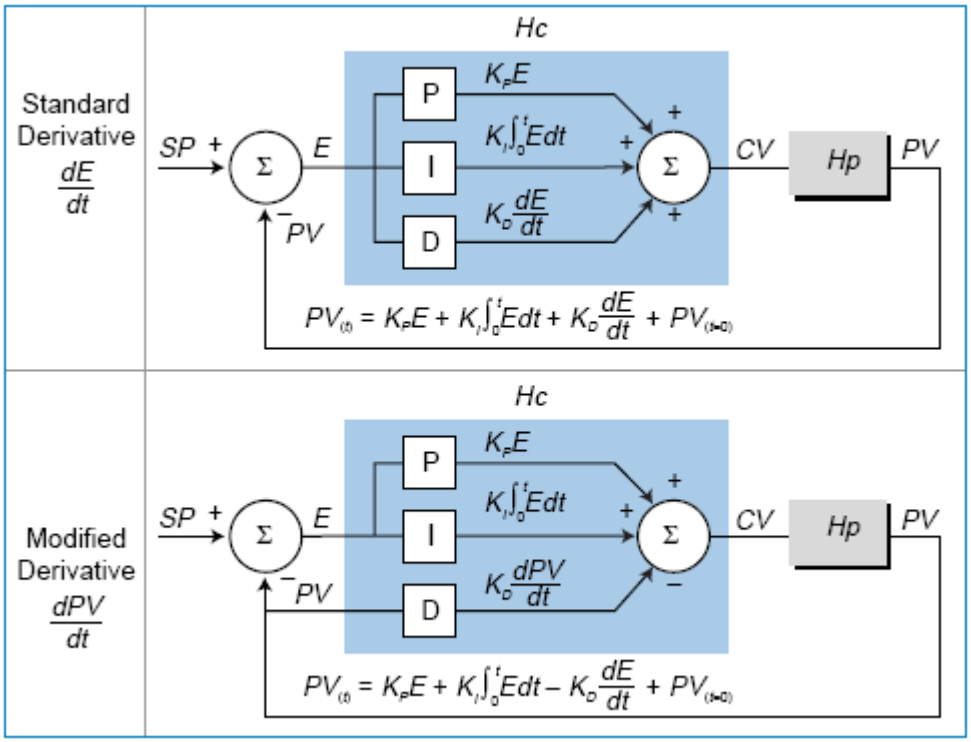
Со делење на секој член со *s* се добива

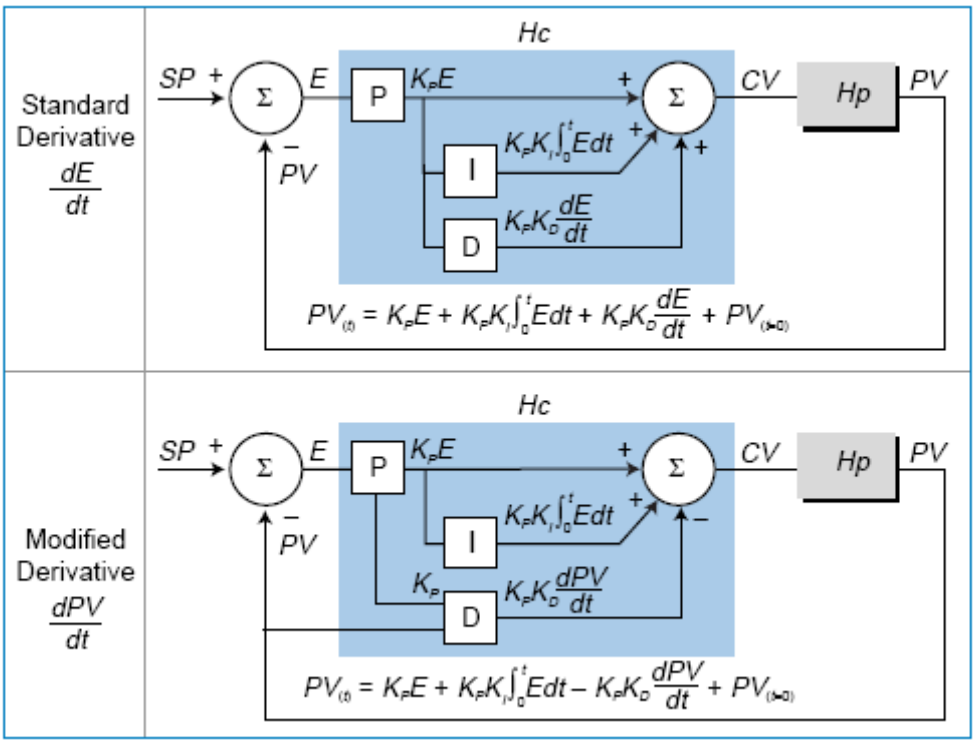
$$Hc_{(s)} = A_1\left[\frac{\tau_1\tau_2 s^2}{s} + \frac{(\tau_1+\tau_2)s}{s} + \frac{1}{s}\right]$$
$$= A_1\left[\tau_1\tau_2 s + (\tau_1+\tau_2) + \frac{1}{s}\right]$$

$$Hc_{(s)} = \underbrace{A_1(\tau_1+\tau_2)}_{\text{P}} + \underbrace{\frac{A_1}{s}}_{\text{I}} + \underbrace{A_1\tau_1\tau_2 s}_{\text{D}}$$

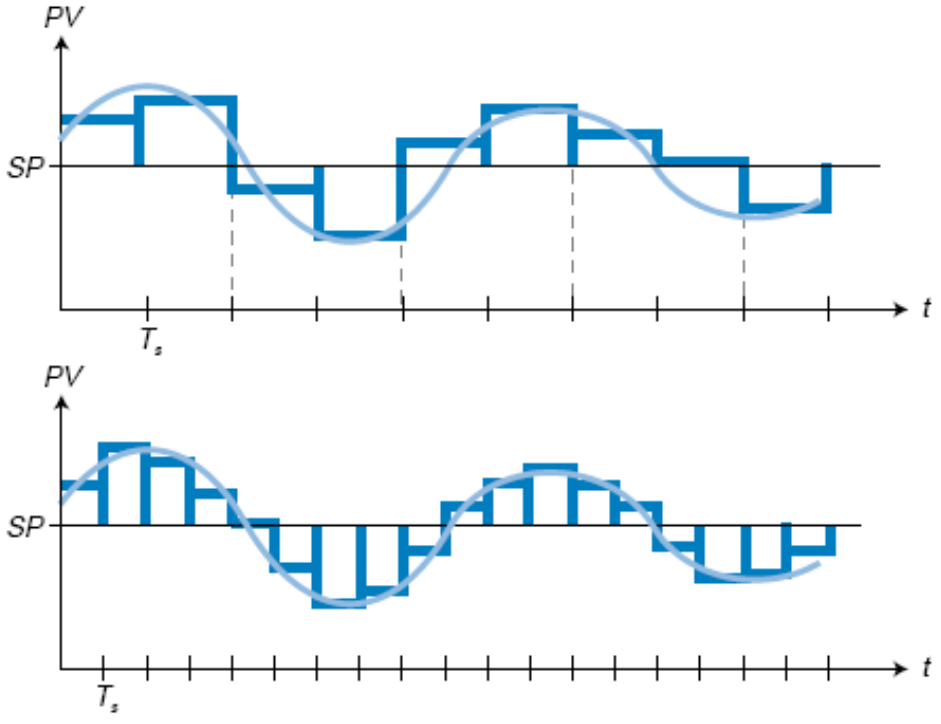
Бидејќи PID управувачот е природна изведба од перфектен систем се смета за универзален тип за контрола на производствените процеси.

PID управувачот може да биде конфигуриран или како сериски или паралелен со стандарден или модификована диференцијативна акција (*dE/dt и dPV/dt респективно*):





PID контроланта единица на PLC ги изведува сите потребни калкулации на итеративна основа, што значи не се континуални, туку се иницирани од временска функција. PLC во точни временски интервали врши семплирање на процесната променлива(PV) и командната вредност(SP) и ги дигитализира и потоа сите пропорцинални, интегративни и диференцијални функции се израчунуваат и сумираат за да ја дадат контролната променлива(CV). Потоа PLC чека на поминување на **семплирачкото време T_s** (од десетици делови од секундата до неколку стотини секунди) до повторно извршување на нова калкулација. Изборот на коректно време на семплирање игра голема важност што можи да се види од сликата



Функцијата на PID управувачот во дигитален домен е дадена со р-ката:

$$CV = k_p\left[(SP - PV) + k_i\sum_0^t (SP - PV)\Delta t + k_d\frac{\Delta(SP - PV)}{\Delta t}\right]$$

Во нумеричкиот дел на интегралот $SP - PV$ е грешка на сигналот, па $\sum_0^t (SP - PV)\Delta t$ е сума на површините (грешката помножена со времето) почнувајќи од уклучувањето на системот до моменталното време t. $\sum_0^t \frac{(SP - PV)}{\Delta t}$ го преставува промента(скокот) на грешката во времето.

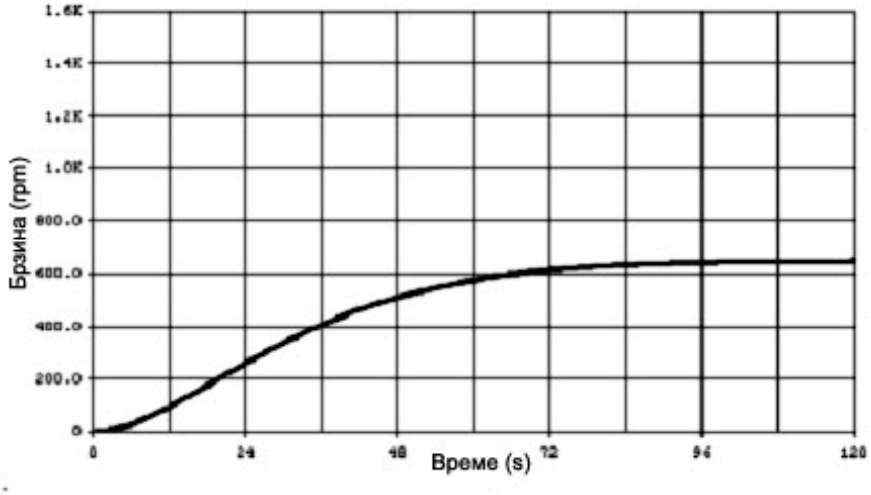
УЛОГА НА PID УПРАВУВАЧИТЕ

Главна улога на PID управувачите при контрола на процесите е решавање на следните проблеми:

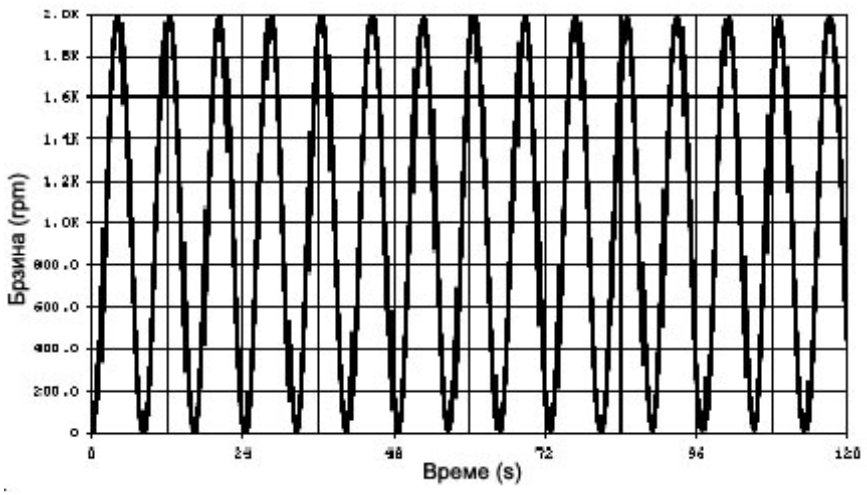
- голем офсет (*разлика меѓу командната вредност и вредноста на излез*)
- спор одзив на промени на командната вредност (*лаг од прв ред*)
- нестабилност (*лаг од втор ред*)

За да се потенцираат проблемите кој ги решава PID управувачот при контрола на актуатори ќе го разгледаме двата графици на одзив на DC моторот од егзампларниот систем командуван да забрза од нула до 1000rpm.

На првиот график на брзината, за мала вредност на пропорционално засилување K_p, јасно се гледа дека системот има спор одзив и му требаат 90 секунди да се стабилизира. Командната вредност од 1000rpm бележи на излез офсет од околу 350rpm под саканата вредност.



На вториот график на брзината, за голема вредност на пропорционално засилување K_p со цел да се намали офсетот, јасно се гледа дека осцилаторна нестбилност со пригушување од 2 минути и опасно високи амплитуди кој можат да стрес на механичкиот систем и оштетување на моторот.



Улога на пропроционалната функција во PID управувачот

Пропорционалната функција во управувачот ја подесува контролната променлива на начин пропорционален на грешката. Во зависност од тоа дали е грешката позитивна или негативна управувачот ке ја зголеми или намали вредноста на контролната променлива доведувајќи ја вредноста на грешката на нула.

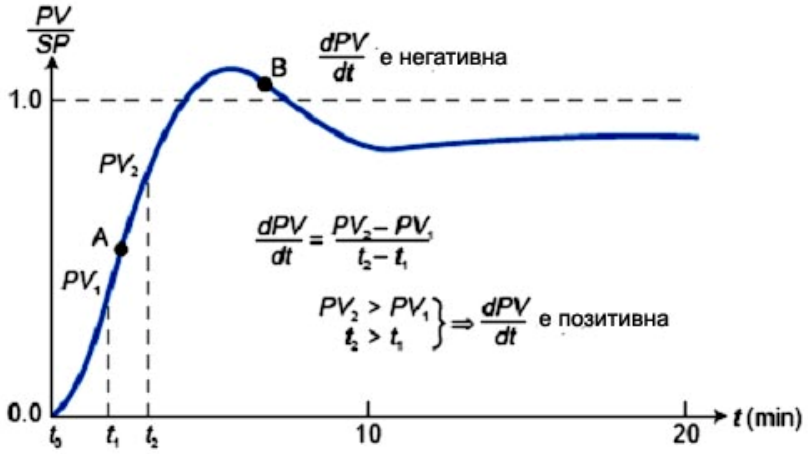
$$CV_{(t)} = K_P E + CV_{(E=0)}$$

каде:
E = грешка
SP =командна вредност
$CV_{(E=0)}$ *=контролна променлива во време t=0 (почетна вредност)*
K_P *= пропроционален коефициент на засилување*

Со правилен избор на коефициентот на пропорционално засилување K_p можи да се подобри брзината на одзивот на системот и да се намали офестот (види слика), но овие проблеми не можат целосно да се отклонат.

Улога на диверенцијалната функција во PID управувачот

Диференцијативната функција е да даде стабилност во повратната врска на системот со намалување на големината на пречекорување и потчекорување на одзивот на системот.



Ако во равенката на контролната променлива на D управувачот се замени изразот за грешката:

$$E = SP - PV$$

$$CV_{(t)} = K_D \frac{dE}{dt} + CV_{(t=0)}$$

$$= K_D\left[\left(\frac{dSP}{dt}\right) - \left(\frac{dPV}{dt}\right)\right] + CV_{(t=0)}$$

$$= \left[K_D\left(\frac{dSP}{dt}\right) - K_D\left(\frac{dPV}{dt}\right)\right] + CV_{(t=0)}$$

се добива изразот

$$CV_{(t)} = -K_D \frac{dPV}{dt} + CV_{(t=0)}$$

каде:
E = грешка
SP =командна вредност
PV =процесна променлива
$CV_{(E=0)}$ *=контролна променлива во време t=0 (почетна вредност)*
K_D *= пропроционален коефициент на засилување*

Од графикот на одзивот на системот на отскочен велез како и од последниот израз јасно се гледа дека

- во случај на позитивна косина (точка А) $\frac{dPv}{dt} > 0$ негативниот предзнак предизвикува намалување на контролната променлива *CV* K_D - пати по промената на процесната променлива во времето и тенденција на спречување на прекорачување

- во случај на негативна косина (точка B) $\frac{dPv}{dt} < 0$ негативниот предзнак предизвикува зголемување на контролната променлива *CV* K_D - пати по промената на процесната променлива во времето и тенденција на спречување на потчекорување.

За да се види функцијата на диференцијаторот во реалниот егзампларен систем се дава команда на моторот од мирување да постигне брзина 1000rpm. Командната променлива SP во t = 0 од 0V е поставен на волтажа во сооднос со брзината која сакаме да ја постигниме кај моторот. Бидејќи моторот не врти излезот од тахо-генераторот е нула следи дека и процесната променлива PV е истотака нула, грешката ке биде голема и биди идентична по вредност со SP, па на излезот од диференцијатор ке се добие голем позитивен сигнал. Па вредноста на контролната променлива $CV(t=0)=K_P SP+K_D\frac{dSP}{dt}(PV=0 \Rightarrow E=SP)$ ке биде голема и ке предизвика моторот да забрзува и тоа за кратко време со оглед на стрмнината на косината $K_D\frac{dSP}{dt}$.

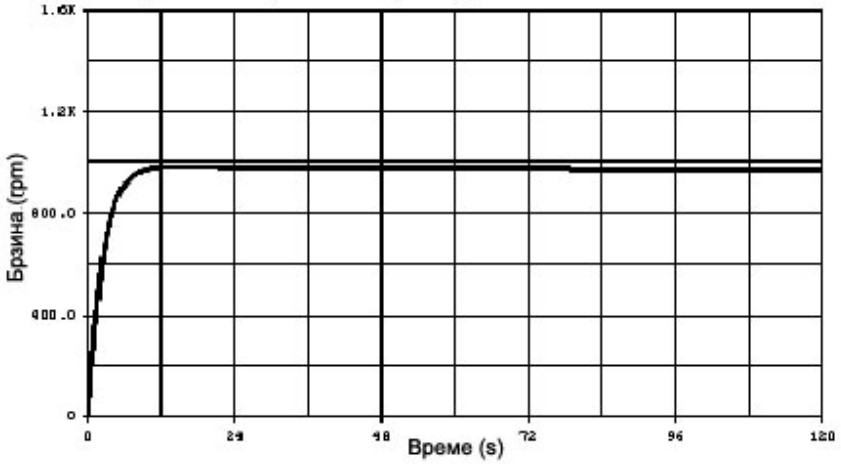
Во време t>0 косината на грешката на напонот ќе биде со спротивен знаг од косината на процесната променлива:

$$\frac{dE}{dt}=\frac{dSP-dPV}{dt}$$
$$SP=const \Rightarrow dSP=0$$
$$\frac{dE}{dt}=-\frac{dPV}{dt}$$

што со други зборови значи дека ако грешката на напонот се зголеми PV ќе се намали и обратно ако грешката се намали PV ќе се зголеми. Брановиот облик на PV е ист со брановиот облик на брзината, па можеме да заклучиме дека косината на грешката е еднаква на негативната косина на брзината. Равенаката на контролната променлива добива облик:

$$CV(t>0)=K_P(SP-speed)-K_D\frac{dspeed}{dt}(PV=speed)$$

од која можи да заклучиме дека CV е намалена K_D - пати по промената на процесната променлива во времето па како моторот забрзува диференцијалниот член ја намалува CV и така покушава да ја намали ратата на забрзување и го спречува прекорачувањето. На ист начин ако моторот успорува, негативната косина на брзината предизвикува диференцијалниот член ја зголемува CV во обид да ја намали ратата на успорување.



Со избор на доволно голем коеф. на диференцијалното засилување K_D и за голема вредност на коеф. пропроционалното засилување K_P се добива брз одзив на промените на влезот и придушување на осцилаторните прекорачување и подкорачувања (намалување на ефектите на лаг од прв и втор ред), но се уште постои офсет.

Улога на интеграторската функција во PID управувачот

Интерграторската функција во управувачот обезбедува излез чија фрекфенција на промена е пропорционална на фрекфенцијата на промена на грешката. Тоа значи колку е поголема грешка толку е побрза измената на излезот на управувачот со цел сузбивање на таа грешка и обратно.

$$\frac{dCV}{dt} = K_I E$$

$$CV_{(t)} = K_I \int_0^t E dt + CV_{(t=0)}$$

каде:

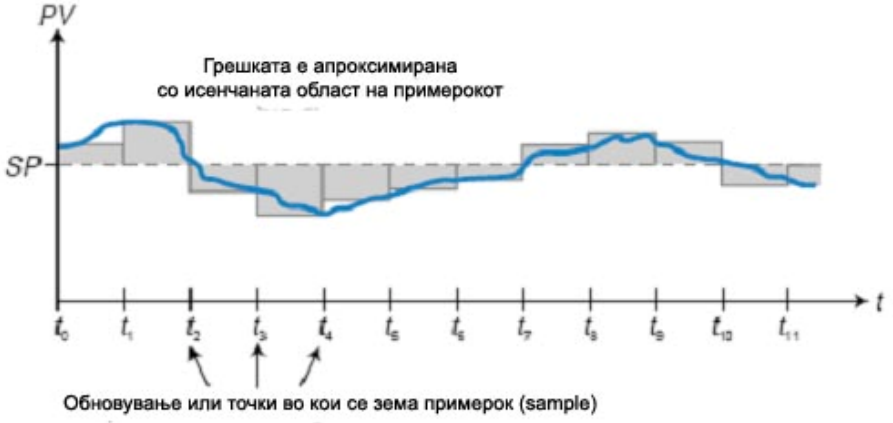
E = грешка

$CV_{(t=0)}$ *=контролна променлива во време t=0 (почетна вредност)*

$CV_{(t)}$ *=контролна променлива во време t*

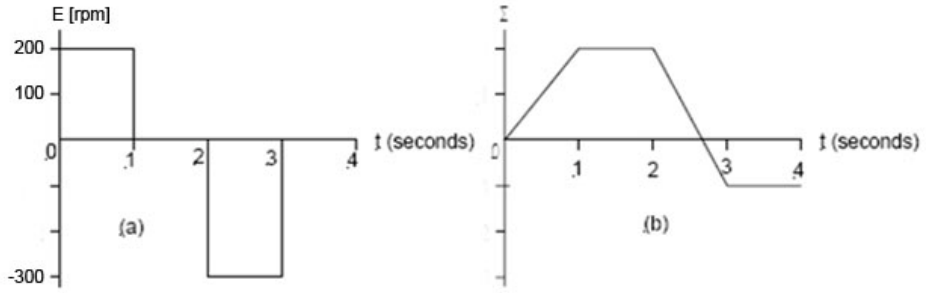
K_I *=интегрален коефициент на засилување*

Интегрирањето на излезот на процеси чии функции се нелинеарни бара тешки математички калкулации и оптоварување на PLC. За намалување на товарот брановидната форма на излезот

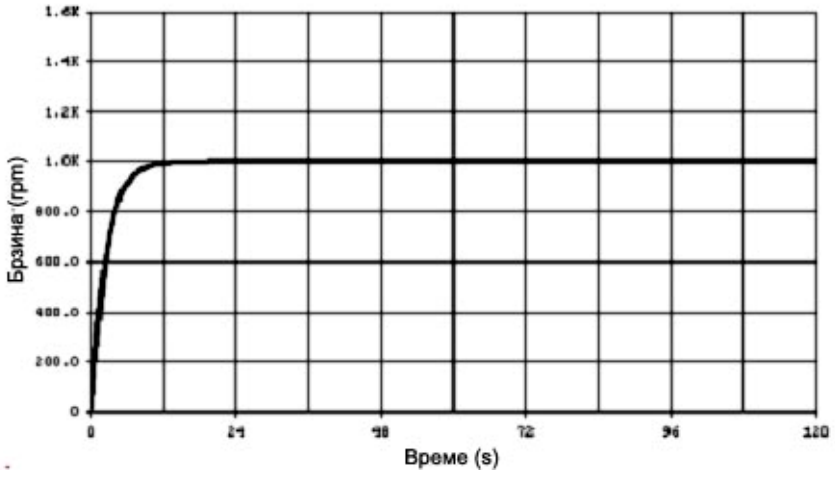


од процесот се дискретизира односно се земаат примероци на одредени еднакви интервали.(види слика)

Интеграторската функција е кумулативна што значи дека врши сумирање на површината на грешката со текот на времето (во t=2 сумата на грешката е 200rpm, во t=3 сумата на грешката е 200-300=-100rpm



За големи офсети, интеграторот ќе акумулира големи вредности и излезот брзо ќе се зголеми. За мали офсети, излезот од интеграторот споро ќе се менува. Како и да е во случај кога офсетот е различен од нула интеграторскот излез ќе се менува во насока да го намали. Со избор на висока вредност на коеф. на пропорцијалното засилување K_p , средна вредност на коеф. на диференцијалното засилување K_D, мала вредност на коефициентот на интегрирање K_I се отстранува офсетот што се гледа од сликата.



ПОДЕСУВАЊЕ

При подесувањето на системот со помош на PID коефициентите можат да се користат математички модели на процесот што во многу случаи е екстремно тешко бидејќи бара одредување на машински параметри (како маса, фрикција, фактор на пригушување, инерција, константа на пружина...) и електрични параметри (како индуктанса, капацитет, отпорност и снага), или подесување со употреба на методите базирани на обиди и грешки.

Математички метод

1. Дефинирање на машинските и електричните параметри на системот
2. Изработка на математичкиот модел со дефинирање на преносните к-ки на сите елементи во директната и повратната врска
3. Со користење на к-ката на колото со отворена повратна врска при услов за потполно следење на влезот од страна на излезот, а со примена на математичкиот апарат се добиваат P,I,D коефициентите.

Практичната примена на математичкот метод ке биде презентиран преку егзампларниот систем кој користи PD управувач за контрола и е прикажан на слика

Параметри на системот:

K_t - торк (0,1 Nm/A)

J - момент на инерција на системот (2x10^{4-} kgm^2)

R - отпор на моторот (2 Ω)

K_a - коефициент на засилувањето (2 A/V)

ω - кружна фрекфенција (500rad/s)

K_f - коефициент на енкодерското засилувањето (318 counts/rev)

Мотор:

$$M(s) = \frac{K_t}{Js^2} = \frac{1000}{s^2}$$

Дигитално-аналоген 16 – битен конвертор :

- разликува 2^{16}=65536 степени
- на секој степен, бидејќи моторот се контролира со напон од +/-10V, отпаѓа:

$$K_{a/d} = \frac{20}{65536} = 0.003\text{V/count}$$

Енкодер:

$K_f = \frac{4N}{2\pi}$ - коефициент на енкодерското засилувањето (636 counts/rad)

N - густина на енкодерските линии(1000 counts/rev)

Преносната к-ка на колото за задршка од 0-ред:

$$H_{ZOH}(s) = \frac{1}{1 + sT/s} = \frac{2000}{s + 2000}$$

каде,

T - периода на земање на примерок (1ms)

К-ка на PD управувачот е дадена со:

$$Hc_{(s)} = P + sD$$

Преносната карактеристика на процесот:

$$Hp_{(s)} = M(s)K_{a/d}K_f H_{ZOH}(s) = \frac{3.17 \bullet 10^6}{s^2(s+2000)}$$

$$Hp_{(j500)} = \frac{3.17 \bullet 10^6}{(j500)^2 \bullet (j500+2000)}$$

$$\left|Hp_{(j500)}\right| = 0.00625$$

$$\text{Arg}[Hp_{(j500)}] = -180° - \tan\text{-}1(500/2000) = -194°$$

Преносната карактеристиката на колото со отворена повратна врска:

$$A(s) = Hc_{(s)} \bullet Hp_{(s)}$$

За да излезот потполно го следи влезот треба се исполнети следните услови:

1. Амплитудата на преносната карактеристика треба да е еднаква на 1

$$\left|A_{(j\omega)}\right| = \left|Hc_{(j\omega)}\right| \bullet \left|Hp_{(j\omega)}\right| = 1$$

значи:

$$|Hc_{(j500)}| = |A_{(j500)}/Hp_{(j500)})| = 160$$

2. Фазната маргина да е 45^o

$$Arg\left[A_{(j\omega)}\right] = Arg\left[Hc_{(j500)}\right] + Arg\left[Hp_{(j500)}\right]$$

$$Arg\left[Hc_{(j500)}\right] = Arg\left[A_{(j500)}\right] - Arg\left[Hp_{(j500)}\right] = -135° + 194° = 59°$$

3. Кружната фрекфенција треба да е поголема од 200rad/s

Со користење на погоре добиените вредности за амплитудата и фазата на функцијата на преносната к-ка на PD управувачот можеме да напишеме,

$$|Hc_{(j500)}| = |P + (j500D)| = 160$$

$$\arg\,[Hc_{(j500)}] = \tan^{-1}[500D/P] = 59°$$

$$P = 160\cos 59° = 82.4$$

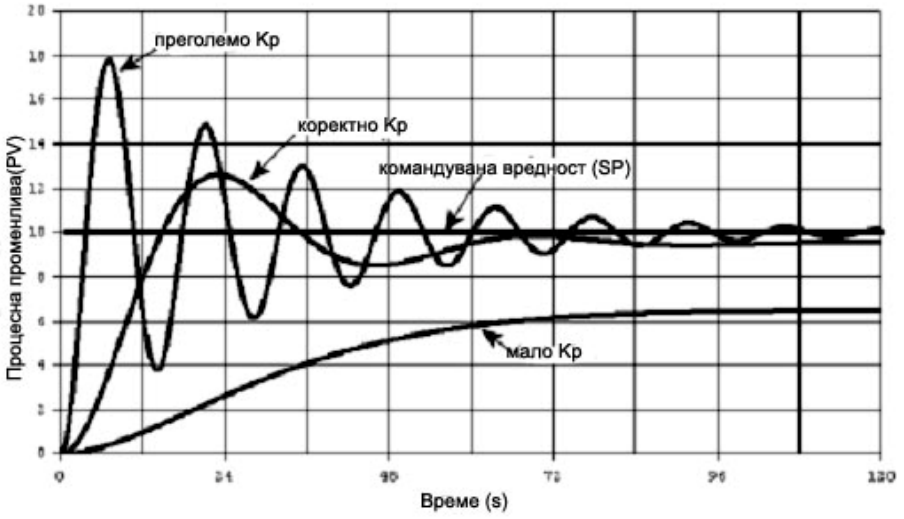
$$500D = 160\sin 59° = 137$$

одкаде со решавање на системот се добиваат пропорцијалната и диференцијалната константа на упраувачот, па неговат к-ка добива облик:

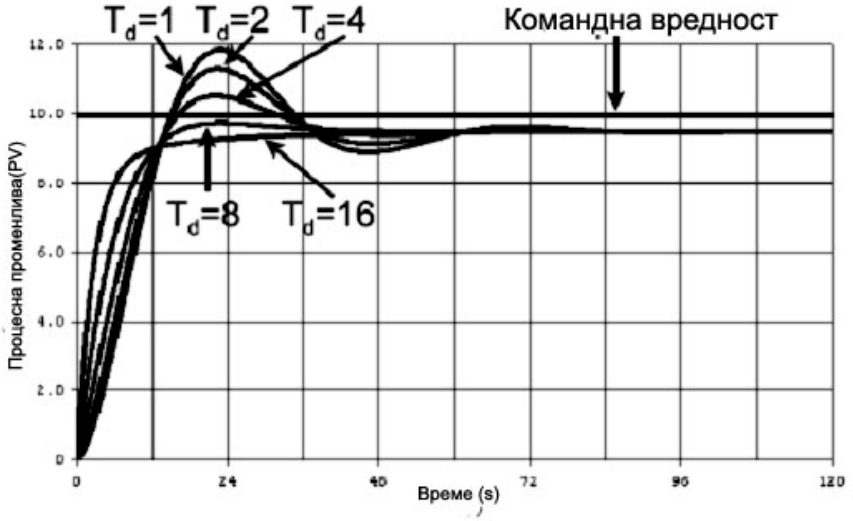
$$Hc_{(j500)} = 82.4 + 0.2744s$$

„Подеси и набљудувај“ метод

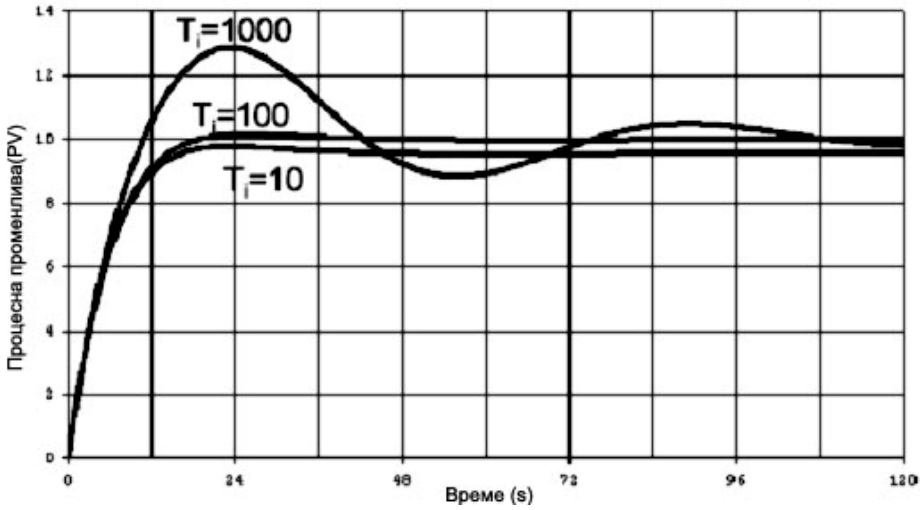
1. Иницијализирај ги PID константите. ($T_D=K_D=0$ и $T_I= 1/ K_I=0$ $K_P=1 – 5$)
2. Во работење на машината брзо измени ја командната вредност. Набљудувај го одзивот. На сликата () е прикажан типичен одзив на отскочна промена на командната вредност од 0 до 10 при различни вредности на K_P. Добра прелиминарна вредност на K_P резултира со одзив со прекорачување од 10%-30% на командната вредност. Во оваа точка на процесот на подесување не водиме сметка за минималните придушени осцилации или офсетот. Тие проблеми ке бидат надминати со подесување на T_D и T_I респективно.



3. Со подесување на T_D ке се елимирнира подчекорувањето и пречекорувањето. Подесувањето започнува со мали вредности па се зголемуваат. Мали вредности ќе предизвикаат мал ефект на кочење во тенденцијата за подчекорувањето и пречекорувањето, додека големи вредности ќе предизвикаат голем офсет.(види слика)



4. Со подесување на Ti ке се елимирнира офсетот. Подесувањето започнува од некоја голема вредност па со намалување со цел да се добие посакуваниот офсет. Ако вредноста на Ti е премногу голема ке предизвика споро елиминирање на офсетот, додека премала вредност премногу брзо елиминирање на офсетот со тенденција да осцилира. На сликата () е прикажан систем со следни вредности за K_P=20, T_D=8 и T_I од 1000, 100 и 10.

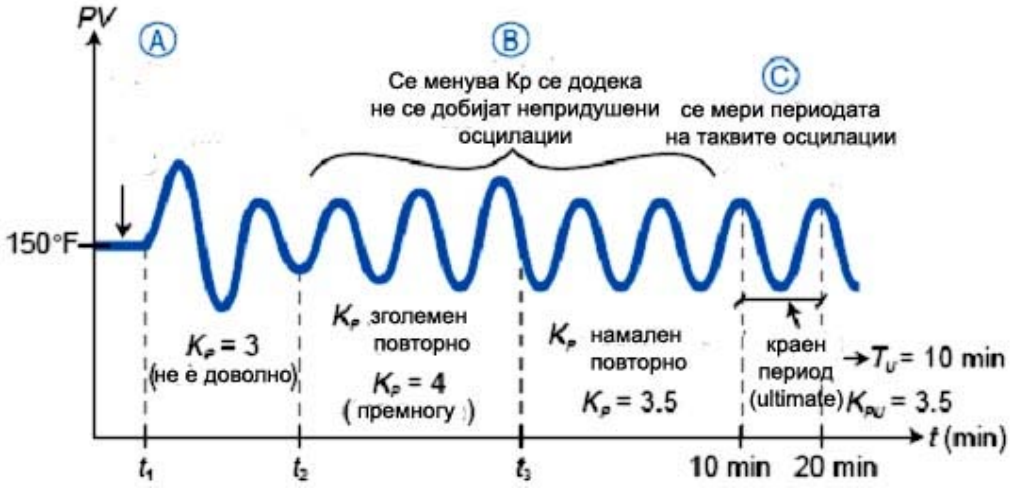


5. Кога подесувањето ке биди завршено, дизајнерот можи да прави понатамошни подесувања на трите константи. Од ова стартна точка дизјанерот можи да пробува со широк опсега на вредности на K_P со цел да постигни побрз одзив. Исто така треба да се тестира стабилноста на ситемот при различни оптертување.

ZIEGLER-NICHOLS метод на подесување на систем со повратна врска

Овој метод се користи за да ги обезбеди константите на K_P, K_I (или T_I), и K_D (или T_D)] во систем со повратна врскain. Методот овозможува подесување на процеси, како серво позиционирачки системи, кое не можат да функционираат во околина со отворена врска.

1. Иницијализирај ги сите PID константи на нула. Уклучете ја машината и системот за контрола на повратната врска.
2. Зголемете го пропорциналниот коефициент на засилување K_P на минимална вредност која ке предизвика системот да почни да осцилира. to oscillate. Потребни се непригушени осцилации т.е амплитудата на осцилациите мора ни да се зголемува ни да се намалува. Можеби ке биди потребно да се направи смена во командната вредност со цел да се индуцираат осцилации.
3. Сними ја вредноста K_P како крајно засилување K_U.

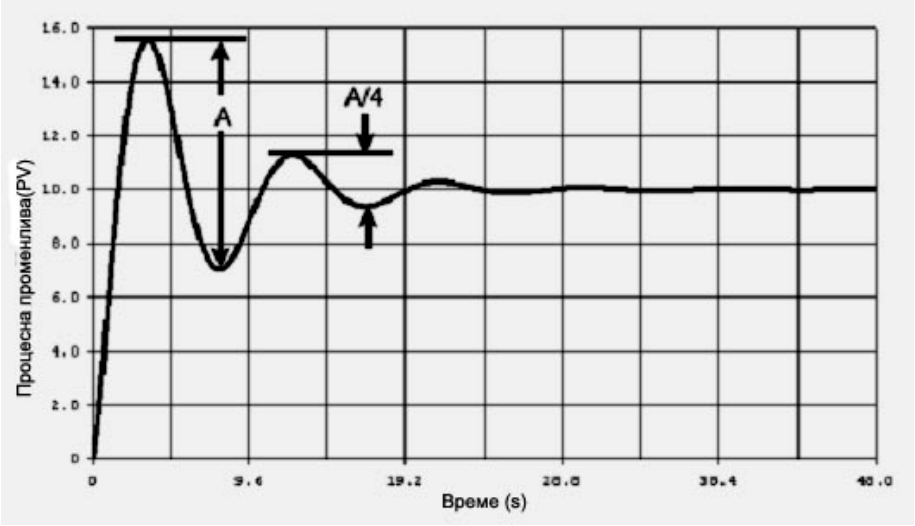


4. Измери го периодот на осцилирачките бранови. Периодот *Tu* е време во секунди за да се изврши еден комплетен циклус.
5. Исклучи го системот и подеси ги вредностите на PID константите според следните формули:

Type of Controller	Loop Tuning Constant	Tuning Equation
Proportional (P)	K_P	$K_P = (0.5)(K_{PU})$
Proportional-Integral (PI)	K_P	$K_P = (0.45)(K_{PU})$
	T_I	$T_I = \frac{T_U}{1.2}$
Proportional-Integral-Derivative (PID)	K_P	$K_P = (0.6)(K_{PU})$
	T_I	$T_I = \frac{T_U}{2}$
	T_D	$T_D = \frac{T_U}{8}$
Note: $\%PB = \frac{1}{K_P}$; $K_I = \frac{1}{T_I}$; $K_D = T_D$		

Модификован ZIEGLER-NICHOLS метод за подесување на систем со повратна врска

Голем проблем во методите кои користат повратна врска е тоа што производните процеси не можат да толерираат осцилации долго, посебно кога се потребни многу обиди за нивно добивање. Целта на овој метод е да се добие пречекорување со амплитуда, од дно до врв, која во секој цикус ќе биде ¼ помала од претходната па затоа овај метод се вика уште метод на четвртинско пригушување на брановите.



Еднаш кога ке се добие таков одзив се мери периодата на четвртинско пригушување $T_{1/4}$ и константата на засилување $K_{1/4}$ која го предизвикува таквиот одзив како што е прикажано на слика

Константите $K_{1/4}$ and $T_{1/4}$ се заменуваат во формулите на стандардниот Ziegler-Nichols метод за подесување на следниот начин:

$$K_{PU} = 2K_{1/4}$$
$$T_U = T_{1/4}$$

$$\begin{aligned} K_P &= 0.5K_{PU} \\ &= 0.5\left(2K_{1/4}\right) \\ &= K_{1/4} \end{aligned}$$

3. АКТУАТОРИ И СЕНЗОРИ

Сензори

Сензорите му дозволуваат на управувачот да ја детектира ситуацијата на процесот и да изврши одредена функција во зависност од тоа. Типчни физички феномени кои ги детектираат сензорите се:

- *Индуктивна близина* – (дали метал е во близина?)
- *Капацитивна близина* – (дали е диелектрик во близина?)
- *Оптичка присуство* (дали објектот ја прекинува светлината или ја рефелктира?)
- *Мехнички контакт* (дали објектот допира прекинувач?)
- *Хемиско присуство* (дали има присуство на специфична хемикалија или класа на хемикалии)
- *Топлота* (дали има покачување на температурата?)
- *Електромагнетно присуство* (дали постои некој вид на електромагнетно зрачење?)

Актуатори

Актуатор е механички уред за движење или контролирање на механизам или систем, најчесто преку претворање на електричната енергија во механичко движење.

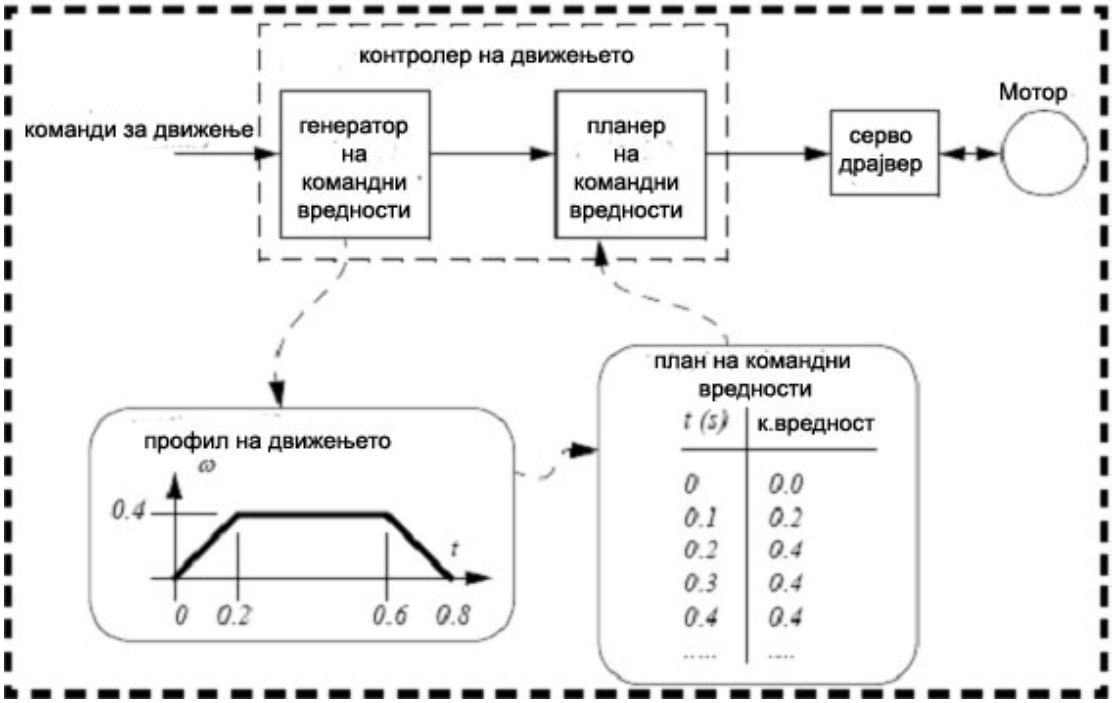
- *Соленоди* – ја претвораат електричната струја во ограничено линерано движење
- *Хидраулики и Пнеуматици* – користат цилиндри за претворање на течностите и гасовите во ограничени линерани движења.
- *Мотори* – се користат кога е потребно циклично движење, но можно нивното цикличното движење да биде претворено во линиско движење

Контролата на најразличните видови сензори и актуатори има свој специфичности и бара многу пообен простор за излагање. Со цел да се види врската меѓу контролата и контролираното, а со оглед на спецификациите на егзампларниот систем ке се насочиме во разгледување на контрола на актуатори т.е контрола на движење.

КОНТРОЛА НА ДВИЖЕЊЕ (Motion control)

Денешниот свет е свет на движење. Па една од најважните задачи на управувачите е контрола на движењето на разни компоненти во производните автоматизациони процеси. Контрола на движењето можи да се подели на неколку конститутивни сегменти (сл.):

- Задавање на команди за движење
- Генерирање на план на командни вредности во зависност од профилот на движењето
- Егзекуција на планот од страна на планерот на движењето



Команди за движење

Командите на движењето ги задава операторот или софтверската апликација и ги соодржат следните вредности:

Θ_{start} – *моменталниот агол на мотороt*

Θ_{end} – *аголот за кои треба да се заврти моторот*

ω_{max} – *брзина на движење*

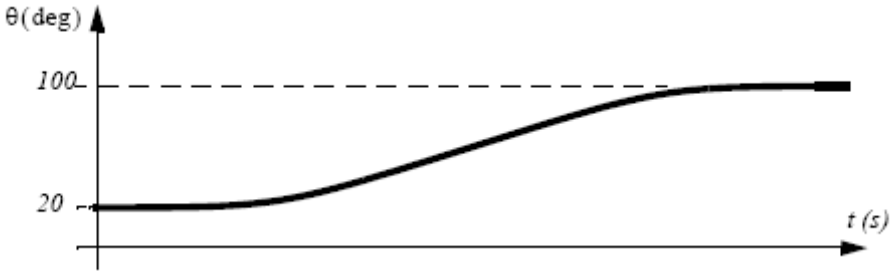
a_{max} – *забрзување*

Најчесто аголот, брзината и забрзувањето се дадени во сооднос со енкодерски единици (*counts*, *counts/s*, *counts/s*^{2} респективно)

Генерирање на план на командни вредности

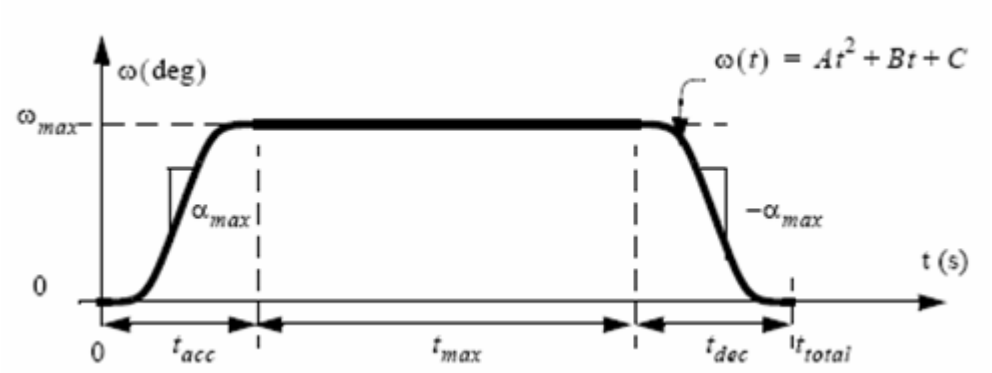
При контрола на позицијата на движењето се настојува да се добие брзина 0 на почеток и на крајот на движењето, и меко забрзување и успорување. Тоа се постигнува со избор на соодветен профил на брзината.

На слика е прикажан профил на брзината за движење со старт во 20 степени и крај од 100 степени.



За намалување на грчот се користат најразлични форми на профилот (рампа).

Егзампларниот систем користи квадратна парабола, но во употреба се разни форми на z –криви и s – криви за дефинрање на профилот на брзината со меко убрзување и успорување и тоа за најкратко време.



каде:

ω_{max} – *максимална брзина*
α_{max} – *максимална брзина*
t_{acc} – *време на забрзување*
t_{dcc} – *време на успорување*
t_{max} – *време на при максимална брзина*
t_{total} – *вкупно време на движењето*

На база на влезните вредности треба да се провери дали е можно креирање на рампа. Тргнуваме од граничните вредности на полиномната функција:

$$\omega(0) = 0 \qquad \omega\left(\frac{t_{acc}}{2}\right) = \frac{\omega_{max}}{2}$$

$$\frac{d}{dt}\omega(0) = 0 \qquad \frac{d}{dt}\omega\left(\frac{t_{acc}}{2}\right) = \alpha_{max}$$

Ова можи да се искористи да се добијат вредностите на коефициентите:

$$\omega(0) = A^2 \cdot 0 + B \cdot 0 = 0$$
$$\omega(0)' = 2A \cdot 0 + B = 0$$
$$=> B = 0$$
$$=> C = 0$$
$$\omega(t) = At$$
$$\omega(t_{acc}) = \omega_{\max} = At_{acc}^{\ 2}$$

$$\omega_{max} = At_{acc}^{\ 2} \qquad A = \frac{\omega_{max}}{t_{acc}^{\ 2}}$$

$$\alpha_{max} = 2At_{acc} \qquad A = \frac{\alpha_{max}}{2t_{acc}}$$

$$A = \frac{\omega_{max}}{t_{acc}^{\ 2}} = \frac{\alpha_{max}}{2t_{acc}} \qquad t_{acc} = \frac{2\omega_{max}}{\alpha_{max}}$$

$$A = \frac{\alpha_{max}}{2t_{acc}} = \frac{\alpha_{max}}{2\left(\frac{2\omega_{max}}{\alpha_{max}}\right)} = \frac{\alpha_{max}^2}{4\omega_{max}}$$

Равенките на за брзината во првиот сегмент се:

$$\omega(t) = \frac{\alpha_{max}^2}{4\omega_{max}} t^2 \qquad 0 \le t < \frac{t_{acc}}{2}$$

Додека равенките на за брзината во вториот сегмент се:

$$\omega(t) = \omega_{max} - \frac{\alpha_{max}^2}{4\omega_{max}}(t_{acc} - t)^2$$

$$\omega(t) = \omega_{max} - \frac{\alpha_{max}^2}{4\omega_{max}}(t^2 - 2t_{acc}t + t_{acc}^2) \qquad \frac{t_{acc}}{2} \le t < t_{acc}$$

Аголот Θ_{acc} за кој ќе се заврти моторот за време на забрзувањето односно успорувањето може да се добие со интегралење на двата сегменити (како делови на парабола) до $t_{acc}/2$ и од $t_{acc}/2$ до t_{acc}:

$$\theta_{acc} = \int_0^{\frac{t_{acc}}{2}} \frac{\alpha_{max}^2}{4\omega_{max}} t^2 dt + \int_{\frac{t_{acc}}{2}}^{t_{acc}} \left(\omega_{max} - \frac{\alpha_{max}^2}{4\omega_{max}} (t^2 - 2t_{acc}t + t_{acc}^2) \right) dt$$

$$\theta_{acc} = \frac{\alpha_{max}^2}{12\omega_{max}} t^3 \Big|_0^{\frac{t_{acc}}{2}} + \left(\omega_{max} t - \frac{\alpha_{max}^2}{4\omega_{max}} \left(\frac{t^3}{3} - t_{acc}t^2 + t_{acc}^2 t \right) \right) \Big|_{\frac{t_{acc}}{2}}^{t_{acc}}$$

$$acc = \frac{\alpha_{max}^2}{12\omega_{max}} \frac{t_{acc}^3}{8} + \omega_{max} t_{acc} - \frac{\alpha_{max}^2}{4\omega_{max}} \left(\frac{t_{acc}^3}{3} - t_{acc}^3 + t_{acc}^3 \right) - \omega_{max} \frac{t_{acc}}{2} + \frac{\alpha_{max}^2}{4\omega_{max}} \left(\frac{t_{acc}^3}{24} - \frac{t_{acc}^3}{4} + \frac{t_{acc}^3}{2} \right)$$

$$\theta_{acc} = \frac{\alpha_{max}^2}{96\omega_{max}} t_{acc}^3 + \frac{\omega_{max} t_{acc}}{2} - \frac{\alpha_{max}^2}{96\omega_{max}} t_{acc}^3$$

$$\theta_{acc} = \frac{\omega_{max} t_{acc}}{2}$$

Аголот за кој треба да се заврти моторот е разлика од моменталниот агол и ново зададениот агол:

$$\Delta\theta = \theta_{start} - \theta_{end}$$
$$\Delta\theta = \theta_{acc} + \theta_{\max} + \theta_{dcc} (\theta_{acc} = \theta_{dcc})$$
$$\Delta\theta = \theta = 2 \cdot \theta_{acc} + \omega_{\max} t_{\max}$$

Од равенката за промената на аголот Θ следи:

$$t_{max} = \frac{(\theta - 2\theta_{acc})}{\omega_{max}}$$

$$t_{max} = \frac{|\theta|}{\omega_{max}} - \frac{|t_{acc}|}{2} - \frac{|t_{dec}|}{2}$$

Од претходната равенка се гледа дека ако t_{max} е негативно моторот нема да ја достигне максималната брзина и профилот на брзината ќе ја изгуби трапезоидалната форма и ќе стане триаголник.

Времињата на забрзување и успорување t_{acc} и t_{dcc} $(t_{acc} = t_{dcc})$ се

$$\frac{\theta}{2} = \frac{1}{2} \alpha_{max} t_{acc}^2$$

$$t_{acc} = \sqrt{\frac{\theta}{\alpha_{max}}}$$

Движењето од почетната точка до крајната точка можи да се опиши со траекторија на позициони точки односно командувани аголни вредности. За добивање на истите потребно е поставување на равенките за моменталниот агол $\theta(t) = \omega(t) * t$:

$$\theta(t) = At^3 + Bt^2 + Ct + D$$

$$\frac{d}{dt}\theta(t) = 3At^2 + 2Bt + C$$

$$\theta(t=0) = \theta_{start}$$

$$\frac{d}{dt}\theta(t=0) = 0$$

Со решавање на системот на равенки за граничните вредносити (t=0) се добиваат коефициентите *A, B, C, D*

$$0 = A0^3 + B0^2 + C0 + D \qquad \therefore D = 0$$

$$0 = 3A0^2 + 2B0 + C \qquad \therefore C = 0$$

$$0 = 3A1^2 + 2B1 \qquad \therefore B = \left(-\frac{3}{2}\right)A$$

$$1 = A1^3 + B1^2 + (0)0 + 0 \qquad \therefore A = -2$$

$$\therefore B = 3$$

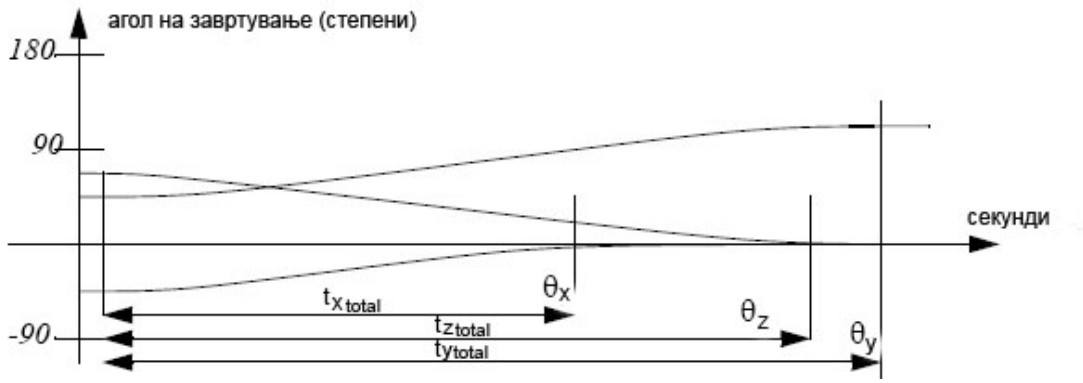
Генераторот користејќи ја формулита за $\Theta(t)$ ја пополнува табелата на планирани командни вредности(позициони точки) за секој временски циклус (*clock*) (пример за секоја секунда на часовникот во табелата постои чекорен агол од почетниот 75 ^{o} до -35 ^{o}, види табела)

t (sec)	theta(t)
0	75
1	71.92
2	63.56
3	51.24
4	36.28
5	20
6	3.72
7	-11.24
8	-23.56
9	-31.92
10	-35

Интерполација на движењето

Најчесто контролираните системи се составени од повеќе мотори кои овозможуваат движење во различни рамнини *x, y, z,* .. респективно. Затоа потребно е ускладување на нивните меѓусебни движења во времето па се врши интерполација на движењето. Интеполацијата се состои во тоа да се прилагодат брзините во однос на најспорото движење со цел движењата по сите оски да завршат истовремено.

Со користење на формулата () за секоја оска посебно се пресметуваат времињата на завртување. Ако времето на завртување t_{ymax} по y-оската, како што се гледа од слика(), е најголемо, потребна е рекалкулација на брзините ωx_{max} и ωz_{max}



$$\omega_{\max} t_{\max} = \Delta\theta - 2\theta_{acc}$$
$$t_{total} = 2t_{acc} + t_{\max}$$
$$\Rightarrow \omega_{\max}(t_{total} - 2t_{acc}) = \Delta\theta - \frac{\omega_{\max} t_{acc}}{2}$$
$$\omega_{\max} t_{total} - \omega_{\max} t_{acc} - \Delta\theta = 0$$

Со замена на вредноста на t_{acc} ($t_{acc} = \frac{2\omega_{\max}}{\alpha_{\max}}$)се добива квадратна равенка:

$$-2\frac{\omega_{\max}^{2}}{\alpha_{\max}} + \omega_{\max} t_{total}\alpha_{\max} - \Delta\theta\alpha_{\max} = 0$$

Со чие решавање се добива изразот за рекалкулација на брзините ωx_{max} и ωz_{max}

$$\omega_{x,z\max} = \frac{-t_{ytotal}\alpha_{x,z\max} \pm \sqrt{(t_{ytotal}\alpha_{x,z\max})^2 - 8\Delta\theta}}{-4}$$

Егзекуција на планот

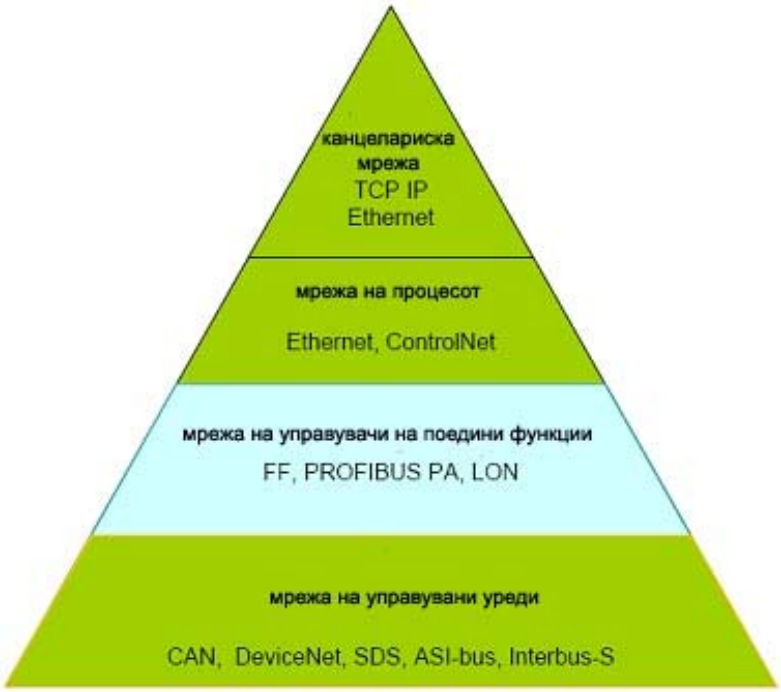
Кога планот на движењето ке биде направен планерот на движење го извршува планот со обновување на командната вредност со секој прекин(*interrupt*) иницран од завршувањето на временскиот циклус (clock). Командата вредност која преставува агол на завртување се испраќа до драјверот на моторот кој ја претвора во соодветен актуаторски сигнал (најчесто напон). Рутината се повторува за секоја оска (X,YZ,…) посебно.

4.КОМУНИКАЦИЈА

Начинот на поврзување на интелегентните компонетни во хиерархиските системи во голема мера го одредуваат квалитетот и функционалноста на целиот систем. Изборот на одговарачки интерфејс и комуникационен протокол зависи од многу фактори:

- одберената топологија на мрежата (прстен, магистрала, ѕвезда...)
- максималната брзина на пренос
- најдолго време на чекање при воспоставување на врската
- максимален број на чворови
- начин на приклучување на чворовите
- број и тип на потребните проводници за реализација на мрежата
- цена и сложеност
- отпорност на пречки
- тип на детекција на грешки при преносот и сигурност на преносот во целина
- лесно детектирање и отклонување на грешките и проблемите при комуникацијата (анализатори на протоколот)

Употребата на одредени комуникациски протоколи во зависност од нивота на контрола и управување е дадена на сл()



Споредба на карактериситките на различните мрежи е дадена во табела:

Network	topology	addresses	length	speed	packet size
Bluetooth	wireless	8	10	64Kbps	continuous
CANopen	bus	127	25m-1000m	1Mbps-10Kbps	8 bytes
ControlNet	bus or star	99	250m-1000m wire, 3-30km fiber	5Mbps	0-510 bytes
Devicenet	bus	64	500m	125-500Kbps	8 bytes
Ethernet	bus, star	1024	85m coax, 100m twisted pair, 400m-50km fiber	10-1000Gbps	46-1500bytes
Foundation Fieldbus	star	unlimited	100m twisted pair, 2km fiber	100Mbps	<=1500 bytes
Interbus	bus	512	12.8km with 400m segments	500-2000 Kbps	0-246 bytes
Lonworks	bus, ring, star	32,000	<=2km	78Kbps-1.25Mbps	228 bytes
Modbus	bus, star	250	350m	300bps-38.4Kbps	0-254 bytes
Profibus	bus, star, ring	126	100-1900m	9.6Kbps-12Mbps	0-244bytes
Sercos	rings	254	800m	2-16Mbps	32bits
USB	star	127	5m	>100Mbps	1-1000bytes

Во понатамошното излагање ќе бидат разгледани следните комуникациони протоколи:

- *DeviceNet (CAN Bus)* - ниво на управувани уреди,
- *Controlnet* – на ниво на управување со поединечните функции, уреди и регулациони уреди
- *Ethernet* - на нивото на управување на процеси

ниво

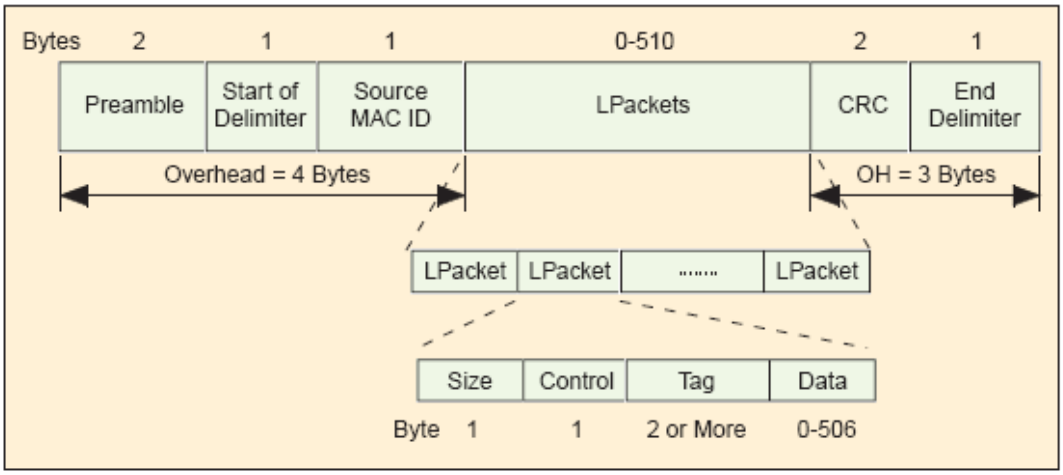
- *SERCOS* – дигитални протоколи за комуникација во реално време
- *Asynchronous u Synchronous CDMA* – бежични протоколи

ControlNet

Стандардоте е дизајниран за комуникција меѓу контролерите и дозволува покомплексни пораки од *Devicenet*. Овие мрежи се детерминистички и бидејќи максималното време на чекање пред да се испрати рамката со пораката е одредено со времето на ротација на симболот т.н мрежен интервал на обнова *network update interval* - (NUT). Најбрзиот NUT што може да се специфицира е 2 ms.

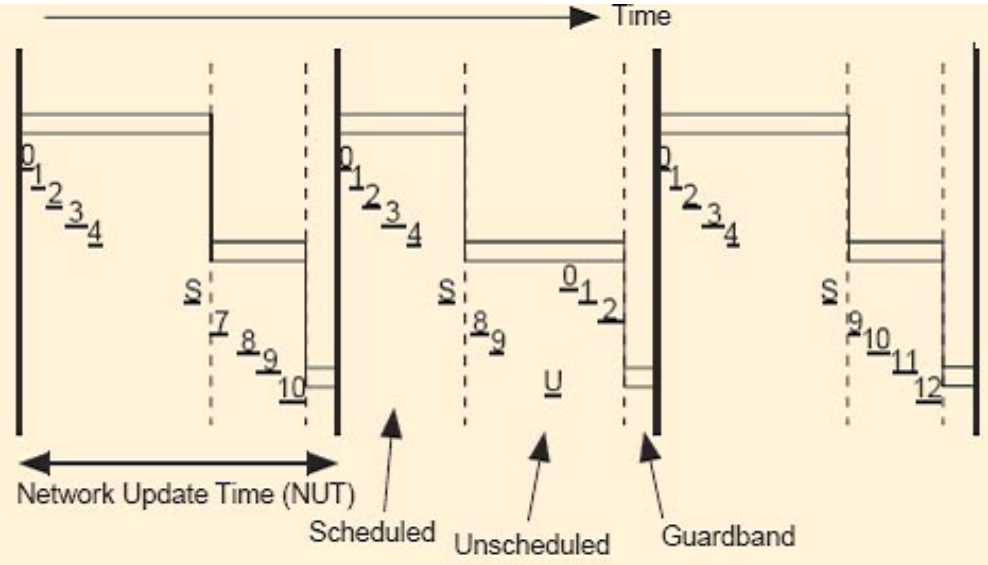
Овој протокол дозволува линерна, во вид на дрво или сегментирана топологија. ControlNet –товата метода за контрола врз медиумот (MAC) ги искористува сите предности на Producer/Consumer модел и дозволува контрола на повеќе контролери да го контролираат излезот/влезот на иста жица. Ова обезбедува значителна предност над другите мрежи, кои дозволуваат само еден главен (*master*) контролер на жицата.

Чворовите во прстенот се поставени логички во вид на прстен и во случај на ControlNet, секој чвор ја знае адресата на својот претходник и следбеник. За време на операцијата на мрежата, чворот со симбол испраќа рамки со податоци се додека или нема повеќе или времето кое го држи симболот го достигне својот лимит. Чворот потоа регенерира симбол и го испраќа на својот логички следбеник на мрежата. Ако чворот нема пораки за праќање само го пропушта симболот на следниот чвор. Физичката локација на следниот не е важна бидејќи симболот се праќа на логичкиот сосед.
Рамките не се судират бидејќи само еден чвор може да испраќа. Протоколот го гарантира максималното време меѓу пристапите на мрежата за секој чвор и обезбедува регенерирање на симболот ако чворот што го чува симболот престане да испраќа и не го пушта симболот на својот следбеник. Чворвите истотака можат да се додаваат динамички на магистралата и да побараат да бидат отпуштени од логичкиот престен.
Формата на рамката на пораката на ControlNet е прикажана на сл(). Вкупното заглавие 7 бајти вклучувајќи преамбула, стартен одвојоувач, (CRC), и краен одвојоувач. Рамката со пакети со податоци уште наречен линк пакет (*Lpacket*) рамка, можи да содржи повеќе пакети од кој секој од нив соодржи величина, контолен бит,ознака и поле за податоците со големина меѓу 0 и 510 бајти. Индивидуалната дестинациона адреса е специфицирана во полето со ознака. Полето за величината специфицира број на бајт парови од (3 до 255) соодржани во индивидуалните Lpacket, вклучувај го полето за величина, контрола, ознака и линк полето за податоци.



ControlNet протоколот има прифатено имплицитен механизам за проследување на симболот и доделува уникатни MAC ID (од 1 до 99) на секој чвор. Всушност нема вистински симбол кој се проследува низ мрежата. Наместо тоа секој чвор го следи *MAC ID*-то на изворот на секоја примена порака. На крајот рамката на пораката, секој чвор сетира имплицитен симбол регистер на вредност на вредноста на MAC ID на изворот на пораката зголемена за еден(MAC ID + 1). Ако имплицитниот регистер еднаков е на MAC ID на самиот чвор, тој чвор го има симболот значи може да испараќа пораки. Сите чворови имаат иста вредност во нивните имплицитни симбол регистри со што ја спречуваат колизијата во медиумот. Ако чворот нема податоци за испраќање, испраќа порака со празно *Lpacket* поле, наречено нулта рамка.

Мрежниот пристап е контролиран од алгоритам заснован на временска поделба наречен "Конкурентен повекекратен пристап во временскиот домен (*Concurrent Time Domain Multiple Access* - CTDMA), кој ги регулира можностите на чворовите да испраќаат секој мрежен интервал.
Должината на циклусот, наречен мрежно време на обнова (*network update time*
-NUT) во ControlNet или време на ротација на симболот (token rotation time - TRT) генерално, е поделен на три главни делови: планиран, непланиран и заштитен како што е прикажан на слика().



За време на планираниот дел на NUT-тот, секој чвор може да испарќа временски-критични/планирани податоци со обезбедување на имплицитен симбол од 0 до *S*. За време на непланираниот дел на NUT-тот, чворовите од 0 до *U* ја делат можноста да испраќаат временски не критични податоци во *round-robin* стил се додека алоцираното време за непланираниот дел помини. Кога ќе се достигни до времето на заштитниот дел сите чворови престануваат со испраќање и само чворот со најнизок MACID, наречен модератор може да испраќа пораки за одржување, наречени модераторски рамки, кои извршуваат синхронизација на сите тајмери во секој чвор и објавуваат критичните линк параметри како NUT, времето на чворот, *S*, *U*, и итн. Ако модераторската рамка не е слушната во две последователни NUT-а, чворот со најнизок MAC ID ќе почни да испраќа модераторска рамка во заштитниот дел на третиот NUT. Ако модераторот забележи дека некој друг чвор има понизок MAC ID од неговиот, тој веднаш ја откажува модераторската улога.

Предности

Token bus (магистрала на симболи) протоколот е детерминистички и обезбедува одличен капацитет и ефикасност при високи оптоварувања на мрежата. За време на мрежната операција, можат динамички да се додаваат или одземаат чворови од мрежата што е контраст на *token ring* (прстени од симболи). Планираните и не планираните сегменти во секој NUT циклус го прават ControlNet-тот погоден за двете и временски критични и обични пораки.

Недостатоци

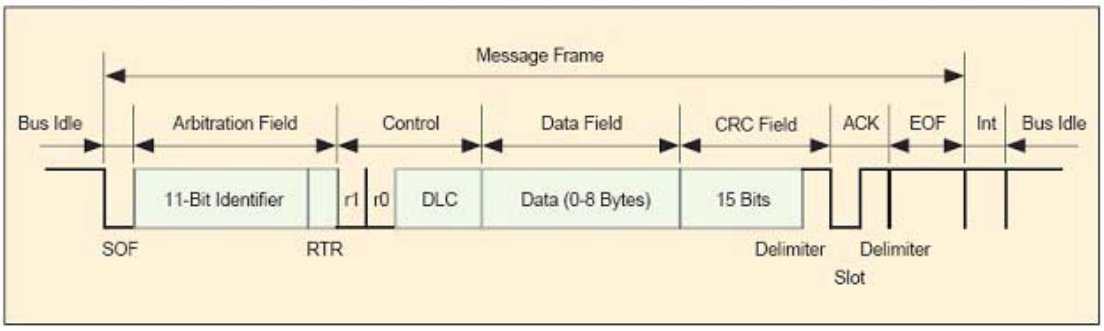
И покрај тоа што *token bus* протоколот е ефикасен и детерминистички при високи оптеретувања на мрежата, при мал сообраќај во каналот неговите перформансите не можат да се споредува со поротоколи со колизија. Општо земено, кога има многу чворови во еден логичен круг, голем процент од мрежното време се користи за праќање на симболот меѓу чворовите кога мрежниот сообраќај е мал.

DeviceNet (CAN Bus)

CAN е сериска комуникација развиена во главно за апликации во автомбилската индустрија но е способна да понуди добри преформанси во сите временско-критични индустириски

апликации. Тоа е релативно ефтина комуникација која ги поврзува уредите во мрежата, и е доста прифатена на во фабриките на нивото на управувани уреди. Овој протокол е оптимизиран за кратки пораки и користи CSMA/arbitration on message priority (CSMA/AMP) алгоритам за детекција на колизија.

Пораките имаат специфичен приоритет кој се користи во процесот на пристап на магистралата со арбитража во случај на симултана трансмисија. Протокот на битови при трансмисијата е сихронизиран со старт бит и арбитража е изведена врз идентификаторот на следната порака, во кој логичка нула е доминатна во однос на логичка единица. Чвор кој сака да испраќа порака чека додека е слободна магистралата и тогаш почнува го испраќа идентификаторот на својата порака бит по бит. Конфликтите за пристап на магистралата се решаваат за време на трансмисијата од старана на арбитражниот процес на нивото на битовите на арбитржното поле кое е иницијален дел на секоја рамка. Ако два уреди сакаат да испраќаат пораки во исто време, тие прво продолжуваат да праќаат рамки и да слушаат на мрежата. Ако еде од нив добие бит различен од онај кој го има испратено, губи право да продожи со испраќање, и другиот победува во арбитражата. Со овој метод трансмисијата што е во тек никогаш не е нарушена. Во CAN-базираната мрежа, податоците се испратени и примени со користење на рамки кои носат податоци до еден и повеќе чворови. Испратените податоци не мора да соодржат секогаш адреса на или на изворот или на дестинацијата на пораката. Наместо тоа, секоја порака е означена со идентификатор које е уникатен низ мрежата. Сите чворови на мрежата ја примаат пораката и ја прифаќаат или ја одбиваат, во зависност од конфигурацијата на маската на филтерите за идентификаторот. Овој мод е познат како продавач/корисник(*producer/consumer*) и е има предности во однос на TCP/IP чија комуникација е извор/дестинација *(source/destionation)* особено комуникациите во реално време каде повеќе дестинациони адреси треба да се обноват во исто време.



Рамката е долг сериски бајт, кој почнува со старт бит. Потоа следува идентификаторот на пораката кој е составен од 5-битен адресен коде (за 64 чворови) и 6-битен команден код. Битот спремен да прима(RTR *ready to receive it* bit) ќе биди поставе од машината која прима.
(Забелешка: двата и испраќачот и примачот делат иста жица) . Ако машината која прима не го постави битот потсетникот на пораката е откажан и пораката е пуштена подоцна. Следат 6 бита кои го индентификуваат бројот на бајти за испраќање, од 0-8, потоа 2 множества на битови за CRC (Cyclic Redundancy Check) за проверка на грешка т.н сума за проверка. Следниот бит *ACK slot* го поставува чворот кој прима ако пораката е успешно примена. Ако постои CRC грешка овој бит нема да биде испрате и пораката ќе биде препратена. На крајот на рамката следат битовите за крај(*end of frame* bits) кои се еквивалентни на стоп битови. Пред да се испрати следна порака мора да постои 3 бита интермисионо поле (INT) .

За разлика од CAN уредите, во DeviceNet, уредите со низок MAC ID не мора да го добиваат пристапот на магистралата. Приоритет за пристап зависи од групата на која припаѓа пораката т.е припадност на идентификаторот **(CAN ID)** во одреден опсег**.**

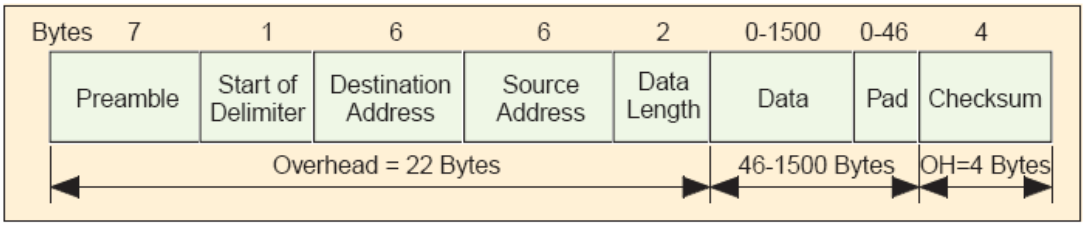
Message Group	CAN ID Range	Description
Group 1	0x000 to 0x3FF	Временски критични I/O пораки. Висок приоритет.
Group 2	0x400 to 0x5FF	Експлицитни и I/O пораки за предефинирани Master/Slave конекции. Дуплициран MAC ID проверка.
Group 3	0x600 to 0x7C0	Временски не критични пораки. Се користат за мониторинг и дијагностика
Group 4	0x7C0 to	Се користат за Off Line комуникација.

	0x7CF	

Предности
CAN е детерминистички протокол и е оптимизиран за кратки пораки. Приоритетот на пораките е специфициран во арбитрационото поле. Пораките со висок приорите секогаш добиваат пристап до медиумите со помош на арбитрацијата со што нивното изведување ке биде загарантирано.

Недостатоци
Главна мана на CAN –от во споредба со другите мрежи е малиот проток на податоци од 500 Kb/s). Затоа добивката од протоколот е ограничена. *Бит-синхронизацијата* ја лимитира максималната должина на *DeviceNet* мрежата. Истотака овој протокол не одговара за трансмисија на големи пораки и не подржува фрагментација на податоци кои се кои се повеќе од 8 бајти.

Ethernet
Ethernet користи CSMA/CD протокол за решавање на колизија во мрежата. Ако мрежата, чворот чека да се ослободи мрежата, во друг случај испраќа. Ако два и повеќе чворови одлучат да испраќаат во симултано пораките се корумпирани, чворовите престануваат со трансмисија и чекаат случајно време и потоа прават ретрансмисија. Случајното време се одредува според binary exponential backoff (BEB) алгоритам кој го одредува времето за ретрансмисија со избор меѓу 0 и (2^i −1) –от временски слот, каде i пресатвув i-тиот случај на детекција на колизија. После 10 колизии интервалот се фиксира на максимум 1024 слота. После 16 колизии чворот престанува да испраќа и го известува мрежниот микропроцесор за грешка. Форматот на рамката на Ethernet е преставен на слика()



Заглавието се состои од 22 бајти, делот за податоци од 46-1500 бајти. Рамките мора да бидат најмалку 64 бајти должина од дестинационата адреса до сумата за проверка (72 бајти, вкулучувајќи ја преамбулата и делимитерот). Ако големината на податокот е помала од 46 бајти остатокот се пополнува со празни места (*pad* полето).
Две причини се за ограничувањето на минимална должина. Прво, го олеснува разликувањето коректни рамки од „ѓубре“ што останува при отсекувањето на пораката при детекција колизија од страна на чворот кој трансмитира. Второ, го ограничува чворот да ја заврши трансмисијата на кратка рамка пред првиот бит да стигни до крајот на кабелот, каде можеби ќе се судри со друг чвор. За 10-Mb/s Ethernet максималната должина е 2500m и четри репитери со максимално дозволено време за рамката или слот време од 51.2 μs, време потребно за трансмисија на 64бајти во секунда.

Предности
Заради нискиот пристап на медиумот, Ethernet користи едноставен алгоритам и скоро и нема каснење при ниски оптоварувања на мрежата. Не е потребен дел од пропусниот опсег за пристап на мрежата споредено со мрежите кој користат магистрала или симболен прстен.

Недостатоци
Ethernet не е детерминистички протокол па не подржува приоритеризација на пораки. При високи оптертување на мрежата, колизијата на пораките преставуваат главен проблем бидејќи влијаат на протокот на податоци и временското каснење, што можи да е неограничено. Ethernet –виот ефект на фаќање вграден во BEB алгоритмот, во кој еден чвор врши трансмисија ексклузивно веќе подолго време и поткрај тоа што другите чворови чекаат за пристап на медиумот, е нефер и предизвикува незанемарливи деградација на преформансите. Базирани на BEB алгоритмот пораките се отпуштаат после серија на колизии па комуникација од едниот на другиот крај не е гарантирана. Заради бараната минимална големина на рамката, Ethernet користи големи пораки за премнос на мала количина на податоци.
Неколку решенија се препорачани при Ethernet во контролните апликации. На пример секоја порака можи да содржи временски печат пред да се испрати. Ова бара сихнронизација на сатовите што и не е така лесно особено со мрежа од ваков тип.

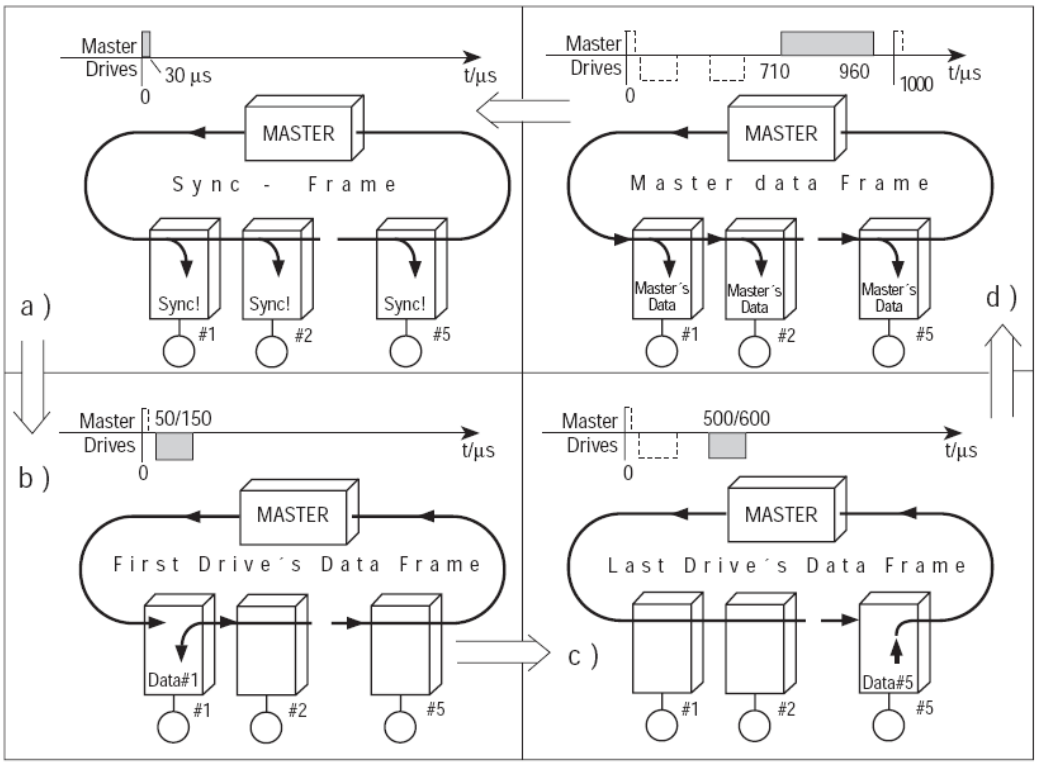
Разни шеми базирани на детерминистичка ретрансмисионо каснење за судрените пакети на базирање CSMA/CD протокол резултира намалување на каснењето на сите пакети. Како и да е ова е постигнато на сметка на инфериорните преформанси CSMA/CD при ниски и средни користење на каналот.
Други решенија се да се приоритизира CSMA/CD (т.е LonWorks) за да се подобри времето на одзив на временски критичните пакети. Користењето на switched Ethernet со делење на мрежната структура е уште еден начин да се зголеми ефикасноста.

Sercos

Сериската комуникација во реално време (***S**Erial **R**eal-time **CO**mmunication **S**ystem - SERCOS*) е отворен стандард дизајниран за мултиосните ситеми со контрола на движењето (*motion control*).
Комуникацијата се изведува циклично како master/slave комуникација со циклично време селектирано при иницијализација. Цикличното време може да биде со вредност 62 μs, 125 μs, 250 μs или било кој цел мултипликатор од 250 μs до 65 ms и е потребно ради сихронизација. Нумеричкиот контролер е секогаш комуникациониот master во SERCOS интерфејсот. Можни се
комуникации
контролер-контролер (C2C =Controller-to-Controller),
работник-работник(S2S = Slave-to-Slave),
работник-господар-работник(Cross CommunicationSMS)

При комуникацијата се користат 3 видови на пораки:

- синхронизациони пораки од master-от (ги добиваат сите двигатели и се користи за синхронизација на сите временско поврзани акции во нумеричкиот контролер и двигателите.
- податоци од master-от (се добиваат симултано од сите мотори и соодржат циклични податоци и сервисни податоци за сите двигатели во прстенот)
- пораки од двигателите (се испраќаат едно по друго во доделени временски слотови)

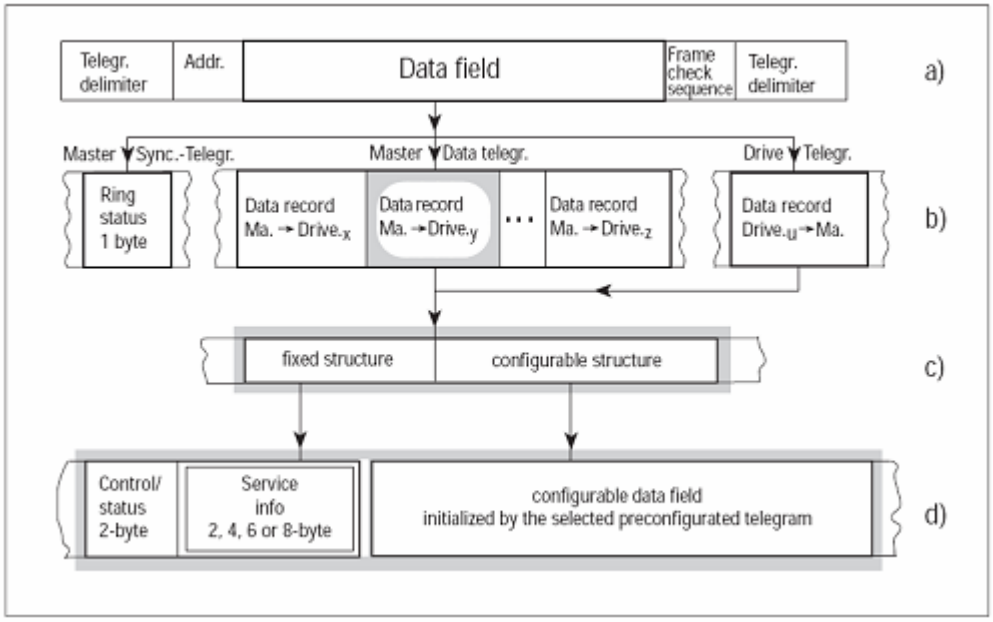


Пораките ја имаат следната структура:
-6 бајтна изворна адреса
-6 бајтна дестинацион адреса

-2 бајтен податок за типот на етернетот
-46-1500 бајти податоци
-4 бајти сума за проверка CRC

Со секој комуникационен циклус податоците во реално време можат да се трансмитираат во таканаречената „**конфигурабилно поле**“ за податоци. Настрана од нумеричките податоци како што се командни вредности и моментални вредности, можат да се пренесуваат бит листи со излезно/влезни инструкции.

Сервисни податоци се разменуваат само на барање на master-то со *handshaking* процедура во порции од 2, 4, 6 или 8-бајти во сервисното-инфо поле (service info field) и после се реасемблираат кај примачот.



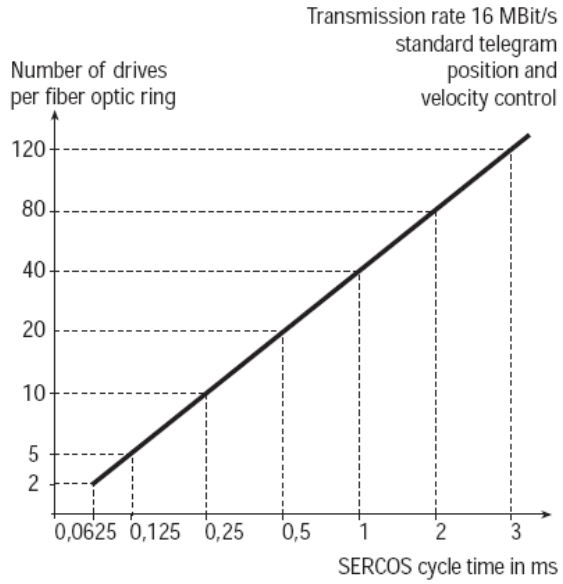
За време на размената на сервисни податоци и дефиниција на податоците во реално време, податоците се адресирани во SERCOS интерфејсот преку идентификациони броеви.
ID броеви од 1 до 32767 се резервирани за податоци специфицирани SERCOS асоцијацијата. додека броевите од 32768 од 65535 се достапни на производителите на производот за дефинирање на податоците или параметрите не покриени со стандардот но кои се потребни за операциите на производот.

IDN:	**S-0-0036**
Name:	**Velocity Command Value**
Attribute:	**4 bytes, signed integer, 1 DP**
Unit:	**RPM**
Minimum:	**-2^{31}**
Maximum	**$+2^{31} - 1$**
Operation Data	**1000.0**

S-0-0038 Positive Velocity Limit Value
S-0-0039 Negative Velocity Limit Value
S-0-0040 Velocity Feedback Value
S-0-0043 Velocity Polarity Parameter
S-0-0092 Bipolar Torque Limit Value
S-0-0124 Standstill Window
S-0-0155 Friction Torque Compensation

Како додаток на податоците во реално време се разменуваат и следните параметри
• Комуникациони подесувања
• Селекција на модот (торк контрола, брзинска контрола, позициона контрола)
• Прилагодувања на различни механички системи
• Прилагодување на различни транскондуктери за мерење и организирање
• Прилагодување на командните вредности и моменталните вредности на тие во нумеричкиот контролер

Должината на секоја трансмисиона секција може да биде до 50 метри со пластичени фибер оптички кабли и до 250 метри со стаклени фибер кабли и 254 на преплатници со секој фибер оптички прстен. Зависноста на бројот на претплатници со цикличното време е дадено на сл.



Грешките во трансмисијата на сервисните податоци (параметри и дијагностички сигнали) се корегираат преку *handshaking* процедура. Грешните податоци се ретрансмитираат.
Податоците во реално време (командните вредности и моменталните вредности) се корегираат автоматски со обновување со секој комуникационен циклус. Во случај на комуникациона грешка , последните валидни командни вредности се користат до следниот циклус. Двигателите се стопираат во случај на две последователни грешни трансмисии.

***Синхроно*-мултиплексирање со поделба на кодот** *Code Division Multiplexing (Синхрон CDMA)*

Синхрониот CDMA, и уште позната како мултиплексирање на поделениот(на делови) (*Code Division Multiplexing* -CDM), ги искористува математичките својства на ортогоналноста.

Да претпоставиме дека сигналите со податоците се преставени преку вектори. Пример бинарниот стринг "1011" преставен со вектор **a**(1,0,1,1) има ортогонален вектор **b**(1, -1, -1, 0) кој е добиен од својството да скаларен производ $\mathbf{a} \cdot \mathbf{b}$ е еднаков на нула:

$$(1) \times (1) + (0) \times (-1) + (1) \times (-1) + (1) \times (0) = 1 + (-1) = 0.$$

Да претпставиме дека имаме множество на вектори кои се меѓусебно ортогонални (овие вектори се конструираат за полесно декодирање на колоните и редовите од Волшовата матрица конструирана од Волшовите функции. Сега на секој испраќач на вектори му се доделува чип код **v** . Секоја 0 во векторот **a** се заменува со -**v**, и секоја единица со **v** па се добива (**v**, -**v**, **v**, **v**). што значи дека за **v**=(1,-1) би се добил трансмисион вектор ((1,-1),(-1,1),(1,-1),(1,-1)).
Секој испраќач има уникатен чип код (еден ред или колона од Волшовата матрица, во нашиот пример со димензии 2X2 вкупно 4 корисници), но конструирањето на трансмисиониот векторе е идентичен.

Физичките особини на интерфејсот велат дека ако два сигнали во оваа точка се во фаза, тие ќе се соберат за да дадат двапати поголема амплитуда за секој сигнал, но ако се во против фаза, тие ќе се одземат и ќе дадат сигнал кој е разлика на амплитудите. Дигитално, оваа особина може да се измоделира едноставно со собирање на трасмисиониот векотор, компонента по компонента.

Да речеме дека постојат испраќач0 со чип код (1,-1) и податоци (1,0,1,1), и испраќач1 со чип код (1,1) и податоци (0,0,1,1) и дека тие вршат трансмисиија во исто време во исто тело на воздухот.

Чекор	Кодирање на испраќачот0	Кодирање на испраќачот1
0	чипкод0=(1,-1), податок0=(1,0,1,1)=(1,-1,1,1)	чипкод1=(1,1), податок1=(0,0,1,1)=(-1,-1,1,1)

1	кодирање0=чипкод0.податок0	кодирање1=чипкод1.податок1
2	кодирање0=(1,-1).(1,-1,1,1)	кодирање1=(1,1).(-1,-1,1,1)
3	транс вектор0=((1,-1),(-1,1),(1,-1),(1,-1))	транс вектор1=((-1,-1),(-1,-1),(1,1),(1,1))
4	сигнал0=(1,-1,-1,1,1,-1,1,-1)	сигнал1=(-1,-1,-1,-1,1,1,1,1)

Бидејќи сигналот на испраќачите се испраќа истовремено сировиот сигнал во воздухот ќе биде: (1,-1,-1,1,1,-1,1,-1) + (-1,-1,-1,-1,1,1,1,1) = (0,-2,-2,0,2,0,2,0)

Сировиот сигнал може да се нарече интерфејс низ. Примателот ги знае чип кодовите на секој испраќач, и ова знаење може да се комбинира со примениот интерфејс низ за да се екстрактова разбирлив сигнал од секој испраќач.

Чекор	Декодирање на испраќачот0	Декодирање на испраќачот1
0	чипкод0=(1,-1), добиен низ=(0,-2,-2,0,2,0,2,0)	чипкод1=(1,1), добиен низ=(0,-2,-2,0,2,0,2,0)
1	decode0=pattern.vector0	decode1=pattern.vector1
2	decode0=((0,-2),(-2,0),(2,0),(2,0)).(1,-1)	decode1=((0,-2),(-2,0),(2,0),(2,0)).(1,1)
3	decode0=((0+2),(-2+0),(2+0),(2+0))	decode1=((0-2),(-2+0),(2+0),(2+0))
4	data0=(2,-2,2,2)=(1,0,1,1)	data1=(-2,-2,2,2)=(0,0,1,1)

Декодирањето се врши на тој начин што чипкод векторот **v**=(a,b) се множи скаларно со секој бит пар **u**=(c,d) од добиениот ситнал (a*c + b*d). Така при декодирање на првиот бит пар (0,-2) се добива :

(0,-2).(1,-1) = (0*1 + -2*-1) = (0+2) = 2

На крајот, после декодирањето, секоја вредност што е поголема од 0 се интрепретира со 1 додека сите вредности помали од 0 се интерпретираат со 0. Така примениот податок0 (2,-2,2,2), ќе биде интерпретиран како (1,0,1,1).

Мултиплексирањето бара корисниците да бидат координирани така да секој врши трансмисија на чипкод секвенцата почнувајќи точно во исто време Оваа техника наоѓа употребаво база-мобилни линкови каде сите трансмисии потекнуваат од ист предавател и можат префекно да се координираат

***Асинхрono*-мултиплексирање со поделба на кодот -** *Code Division Multiplexing (Асинхрон CDMA*)

Претходниот протокол покажува како два корисници можат да комуницираат мултиплексирано во синхрон систем. Од друга страна, мобилни-база линковите не можат синхроно прецизно да се координираат, дел заради мобилноста на слушалките, и бара поинаков пристап. Бидејќи не е можно математички да се креира потписна секвенца кои се ортогонални во арбитражни случајни стартни точки, уникатни „псеудо-случајни“ или „псеудо шум“ (PN) секвенца се користат во Асинхроните CDMA системи. Овие PN секвенци се статички не се во меѓусебна врска, и сума од голем број на PN секвенци резултира со повеќекратен интерфејс за пристап (*Multiple Access Interference* -MAI) кој е апроксимиран со Гаусов процес за шумови (*Gaussian noise process*) преку теоремата „закон за големи броеви“ во статистиката . Ако сите корисници примиле исто ниво на снагата, тогаш варирањето (т.е шумот на снагата) на MAI-от се зголемува директно пропорционално со бројот на корисници.

Сите форми на CDMA користат широк спектар на процесното засилување за да му овозможат делумна дискриминација на примателите во однос на несаканите сигнали. Сигналите со саканиот чип код и време се примени, додека сигналите со различени чип кодови (или со ист код за простирање но различни временски офсети) се појавуваат како широкопојасен шум намален од процесното засилување. Како секој корисник генерира MAI, контролирањето на снагата на сигналот како важна особина на CDMA предавателите. CDM (Синхрониот CDMA), TDMA(мултиплексирање со поделба на времескиот домен) или FDMA(мултиплексирање со поделба на фрекфентниот домен) примателот може теоретски комплетно да ги отфрли силните сигнали кои користат различни кодови, времески слотови или фрекфентни канали заради ортогоналноста на овие системи. Ова не е така кај Асинхрониот; отфрлањето на несаканите сигнали е само делумно. Ако некој или сите од несаканите сигнали се многу посилни од саканиот сигнал, тие ќе го препокријат. Ова ќе доведи до главното барање на Асинхрониот CDMA систем да врши апрокимативно споредување на различнита снага на сигналите испратени од предавателите. Во CDMA мобилната телефониј, базната станица користи шема за контрола на снагата со брза повратна врска за цврста контрола на секој мобилена трасмисија.