[P220]

Увод у архитектуру рачунара

10

Саша Малков
Универзитет у Београду
Математички факултет
2013/2014

[P271]

Увод у архитектуру рачунара

Саша Малков

Тема 11

Улазно / излазни уређаји (наставак)

DMA (1)

- У случају уређаја који преносе веће количине података или раде временски захтевне послове, програмирани У/И води великом утрошку времена на чекање
- *DMA* има за циљ да се процесор ослободи старања о преносу података и посвети другим стварима

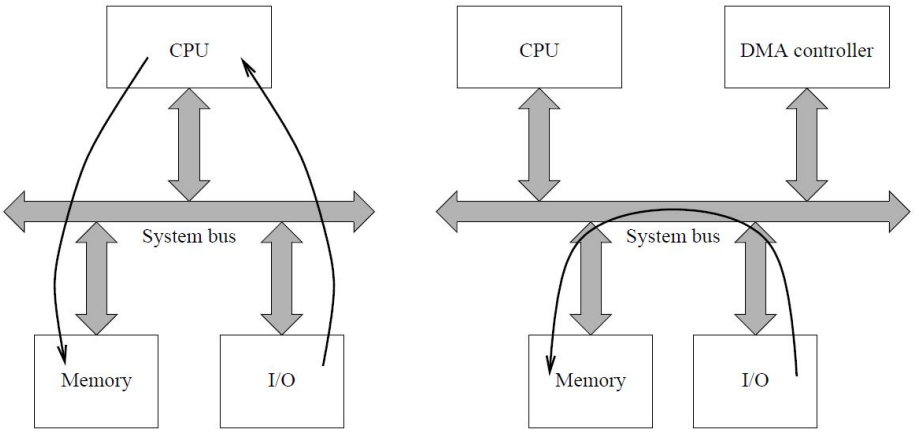
Универзитет у Београду - Математички факултет

DMA (2)

- *DMA* се имплементира помоћу контролера *DMA*
  - Контролер се понаша као подређен уређај у односу на процесор и прима инструкције за пренос података од процесора
  - Затим преузима контролу над магистралом и остварује пренос података
- Контролер уобичајено подржава већи број уређаја
  - сваки се повезује на посебан канал *DMA*

Универзитет у Београду - Математички факултет

Однос програмираног У/И и непосредног приступа меморији



Кораци операције *DMA*

1. Иницијализација канала
2. Преношење података
3. Обавештавање процесора

Кораци операције *DMA*

1. Иницијализација канала:

- процесор иницира контролер *DMA* и шаље му:
  - број уређаја
  - адресу простора у меморији
  - број бајтова који се преносе
  - смер преношења података
- након иницијализације канал је спреман за преношење података

2. Преношење података

3. Обавештавање процесора

Универзитет у Београду - Математички факултет

Кораци операције *DMA*

1. Иницијализација канала:

2. Преношење података

- када У/И уређај буде спреман за преношење података, обавештава о томе контролер *DMA*.
- контролер започиње операцију преноса:
  - контролер захтева магистралу (и добија је уобичајеним поступком арбитраже)
  - поставља меморисјку адресу и одговарајући сигнал (читање или писање) на магистралу
  - довршава пренос и ослобађа магистралу
  - ажурира адресу и бројач
  - ако има још података за преношење, понавља поступак

3. Обавештавање процесора

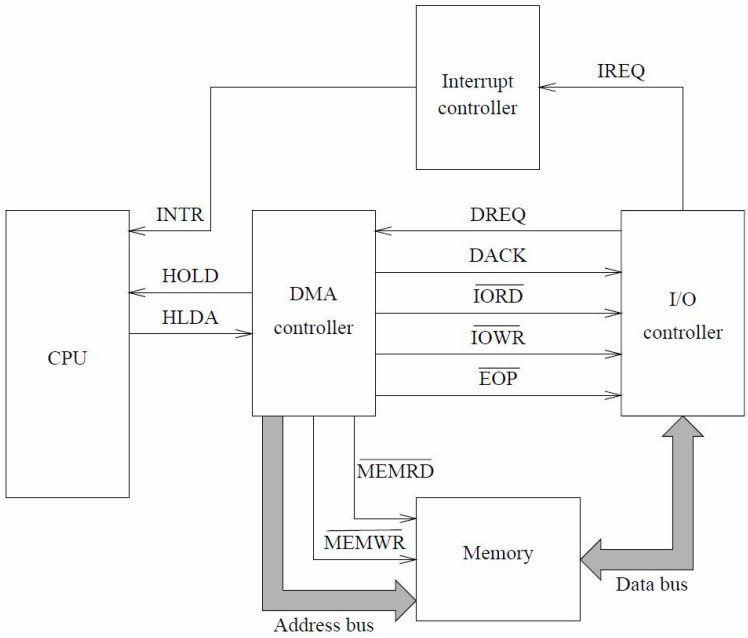
Универзитет у Београду - Математички факултет

Кораци операције *DMA*

1. Иницијализација канала:
2. Преношење података
3. **Обавештавање процесора**
   - након довршеног преноса обавештава се процесор
   - уобичајено се за то користи систем прекида
   - процесор затим проверава стање преноса

Универзитет у Београду - Математички факултет

Поједностављени дијаграм употребе контролера *DMA*



Универзитет у Београду - Математички факултет

Пример преноса преко *DMA*

- У примеру претпостављамо да се ради о операцији читања путем *DMA*
  - одговарајућим командама процесора *DMA* контролеру иницијализован је канал *DMA* да подржи конкретан У/И уређај (или контролер)
  - подаци се са У/И уређаја и уписују у меморију
  - претпостављамо да се преносе две речи података
  - претпостављамо да је у питању брз пренос који потпуно задржава магистралу до краја процеса

Пример преноса преко *DMA* (2)

- Када У/И уређај буде спреман да преноси податке, шаље захтев контролеру *DMA* путем линије *DREQ*
- Када прими сигнал *DREQ*, контролер *DMA* шаље процесору захтев за добијање магистрале, путем сигнала *HOLD*
- Након завршетка текуће инструкције, процесор ослобађа магистралу и то јавља контролеру *DMA* путем сигнала *HLDA*
  - процесор надаље не користи магистралу већ одлаже све операције које захтевају постављање сигнала на магистралу
- По пријему сигнала *HLDA* контролер *DMA* обавештава У/И уређај (путем сигнала *DACK*) да може да почне пренос података
  - након тога сигнал *DREQ* се уобичајено склања

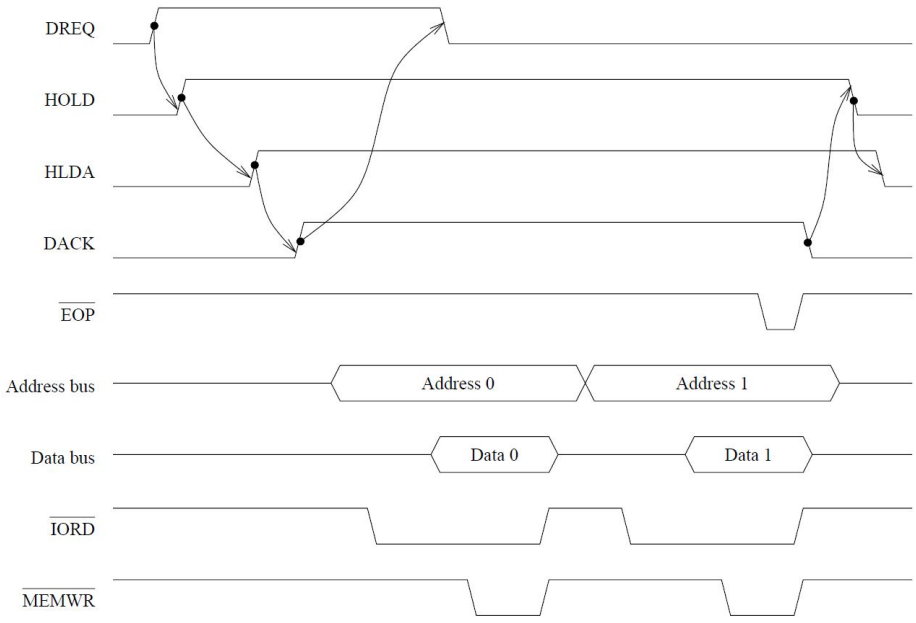
Пример преноса преко *DMA* (3)

- Контролер *DMA* је одговоран за постављање одговарајућих контролних сигнала за читање са уређаја (*IORD'*) и писање у меморију (*MEMWR'*)
  - У/И уређај одговара на сигнал читања и поставља податке на магистралу података
    - не користи се адреса, зато што је уређај и канал везе иденитификован иницијализацијом *DMA*
  - меморија одговара на сигнал писања и уписује податке са магистрале података на одговарајућој адреси
  - бројач пренесених речи се смањује
  - адреса се увећава за величину речи
  - ако је бројач различит од нуле, понавља се овај циклус

Пример преноса преко *DMA* (4)

- Када се преношење података доврши, контролер *DMA* означава да је операција завршена постављањем сигнала *EOP'* ("*end of process*")
  - том приликом контролер склања и сигнал *DACK*
- Затим контролер *DMA* склања сигнал *HOLD* чиме враћа магистралу на употребу процесору
- Процесор одговара склањањем сигнала *HLDA*
- Поступак је тиме довршен

Дијаграм употребе магистрале током преноса путем *DMA*



Универзитет у Београду - Математички факултет

Пример преноса преко *DMA* (5)

- У случају споријих уређаја пренос се одвија у другачијем режиму
  - контролер *DMA* за свако појединачно преношење података понавља захтевање и ослобађање магистрале
  - сваки циклус почиње сигналом *DREQ* од стране У/И уређаја, а завршава ослобађањем магистрале
  - крај процеса се означава сигналом *EOP'* који контролер *DMA* шаље У/И уређају

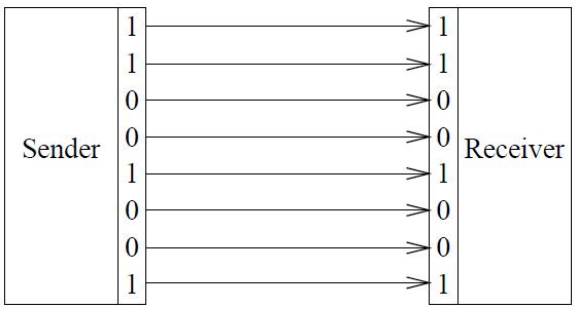
Универзитет у Београду - Математички факултет

Врсте техника преноса података

- Пренос података може бити
  - Паралелан
  - Серијски
    - Синхрони
    - Асинхрони

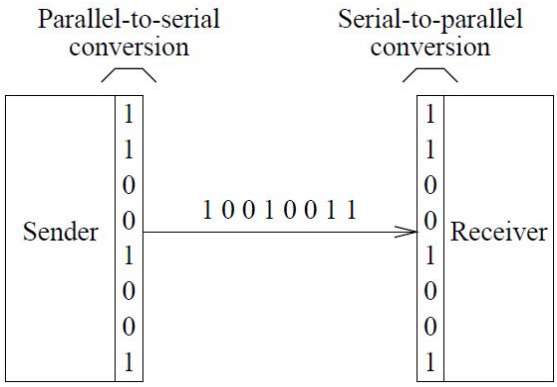
Паралелан пренос података

- Неколико битова се преноси истовремено кроз паралелне водове



Серијски пренос података

- За пренос података се користи један вод
- Нема паралелног преношења података



Универзитет у Београду - Математички факултет

Однос паралелног и серијског преноса

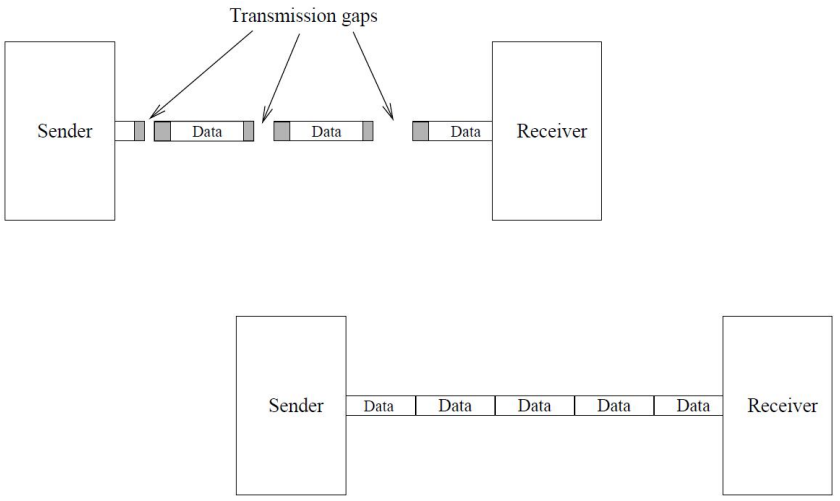
- Парелни пренос
  - кроз n паралелних водова се може истовремено преносити n битова
  - бржи је
  - скупљи је за имплементацију
  - са повећавањем дужине водова и брзине рада расте вероватноћа појављивања *искривљења* (*дисторзије*)
    - неки битови стижу раније или касније, несинхронизовано са осталим битовима
  - користи се углавном на мањим даљинама
- Серијски пренос
  - јевтинији
  - нема могућности искривљења података

Универзитет у Београду - Математички факултет

Серијски пренос података

- Може да буде
  - синхрони
    - часовници пошиљаоца и примаоца су синхропнизовани пре преношења података
  - асинхрони
    - сваки бајт се енкодира за пренос тако да часовници пошиљаоца и примаоца не морају да буду усклађени
    - сваки пакет (нпр. бајт) је окружен тзв. почетним (старт) битовима и зауставним (стоп) битовима
- Синхрони пренос је
  - ефикаснији
  - скупљи

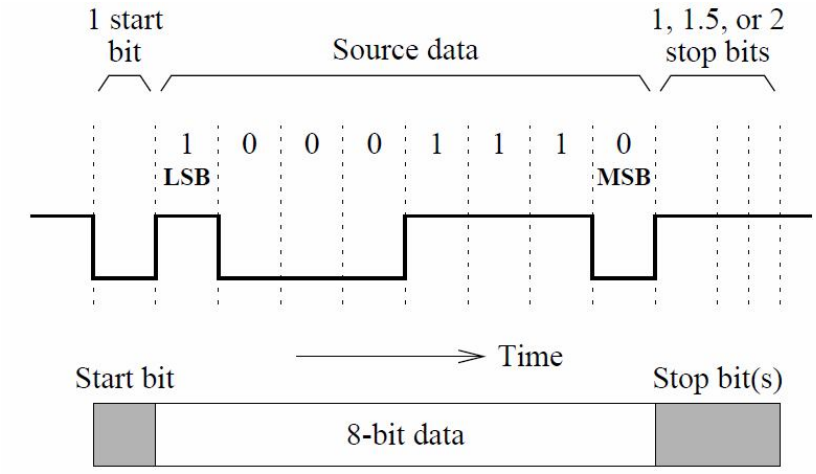
Асинхрони и синхрони пренос података



Асинхрони пренос

- Комуникациона линија је у активном стању док не постоји пренос
- При преношењу података почетни бит спушта стање линије
  - тако прималац препознаје почетак пакета
- Прималац
  - зна дужину битова
  - зна и величину пакета (нпр. 8 битова)
  - чита стања линије у срединама периода
  - игнорише почетни бит и слаже пакет (бајт)
- Зауставни бит има две намене
  - на крају пакета не сме остати 0, зато што прималац не би могао да препозна почетак наредног пакета
  - оставља довољно времена примаоцу да сложи пакет од примљених битова
- Уобичајено се користи 1, 1.5 или 2 зауставна бита

Сигнали при асинхроном преносу података



Пример паралелног интерфејса

- Интерфејс за штампаче (*Centronics*)
  - 25 линија, познат и под именом (*DB-25*)
- Интерфејс је једноставан:
  - 8 линија за податке
  - 1 линија резе за податке (као часовник)
  - руковање (*handshaking*) се остварује сигналом *ACK*
    - након сваког испорученог бајта рачунар чека да прими сигнал *ACK* пре слања наредног бајта
  - 5 линија статуса
    - *busy, out of paper, online/offline, autofeed, fault*
  - 2 контролна сигнала
  - 8 линија уземљења

Пример паралелног интерфејса – сигнали интерфејса *DB*-25

Pin #	Signal	Signal direction	Signal function
1	STROBE	PC ⟹ printer	Clock used to latch data
2	Data 0	PC ⟹ printer	Data bit 0 (LSB)
3	Data 1	PC ⟹ printer	Data bit 1
4	Data 2	PC ⟹ printer	Data bit 2
5	Data 3	PC ⟹ printer	Data bit 3
6	Data 4	PC ⟹ printer	Data bit 4
7	Data 5	PC ⟹ printer	Data bit 5
8	Data 6	PC ⟹ printer	Data bit 6
9	Data 7	PC ⟹ printer	Data bit 7 (MSB)
10	ACK	printer ⟹ PC	Printer acknowledges receipt of data
11	BUSY	printer ⟹ PC	Printer is busy
12	POUT	printer ⟹ PC	Printer is out of paper
13	SEL	printer ⟹ PC	Printer is online
14	AUTO FEED	printer ⟹ PC	Autofeed is on
15	FAULT	printer ⟹ PC	Printer fault
16	INIT	PC ⟹ printer	Clears printer buffer and resets printer
17	SLCT IN	PC ⟹ printer	TTL high level
18–25	Ground	N/A	Ground reference

SCSI

- Паралелни интерфејс
  - ширине 8 битова – *уски SCSI*
  - ширине 16 битова – *широки SCSI*
- Развијен иницијално као *SASI* (*Shugart Associates System Interface*) (1979)
- Касније прихваћен као *ANSI* стандард *SCSI* 1 (1986)
  - 8-битни, 5MB/s
- У више наврата унапређиван
  - Почев од *Ultra* 3 (*Ultra* 160) користи пренос два бита по каналу по циклусу (као *DDR*)

SCSI (2)

SCSI type	Bus width (bits)	Transfer rate MB/s
SCSI 1	8	5
Fast SCSI	8	10
Ultra SCSI	8	20
Ultra 2 SCSI	8	40
Wide Ultra SCSI	16	40
Wide Ultra 2 SCSI	16	80
Ultra 3 (Ultra 160) SCSI	16	160
Ultra 4 (Ultra 320) SCSI	16	320

SCSI (3)

- Права магистрала
  - није *point-to-point* интерфејс (ограничен на међусобну комуникацију два уређаја)
  - сваки уређај се идентификује јединственим бројем
    - уска верзија (8-битна) има бројеве уређаја 0-7
    - широка верзија (16-битна) има бројеве 0-15
  - уска верзија има интерфејс са 50-линија
  - широка верзија има интерфејс са 68 линија
  - подржава интерне и екстерне уређаје
    - повезивање спољашњих уређаја се остварује уланчавањем
    - да би се омогућило даље повезивање, сваки екстерни уређај мора да има по један улазни и један излазни конектор

SCSI (4)

- Начелна допуштена дужина кабла је око 25m
- Са повећавањем броја уређаја смањује се дужина
  - ако су повезана више од два уређаја, допуштена дужина се уобичајено редукује на 12m
- Арбитража магистрале се остварује применом једноставне политике приоритета

SCSI (5)

- Линије
  - Ради смањивања утицаја шумова, свака линија има одговарајућу линију уземљења
  - Користи један бит парности за сваки пренесени бајт
  - Додатно се користи алгоритам *CRC* за заштиту интегритета пренесених података

SCSI (5)

- Користи модел клијент-сервер (*иницијатор* и *циљ*)
  - уобичајено, иницијатори су контролери а циљеви уређаји попут дискова или скенера
- Комуникација:
  - иницијатор шаље команду одабраном циљу
  - циљ потврђује избор и прима податке од иницијатора
  - пренос података тече у фазама
    - команда
    - читање поруке (*message in*)
    - писање поруке (*message out*)
    - читање података (*data in*)
    - писање података (*data out*)
  - циљ је одговоран за управљање магистралом између фаза

SCSI (6)

- Подржава асинхрони режим рада
  - метод руковања сигналима *REQ* и *ACK* за сваки пренесени пакет (8 или 16 битова)
- Подржава и синхрони режим рада
  - не користе се сигнали *REQ* и *ACK*
  - више перформансе

Universal Serial Bus – USB

- Изворно развијен 1995.
- Најшира до сада спроведена иницијатива за пројектовање интерфејса за повезивање периферија
- Основни циљ
  - омогућити повезивање рачунарских периферија једнако једноставно као што се повезују телефон или пегла
- Стандарди
  - 1.0 из 1996.
    - *ниска* брзина 1.5Mbps, *пуна* брзина 12Mbps
  - 1.1 из 1998.
    - разрешени неки проблеми са протоколом
    - први широко распрострањен стандард
  - 2.0 из 2000.
    - *висока* брзина, до 480 Mbps
  - 3.0 спецификације објављене 2008.
    - брзина до 4800 Mbps

USB – Неки од најважнијих циљева

- Обезбеђивање хомогеног рачунарског интерфејса
  - идеја је да се мноштво различитих и некомпатибилних интерфејса замене једним универзалним интерфејсом
- Обезбеђивање дељивог интерфејса
  - омогућавање серијског везивања уређаја се значајно смањују проблеми са каблирањем
- Превазилажење проблема са ресурсима система
  - додавање нових уређаја доноси проблеме разрешавања конфликата око адреса или прекида
  - *USB* уређаји не захтевају ни меморију ни адресни простор ни прекиде
- Једноставнија инсталација и конфигурација
  - потпуна аутоматизација (*plug-and-play*) за разлику од старијих уређаја, који су често захтевали мануелно конфигурисање
- Повезивање током рада рачунара
  - старији начини повезивања уређаја су захтевали искључивање рачунара пре повезивања
  - повезивање *USB* уређаја се може обављати без искључивања рачунара

Универзитет у Београду - Математички факултет

USB – Додатне напредне особине

- Напајање уређаја кроз интерфејс
  - кабл за повезивање има линије напајања, чиме је омогућено да се уређаји напајају струјом напона +5V и јачине 100-500mA
- Двосмерна контрола уређаја
  - подаци могу да теку у оба смера
  - омогућена је пуна контрола уређаја, као и употреба уређаја за контролисање рачунара
- Проширивост помоћу *хабова*
  - на једно прикључно место се може повезати хаб (разделник) ради повећавање броја прикључака или постављања прикључака на жељено место
- Уштеда енергије
  - *USB* уређаји аутоматски улазе у примирено (суспендовано) стање ако нема активности на магистрали 3ms
  - у примиреном стању захтевају свега око 2.5mA
- Препознавање и отклањање грешака
  - користи се алгоритам *CRC* за препознавање грешака у комуникацији
  - у случају препознавања грешке, трансакција се понавља

Универзитет у Београду - Математички факултет

USB – Пренос података

- Кабл има четири линије:
  - две служе за напајање уређаја
  - две служе за преношење сигнала
- За преношење података се употрбљава схема енкодовања *NRZI* (*nonreturn to zero-inverted*)

Универзитет у Београду - Математички факултет

Схема енкодовања *NRZ* (*nonreturn to zero*)

- 0 је низак а 1 висок ниво сигнала
- Проблеми:
  - Нема промена сигнала ако се преносе дуги низови нула или јединица
    - промене су важне за примаоца због синхонизације и препознавања података
  - У случају шумова, тешко је препознати нивое 0 и 1, али се промене ипак лако препознају

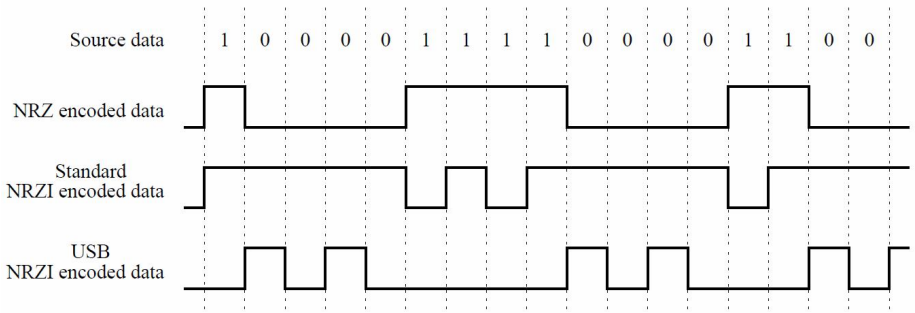
Универзитет у Београду - Математички факултет

Схема енкодовања *NRZI*
(*nonreturn to zero – inverted*)

- Решава неке од проблема схеме *NRZ*
- Не користи апсолутне нивое за представљање података, већ само промене стања
- Стандардно енкодовање *NRZI* подразумева:
  - сигнал се мења ако је наредни бит 1
  - сигнал се не мења ако је наредни бит 0
  - смер промене није значајан
- Енкодовање *NRZI* у случају *USB*-а је другачије:
  - сигнал се мења ако је наредни бит 0
  - сигнал се не мења ако је наредни бит 1
  - смер промене није значајан

Универзитет у Београду - Математички факултет

NRZI



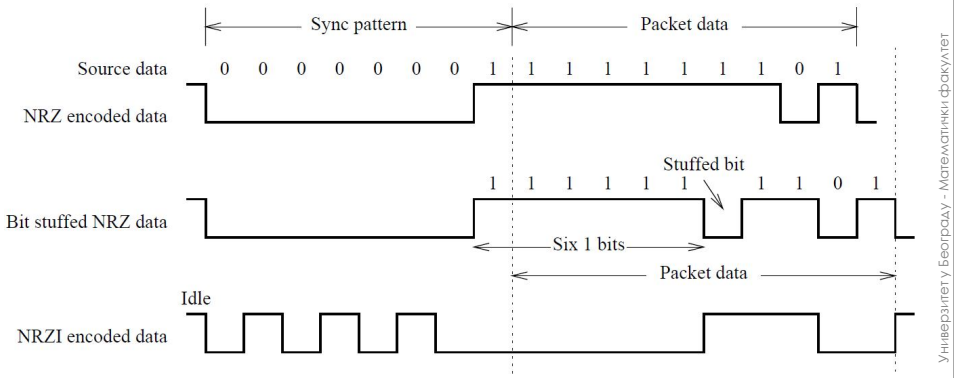
Универзитет у Београду - Математички факултет

Уметање битова

- Описана схема енкодовања *NRZI* решава неке од проблема:
  - ниво сигнала не игра главну улогу
    - посматрају се само транзиције
  - отклањају се дугачки низови непромењених стања у случају низова јединица у оригиналним подацима
- Преостаје проблем:
  - опстају дугачки низови непромењених стања у случају јединица
- Решење – предузима се тзв. *уметање битова*
  - након сваких 6 узастопних јединица се умеће једна 0
  - уметање је на нивоу података, а не сигнала

Универзитет у Београду - Математички факултет

Уметање битова (2)



Универзитет у Београду - Математички факултет

USB – Пренос података (2)

- Подржана су четири типа преноса података:
  - употреба прекида
  - изохрони пренос
  - контролисани пренос
  - пакетни пренос (*bulk*)

Употреба прекида

- Намењен за уређаје који примају или производе мале количине података, са малим асинхроним учесталостима
- *USB* не подржава уобичајене прекиде
- Напротив, користи испитивање стања
- Да би се подигле перформансе, може се бирати одговарајућа учесталост проверавања
  - *USB* 1.1 допушта интервале од 1-255ms, тј. од 1000 до око 4 пута у секунди
- У једном кораку се може пренети највише
  - 8 бајтова – ниска брзина
  - 64 бајта – пуна брзина
  - 1024 бајта – висока брзина

Изохрони пренос података

- Намењен за примене у реалном времену и уређаје који захтевају сталну брзину прилива/одлива података
  - мултимедија, телефонија
- Не користи се препознавање и отклањање грешака
- Овај вид преноса није подржан при ниској брзини рада

Контролисани пренос података

- Намењен за конфигурисање и припрему уређаја за рад
- Представља веома брз непериодичан начин рада
- Пренос обухвата три корака:
  - *корак припреме*
    - циљном уређају се упућује одговарајући захтев
  - *корак података*
    - опциони корак – уређај испоручује одговарајуће податке
  - *корак статуса*
    - извештавање о успеху операције
- Овај вид преноса је подржан при свим брзинама рада

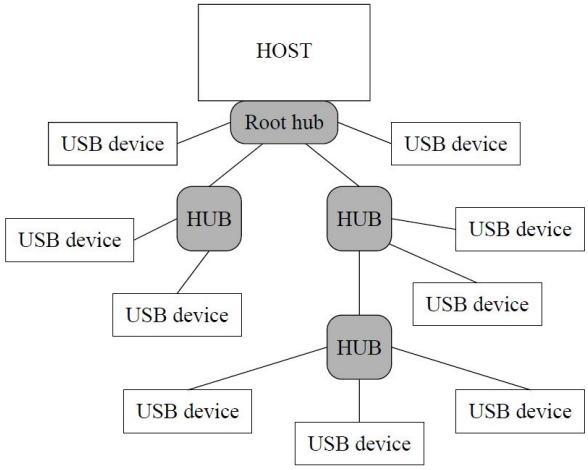
Пакетни пренос података (*bulk*)

- Намењен за уређаје који немају унапред познат ритам рада или преносе веће количине података без стриктне зависности од времена
  - преношење материјала за штампање штампачу
  - комуникација са спољашњим диском или флеш диском
  - скенер
- Овај вид преноса има низак приоритет, да не би изгладњивао остале
- Подржава препознавање грешака
- Овај вид преноса није подржан при ниској брзини рада

Архитектура *USB* повезивања

- Основни хардвер чине
  - матични контролер *USB*-а (*host controller*)
    - служи да иницира трансакције
  - корени хаб (*root hub*)
    - служи да успостави везу контролера са циљним уређајем

Пример архитектуре



Универзитет у Београду - Математички факултет

Матични контролери

- Постоје две врсте
  - *Open Host Controller* (*OHC*)
    - *Compaq, Microsoft, National Semiconductor*
  - *Universal Host Controller* (*UHC*)
    - *Intel*

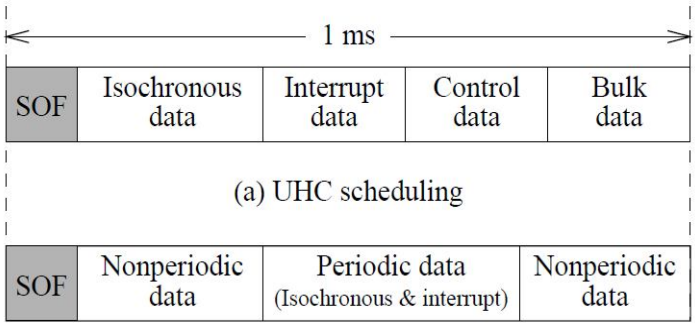
Универзитет у Београду - Математички факултет

Матични контролери (2)

- Основна разлика је у политици распоређивања четири врсте преноса у оквиру оквира
  - У оба случаја оквир траје 1ms
  - *UHC* распоређује
    - периодичне преносе (изохрони и прекидни) на почетак
      - могу да добију до 90% опсега
    - контролни и пакетни пренос иду након њих
      - гарантује се 10% опсега за контролни пренос
  - *OHC*
    - резервише простор за непериодичне преносе на почетку оквира, тако да им гарантује 10% опсега
    - затим се периодичним преносима гарантује 90% опсега
    - ако остане простора, поново се распоређују непериодични преноси

Универзитет у Београду - Математички факултет

Матични контролери (3)



Универзитет у Београду - Математички факултет

[P271]

Увод у архитектуру рачунара

Саша Малков

Тема 12

Систем прекида

Функција система прекида

- Систем прекида представља један од механизама за управљање током рада процесора
  - поред програмских позива и скокова
- У основне примене система прекида улазе
  - комуникација са У/И уређајима
  - употреба основних услуга ОС-а

Начин рада прекида

- Прекиди се могу изазвати хардверски или софтверски
- По наступању прекида, тренутно извршавани програм се прекида и прелази се на контролну процедуру која се назива *рутина за обраду прекида* (*interrupt service routine*) или *опслужилац прекида* (*interrupt handler*)
- Када се опслуживање прекида доврши, програм наставља са извршавањем од места на коме је прекинут
  - слично као у случају позивања потпрограма

Универзитет у Београду - Математички факултет

Хардверски прекиди

- Производе се од стране спољашњих уређаја
- Називају се и *непланski* или *асинхрони* прекиди
- Служе за *скретање пажње* процесора на одговарајући уређај
- Постоје две врсте хардверских прекида
  - Маскирајући прекиди
    - Могу да се одложе до неког специфичног тренутка
    - Нпр, ако прекид наступи током обраде претходног прекида
  - Немаскирајући прекиди
    - Одмах се обрађују од стране процесора

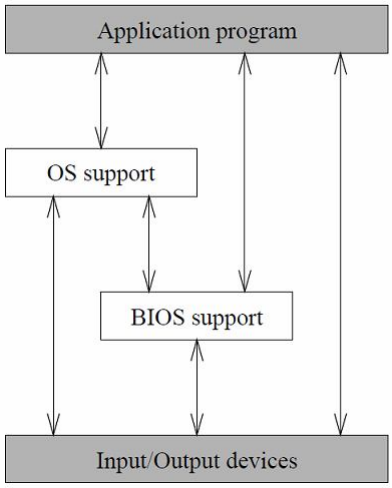
Универзитет у Београду - Математички факултет

Софтверски прекиди

- Називају се и *планski* или *синхрони* прекиди
- Већина процесора подржава софтверске прекиде
  - *Intel x86* – инструкција *int*
  - *PowerPC* – инструкција *sc* (*system call*)
  - *MIPS* – инструкција *syscall* (*system call*)
- У основне намене софтверксих прекида спадају:
  - приступање У/И уређајима
  - коришћење услуга ОС-а или *BIOS*-а

Универзитет у Београду - Математички факултет

Комуникација са У/И уређајима



Универзитет у Београду - Математички факултет

Изузеци

- Посебна подврста хардверских прекида су *изузеци* (*exceptions*)
- Механизам прекида се употребљава и за обраду препознатих неисправности пи извршавању програма
  - на пример, дељење нулом и сл.
- Неки процесори праве разлику између прекида и изузетака (*Intel x86*) а неки не (*MIPS*, *PowerPC*)

Универзитет у Београду - Математички факултет

Начин позивања опслужилаца

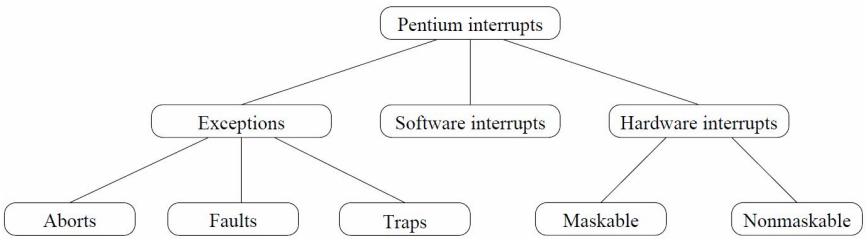
- Остварује се на два основна начина:
  - Векторски прекиди
    - сваком типу прекида је додељена одговарајућа адреса у меморији
    - у случају наступања прекида, извршавање се наставља од те адресе
    - *Intel x86*, *PowerPC*
  - Прекиди са узроком
    - прекиди наступају уз означавање узрока прекида
    - сви прекиди се обрађују истим опслужиоцем прекида
    - опслужилац проверава регистар узрока и на основу његове вредности препознаје врсту прекида и преноси контролу одговарајућем коду ОС-а
    - *MIPS*

Универзитет у Београду - Математички факултет

Пример *Intel x86*

- Систем прекида ће бити описан на примеру фамилије процесора *Intel x86*
- Затим ће бити представљене неке алтернативе на примеру процесора *MIPS*, *PowerPC*

Врсте прекида проц. *Intel x86*



Литература

- *Sivarama Dandamudi*, ***Fundamentals of Computer Organization and Design***, *Springer, 2002.*
- *Michal Ludvig*, ***Intel 80386 Programer's Reference Manual***, *http://www.logix.cz/michal/doc/i386*
- *Andrew Tanenbaum*, ***Архитектура и организација рачунара***, *Микро књига, 2007.*
- *Ненад Митић*, ***Основи рачунарских система***, *Математички факултет, 2002.*