

Увод у организацију и архитектуру рачунара 2

Александар Картељ

kartelj@matf.bg.ac.rs

Напомена: садржај ових слајдова је преузет од проф. Саше Малкова

Улазно/излазни уређаји

Основни концепти

Улазно/излазни уређаји

- Рачунарски систем обично има више различитих улазних и излазних уређаја
 - називају се и *периферни* уређаји, зато што се налазе на периферији рачунарског система
- Обезбеђују две основне функције
 - комуникацију рачунарског система са спољашњим светом и
 - чување података

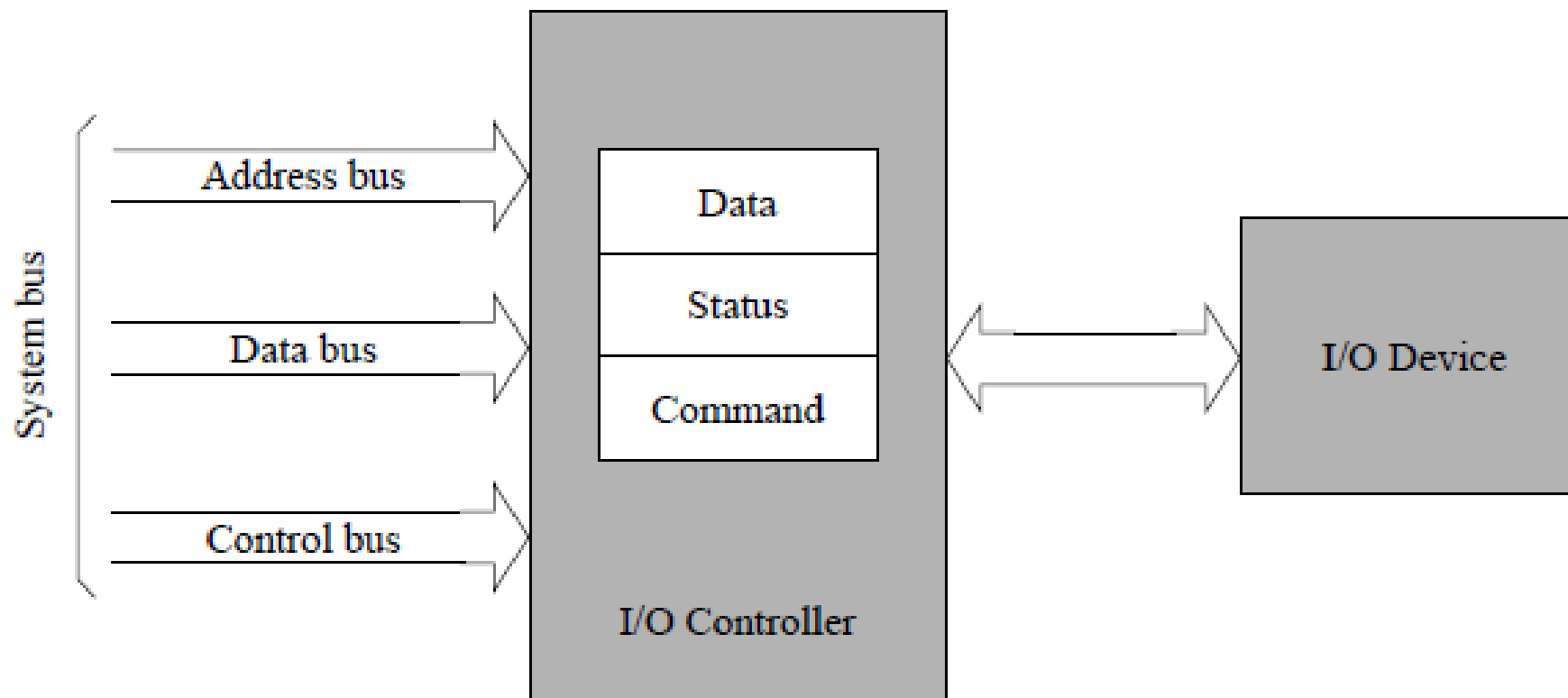
Принципи рада

- Независно од врсте уређаја, принципи рада улазно/излазних уређаја су исти:
 - сви У/И уређаји се на системску магистралу повезују посредством одговарајућег У/И контролера
 - први разлог је у различитости уређаја
 - различити уређаји имају различите протоколе комуникације
 - уместо да процесор и системска магистрала “уче” како да комуницирају са различитим врстама уређаја, те специфичности се препуштају контролерима
 - контролери имају улогу *моста* између центра и периферије
 - други је у техничким ограничењима
 - магистрала ради на високим фреквенцијама
 - да се не би сувише грејала, ради под врло ниским напонима
 - низак напон и висока фреквенција могу да функционишу без сметњи само на врло кратким растојањима (неколико *см*)
 - УИ уређаји захтевају дуже каблове, јачи напон и нижу фреквенцију

У/И контролери

- Процесор се никада не обрађа непосредно уређајима, већ само одговарајућим контролерима
- У/И контролери имају улогу *моста* између центра (процесор, меморија и системска магистрала) и периферије (периферни уређаји рачунарског система)
- Контролери уобичајено имају три врсте интерних регистара:
 - регистар података
 - командни регистар
 - статусни регистар

Блок-дијаграм уопштеног У/И контролера



Улазно/излазни уређаји

Начини употребе, пример рада

Употреба У/И уређаја

- Два основна начина употребе У/И уређаја су
 - пресликавање портова у меморију
 - изоловани У/И

Пресликавање портова у меморију

- Сви У/И портови се пресликавају у меморијски адресни простор
- Није потребан никакав посебан интерфејс процесора
- Процесор користи уређај као меморију
- Сваки процесор може употребљавати уређаје путем пресликавања портова у меморију
- Неки процесори подржавају само овакав начин рада:
 - *PowerPC, MIPS*

Изоловани У/И

- Изоловани У/И подразумева посебан У/И адресни простор, независан од меморијског адресног простора
- *Intel x86* процесори подржавају овакав начин рада
- Системи засновани на процесорима који подржавају изоловани У/И омогућавају избор метода
 - нпр: штампачи се користе путем изолованог У/И, а графички подсистем путем пресликавања у меморију

Пример – тастатура

- Контролер тастатуре испитује тастатуру и извештава о притискању и отпуштању тастера у виду тзв. *кодова тастера* (енгл. *scan code*)
 - код тастера је идентификациони број који се додељује тастеру на основу његове локације
 - улазни потпрограми преводе код тастера у одговарајуће кодове карактера

Пример – тастатура (2)

- Као контролер се (на пример) може користити чип са следећим карактеристикама:
 - чип располаже са три 8-битна регистра (*PA*, *PB*, *PC*)
 - контролер испоручује код тастера путем регистра *PA*
 - битови *PA0-PA6* чине код тастера
 - бит *PA7* означава притисак (0) или отпуштање (1) тастера
 - нека су, нпр., ови регистри адресирани од адресе *60H*:

8255 register	Port address
<i>PA</i> (input port)	60H
<i>PB</i> (output port)	61H
<i>PC</i> (input port)	62H
Command register	63H

Пренос података

- Пренос података између “система” и УИ уређаја подразумева размену података између меморије (или регистара) и УИ уређаја
- Пренос података чине две фазе:
 - фаза преноса података
 - фаза обавештавања о крају
- Неки аутори додају као посебну фазу
 - иницијализацију преноса података

Фаза преноса података

- Током ове фазе се преносе подаци између меморије и УИ уређаја
- Пренос се остварује путем
 - *програмираног У/И* или
 - *непосредног приступа меморији* (енгл. *direct memory access - DMA*)

Фаза обавештавања о крају

- Током ове фазе процесор се информише да је пренос података довршен
- Информисање се остварује путем
 - *система прекида* или
 - *програмираног У/И*

Улазно/излазни уређаји

Програмирани УИ и DMA

Пренос података (2)

- Три значајне технике:
 - програмирани У/И
 - непосредан приступ меморији (*DMA*)
 - система прекида
- Уобичајено:
 - ако се подаци преносе путем *DMA*, онда се обавештавање о крају одвија путем система прекида
 - ако се подаци преносе путем програмираног У/И, онда се и обавештавање о крају остварује истим путем

Илустрација техника

- На примеру радника и шефа:
 - програмирани У/И
 - шеф задаје посао
 - “непрестано” проверава да ли је посао довршен
 - када је довршен, шеф узима резултат рада и ради даље
 - непосредан приступ меморији (*DMA*) / систем прекида
 - шеф задаје посао раднику и наставља са својим послом
 - радник самостално обавља посао
 - када заврши посао, радник обавештава шефа о завршетку
 - шеф реагује на обавештење и затим наставља са својим послом

Програмирани У/И

- У основи програмираног У/И је петља у којој се чека да се доврши задати посао да би се могло наставити са радом
 - чекање се изводи кроз вишеструко проверавање (узорковање, тестирање) да ли је уређај достигао очекивано стање

Програмирани У/И – пример

- Пример тастатуре:
 - читава се регистар *РА*
 - ако бит 7 има вредност 1, значи да није притиснут тастер, па се понавља претходни корак
 - ако бит 7 има вредност 0, значи да је притиснут тастер и прекида се петља
 - затим се прочитани код тастера може даље употребљавати, тј. преводити у код карактера

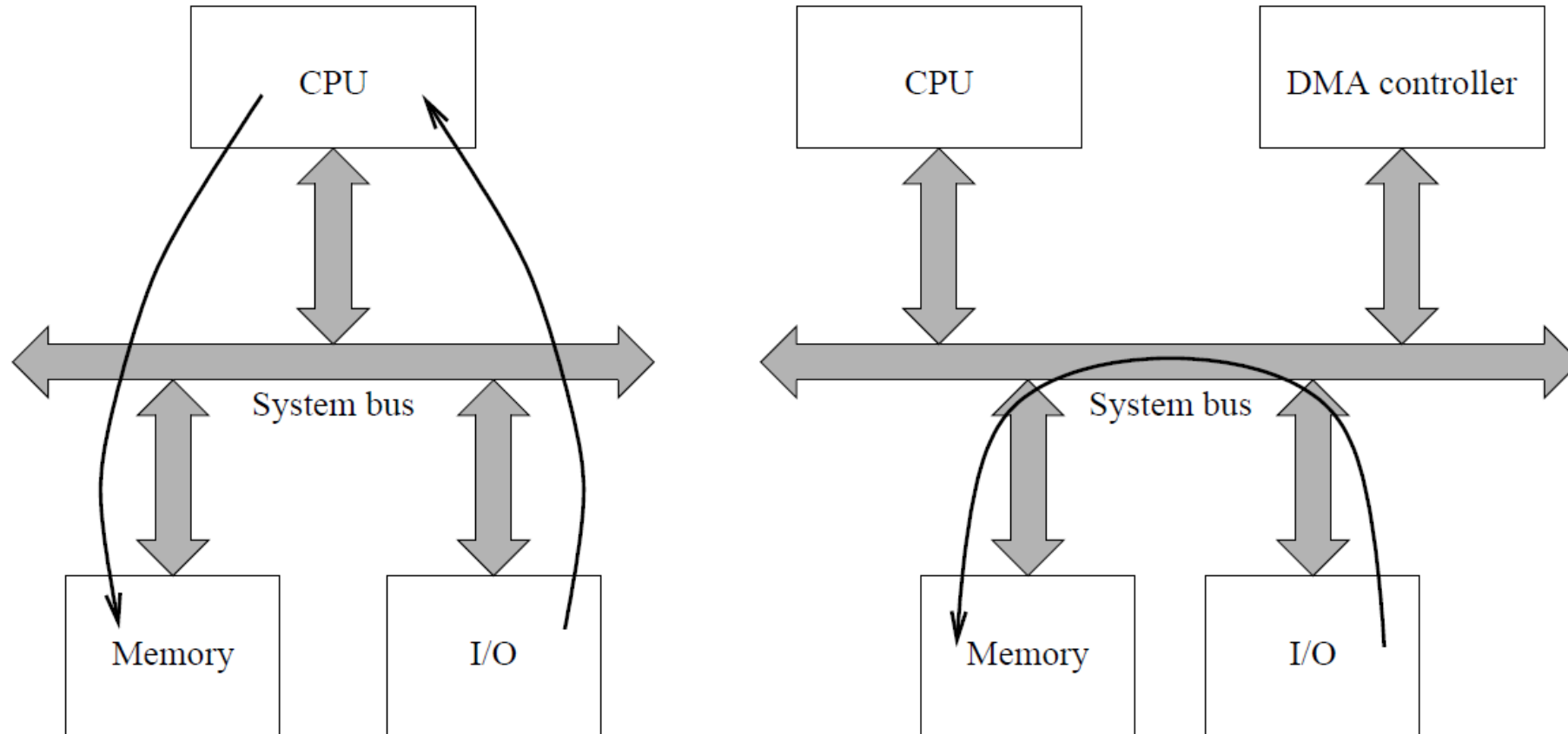
DMA

- 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

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*

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

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

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

Улазно/излазни уређаји

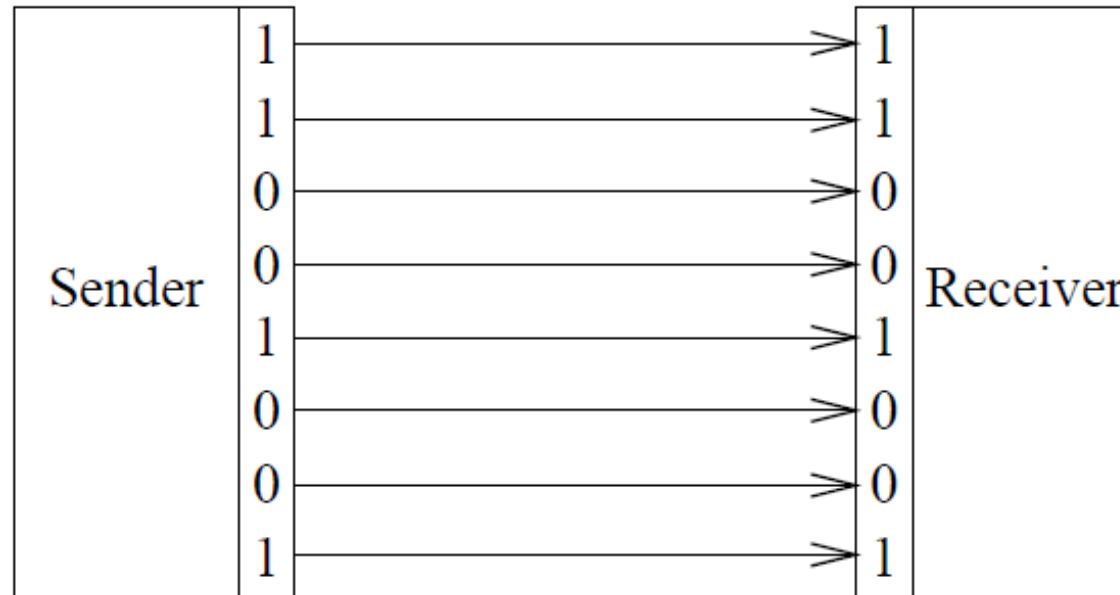
Серијски и паралелни пренос

Врсте техника преноса података

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

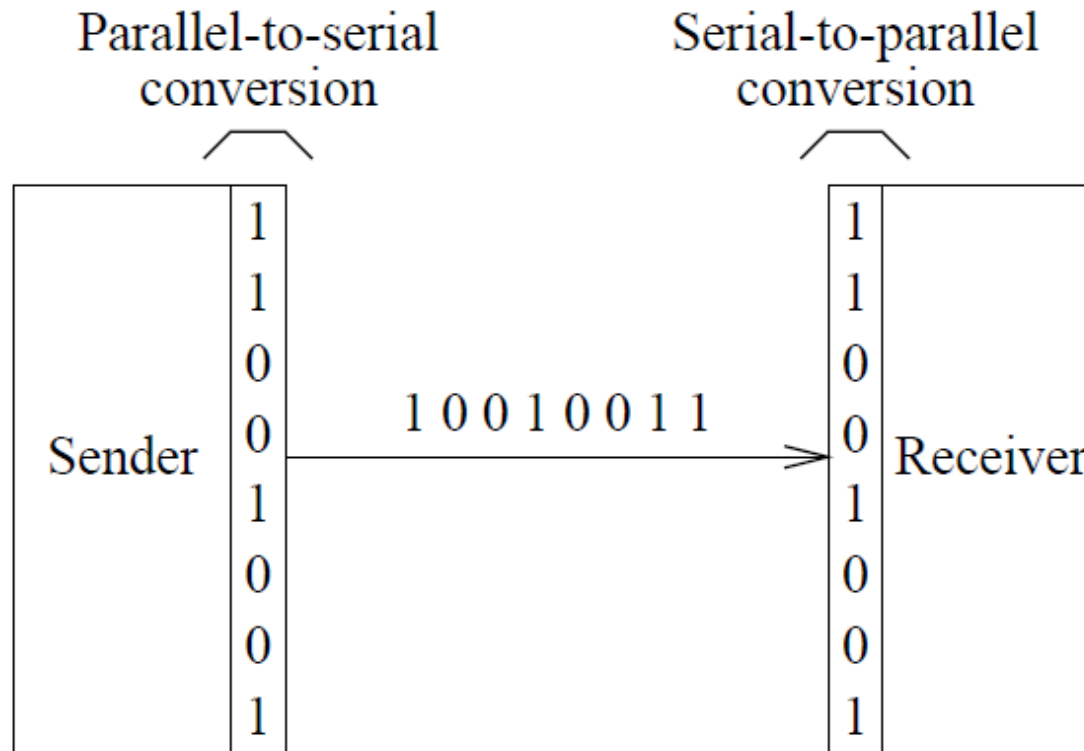
Паралелан пренос података

- Неколико битова се пренести истовремено кроз паралелне водове



Серијски пренос података

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



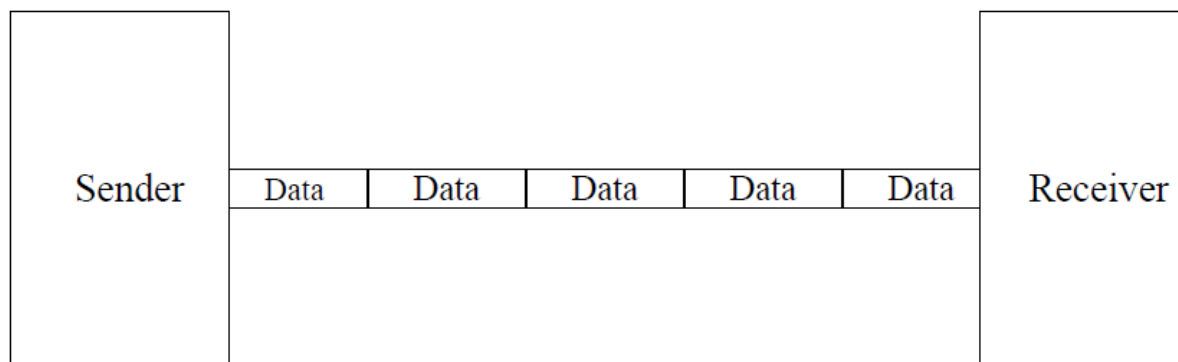
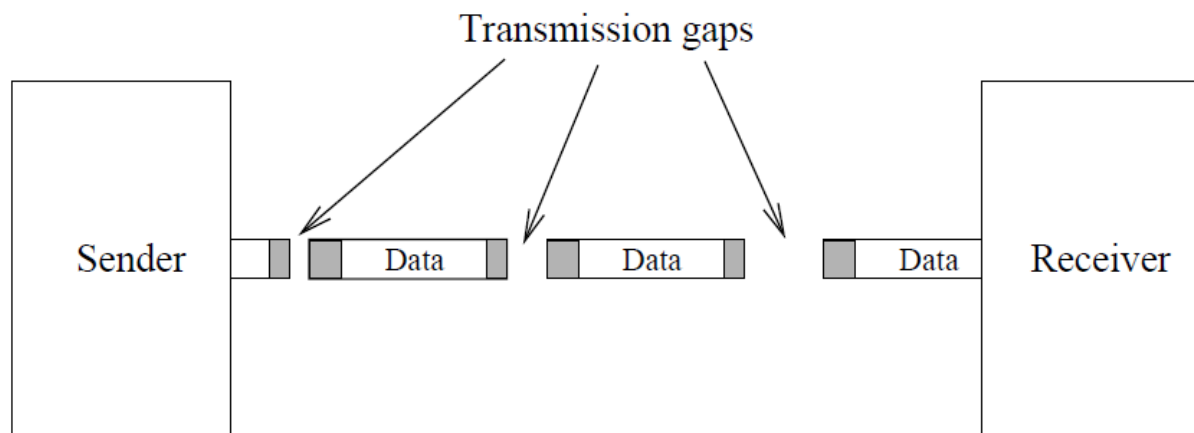
Однос паралелног и серијског преноса

- 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

Серијски пренос података

- Може да буде
 - синхрони
 - часовници пошиљаоца и примаоца су синхронизовани 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

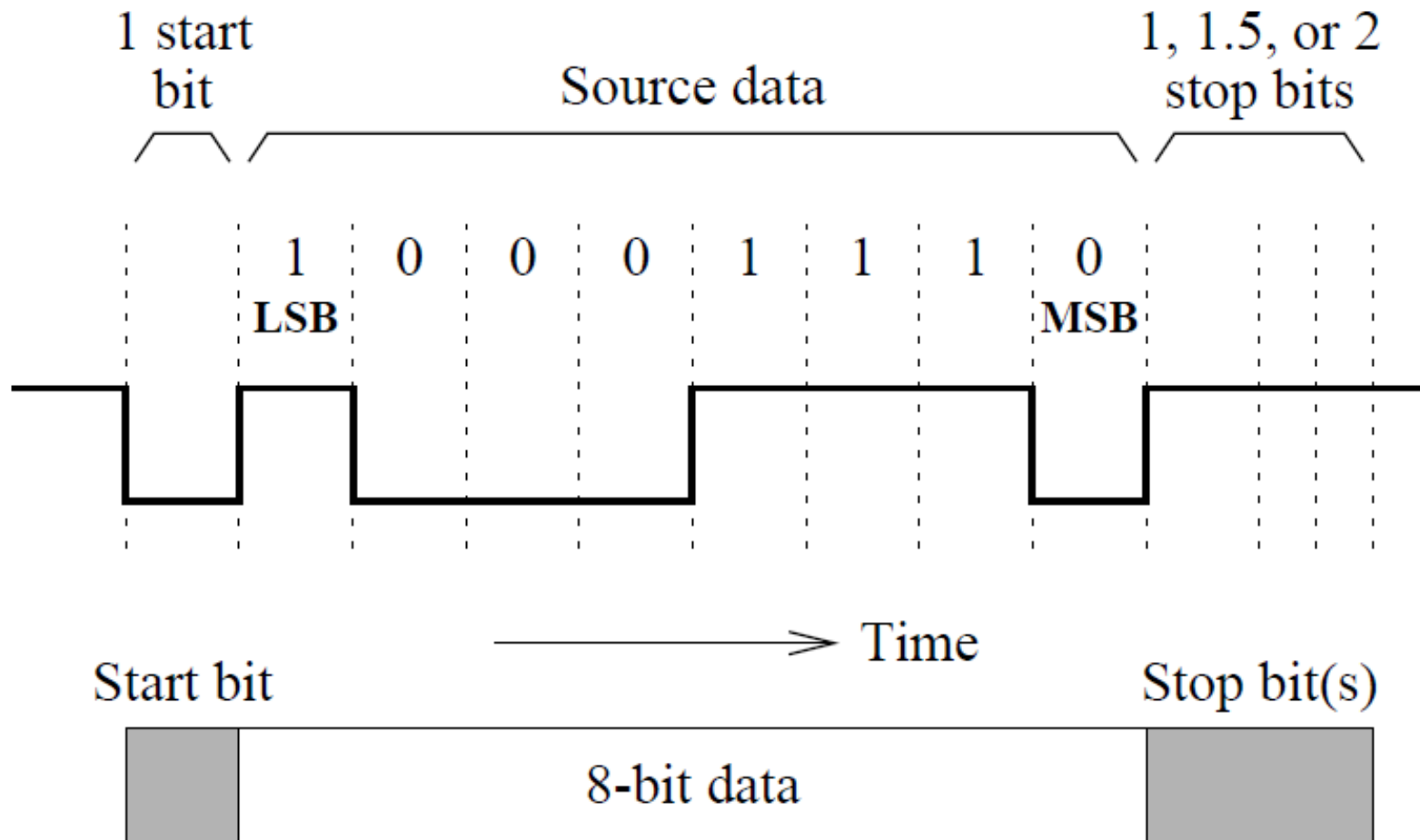
Асинхрони и синхрони пренос података



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)
- Зауоставни 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 зауоставна 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 (2)

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

Universal Serial Bus – USB

- Серијски пренос
- Изворно развијен 1995.
- 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.
 - 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
- Превазилажење проблема са ресурсима система
 - додавање нових уређаја доноси проблеме разрешавања конфликта 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

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
- Проширивост помоћу *хабова*
 - на једно прикључно место се може повезати хаб (разделник) ради повећавања броја прикључака или постављања прикључака на жељено место
- Уштеда енергије
 - *USB* уређаји аутоматски улазе у примирено (суспендовано) стање ако нема активности на магистрали 3ms
- Препознавање и отклањање грешака
 - користи се алгоритам *CRC* за препознавање грешака у комуникацији

USB – Пренос података

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

Схема енкодовања *NRZ*

(nonreturn to zero)

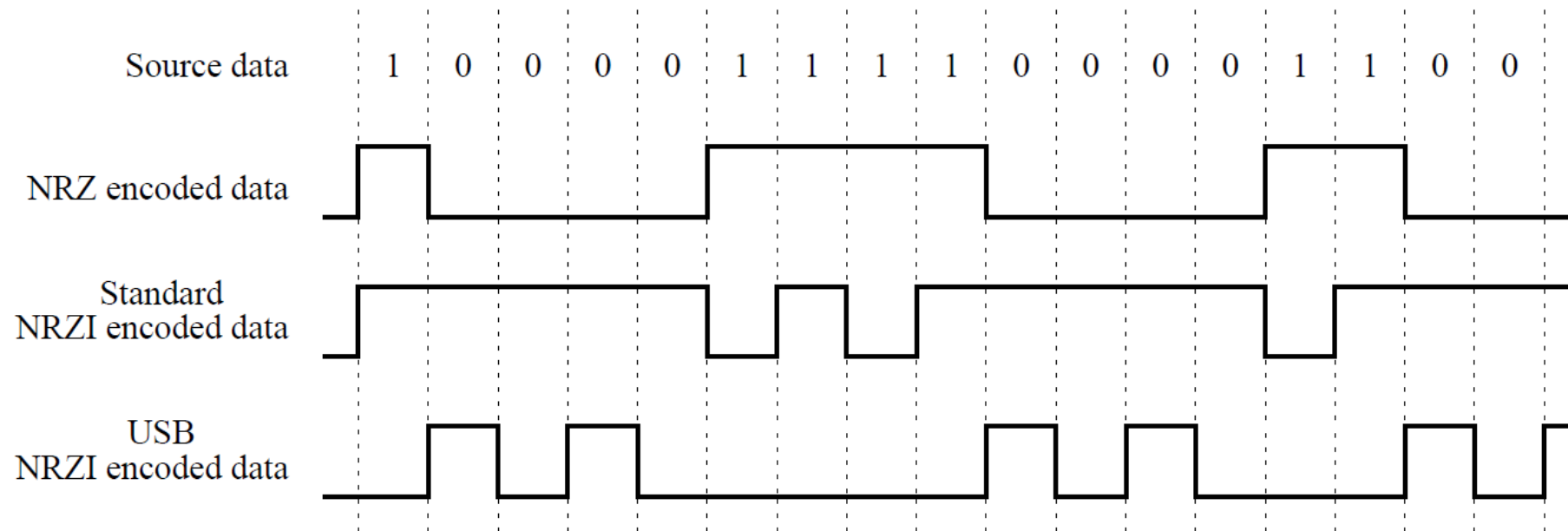
- 0 је низак а 1 висок ниво сигнала
- Проблеми:
 - Нема промена сигнала ако се преносе дуги низови нула или јединица
 - промене су важне за примаоца због синхонизације и препознавања података
 - У случају шумова, тешко је препознати нивое 0 и 1, али се промене ипак лако препознају

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

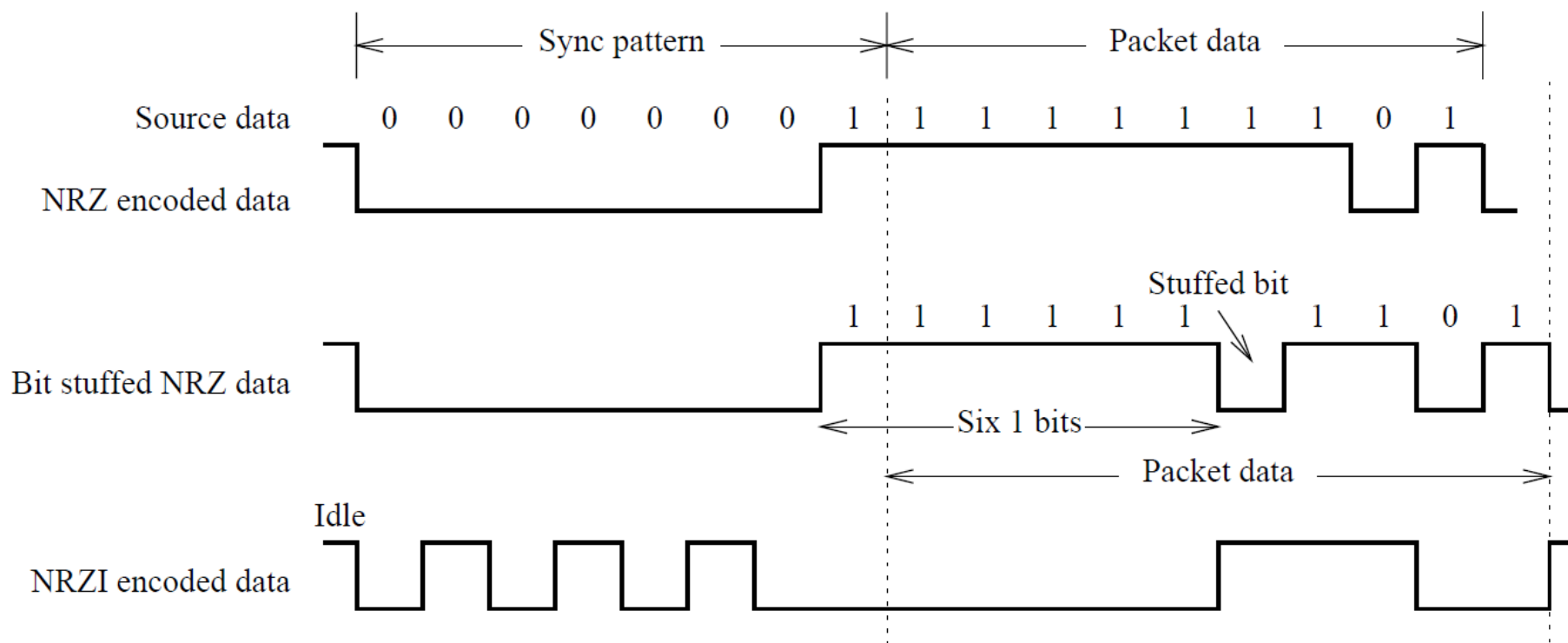
NRZI



Уметање битова

- Описана схема енковања *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

Уметање битова (2)



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 *USB* повезивања

