Увод у организацију и архитектуру рачунара 1

Александар Картељ

kartelj@matf.bg.ac.rs

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

Улазно излазни уређаји

Преглед, подела и карактеристике

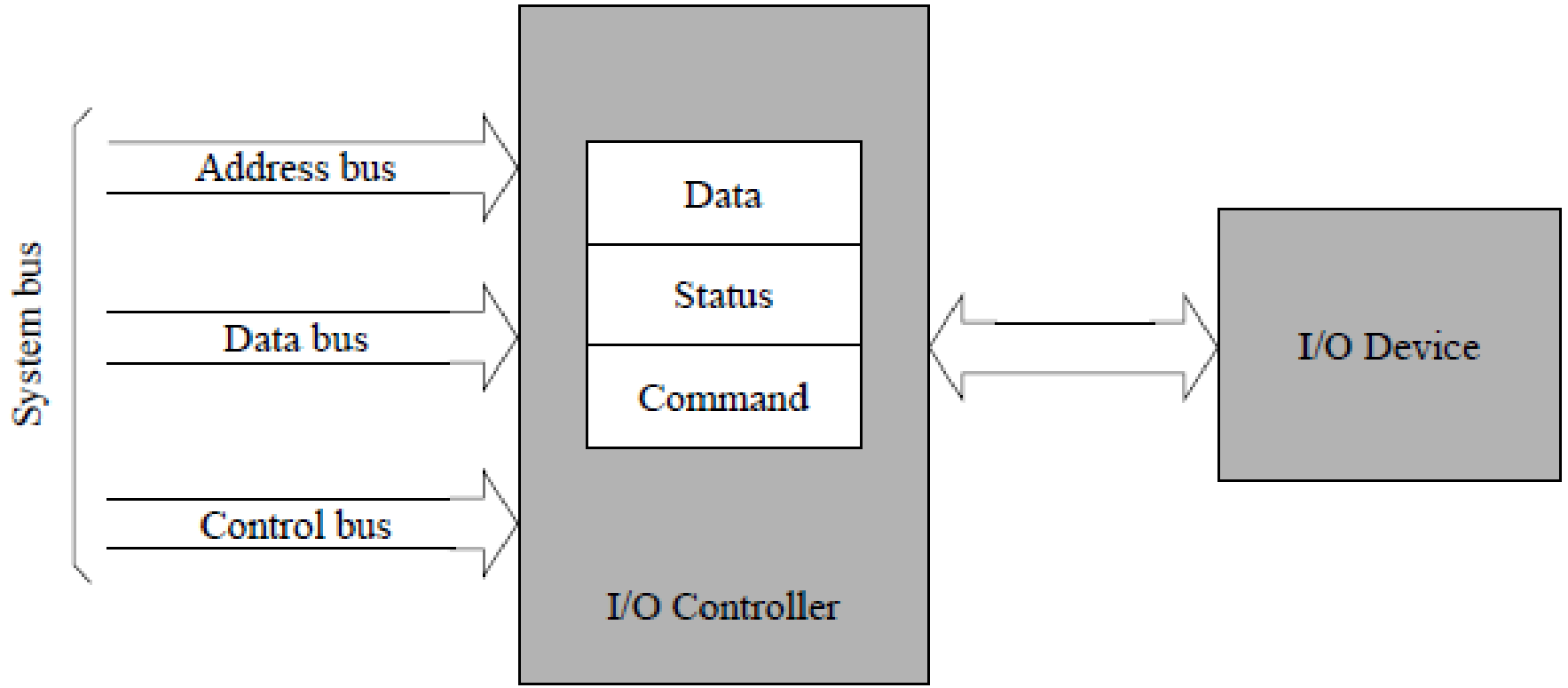
Улазно/излазни уређаји

- 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

У/И контролери

- Процесор се никада не обраћа непосредно уређајима, већ само одговарајућим контролерима
  - Главни разлог је различитост уређаја: У/И контролери су дизајнирани слично за различите уређаје, па процесор не мора да учи „превише“
- 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



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
  - код тастера нема никакве везе са *ASCII* или неким другим кодовима карактера, већ улазни потпрограми преводе код тастера у одговарајуће кодове карактера
  - код послењег притиснутог/отпуштеног тастера се записује у неком унутрашњем регистру контролера и касније испоручује „систему“

Пренос података

- Пренос података између “система” и УИ уређаја подразумева размену података између:
  - меморије или регистара унутар процесора
  - и УИ уређаја

- Пренос података чине две фазе:
  - фаза преноса података
  - фаза обавештавања о крају

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 (ово ћемо прескочити)*
  - или *програмираног У/И*

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

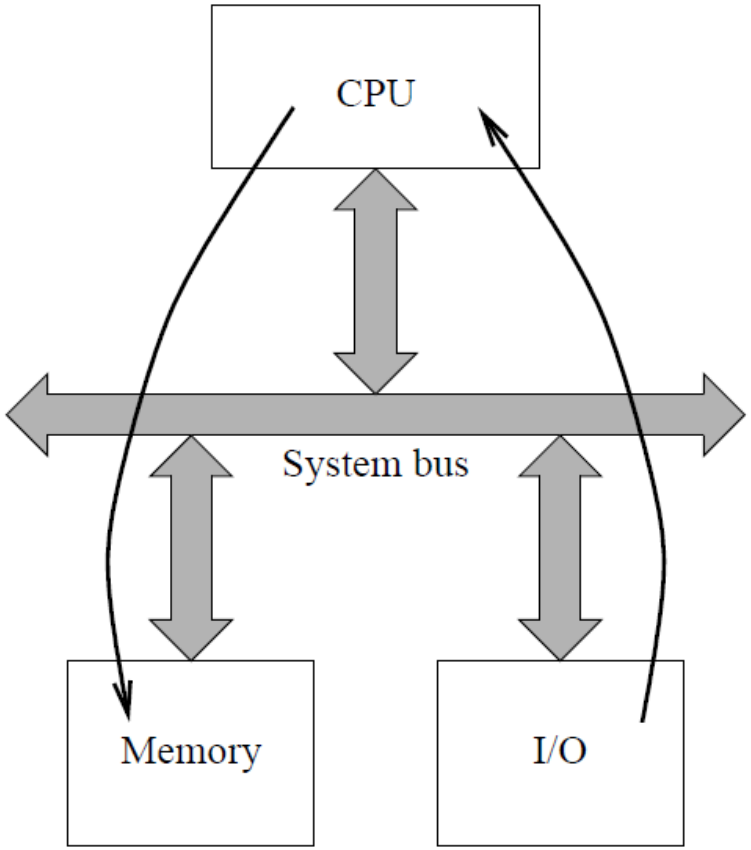
DMA

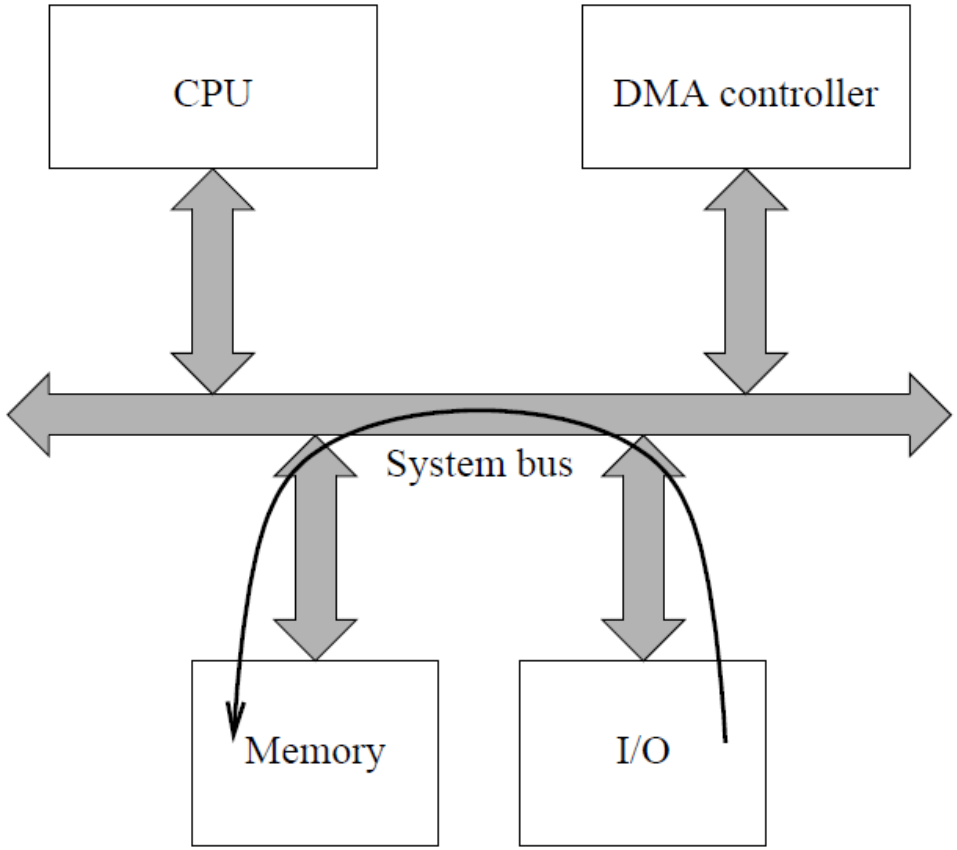
- У случају уређаја који преносе веће количине података
  или раде временски захтевне послове,
  програмирани У/И води великом утрошку времена на чекање
- *DMA* има за циљ да се процесор ослободи старања
  о преносу података и посвети другим стварима

DMA (2)

- *DMA* се имплементира помоћу контролера *DMA*
  - 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
- DMA 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 DMA контролер.
- DMA контролер започиње операцију преноса:
  - најпре 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

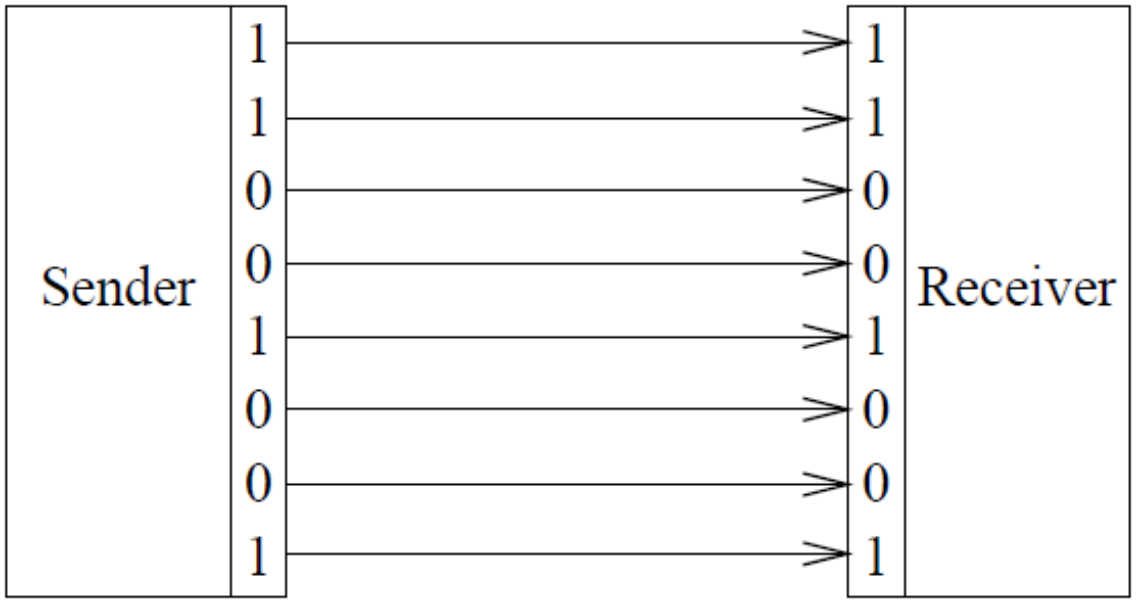
- након довршеног преноса обавештава се процесор
- уобичајено се за то користи систем прекида
- процесор затим проверава стање преноса

Врсте техника преноса података

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

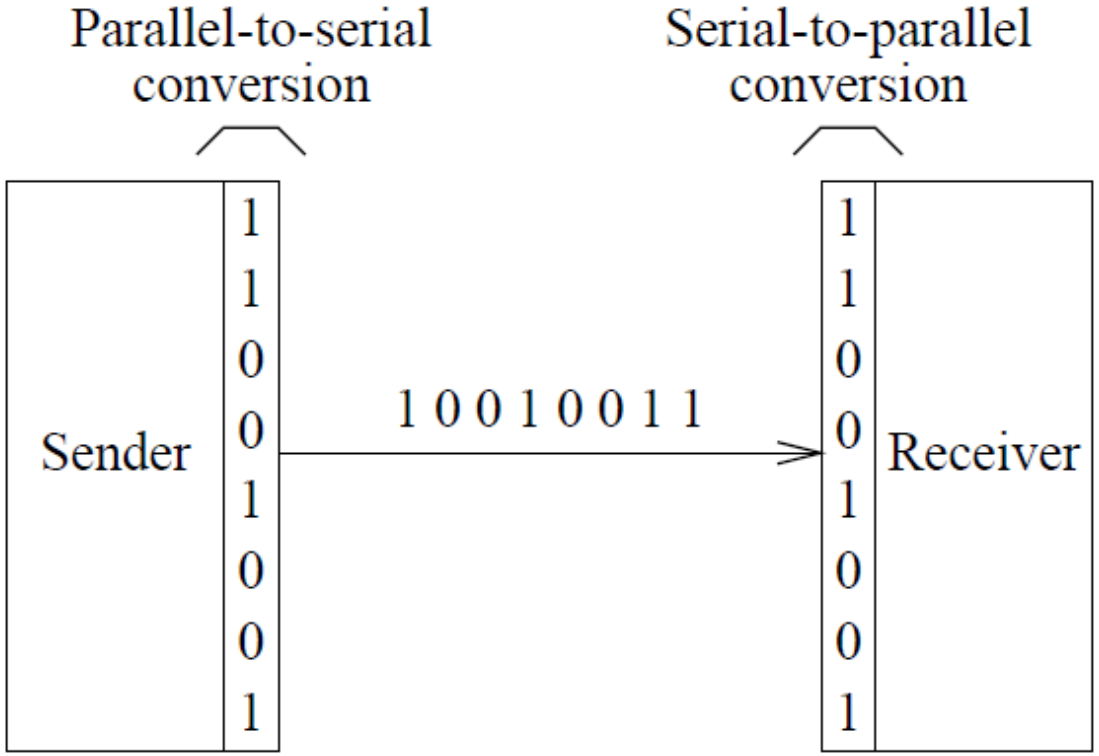
Паралелан пренос података

- Неколико битова се преноси истовремено кроз паралелне водове



Серијски пренос података

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



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
  - јефтинији
  - нема могућности искривљења података

Universal Serial Bus – USB (пример серијског преноса)

- 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.
    - разрешени неки проблеми са протоколом
    - први широко распрострањен стандард
  - 2.0 из 2000.
    - *висока* брзина, до 480 Mbps
  - 3.0 спецификација објављена 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
- 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
  - повезивање *USB* уређаја се може обављати без искључивања рачунара

USB – Додатне напредне особине

- Напајање уређаја кроз интерфејс
  - кабл за повезивање има линије напајања
- Двосмерна контрола уређаја
  - подаци могу да теку у оба смера
- Проширивост помоћу *хабова*
  - на једно прикључно место се може повезати хаб (разделник) ради повећавање броја прикључака

USB – Пренос података

- Кабл има четири линије:
  - две служе за напајање уређаја
  - две служе за преношење сигнала

- За преношење података се употребљава схема енкодовања *NRZI* (*nonreturn to zero-inverted*)

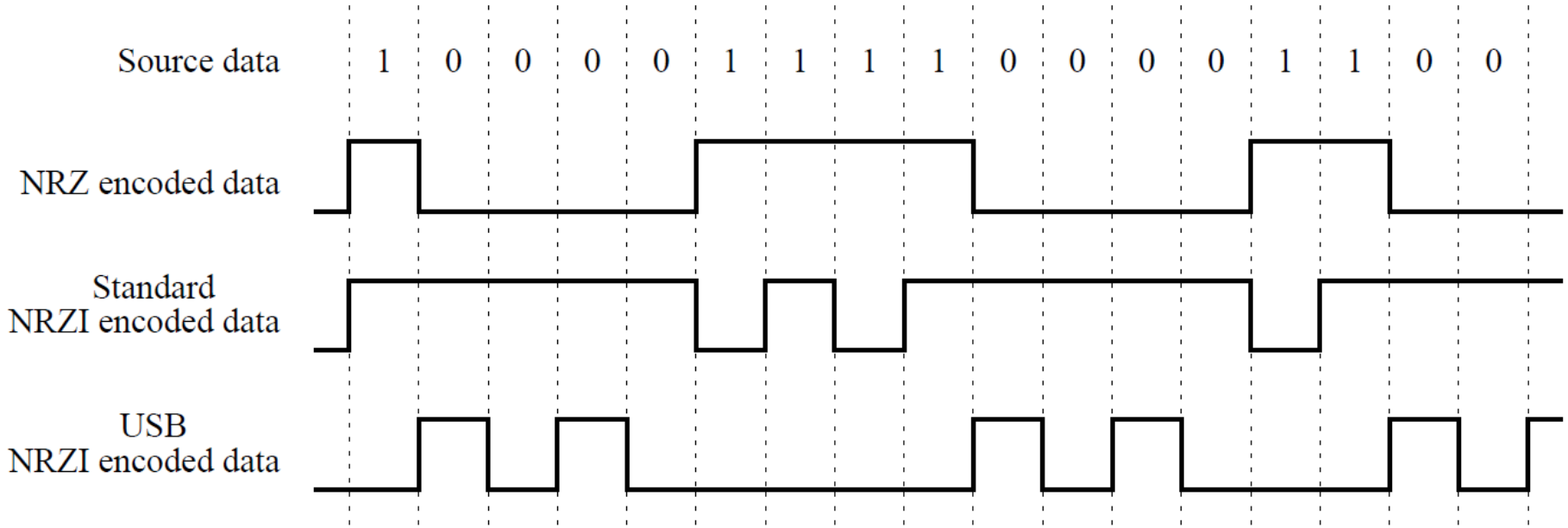
Схема енкодовања *NRZ*
(*nonreturn to zero*)

- 0 је низак а 1 висок ниво сигнала
- Проблеми:
  - Нема промена сигнала ако се преносе дуги низови 0 или 1
    - 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

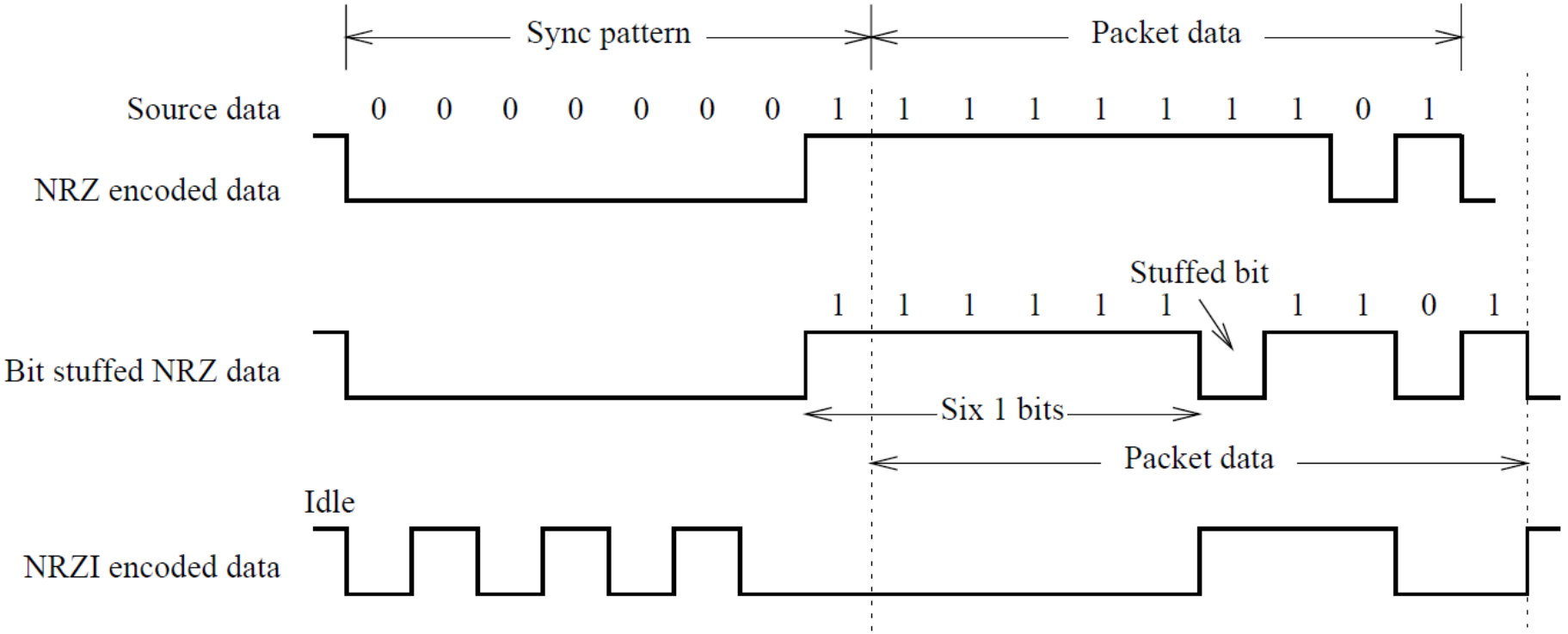
NRZI


Source data
1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 0 0
NRZ encoded data
Standard
NRZI encoded data
USB
NRZI encoded data

Уметање битова

- Описана схема енкодовања USB NRZI решава неке од проблема:
  - ниво сигнала не игра главну улогу
    - посматрају се само транзиције
  - отклањају се дугачки низови непромењених стања у случају низова нула 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)



USB повезивање

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

USB повезивање – дрвенаста структура