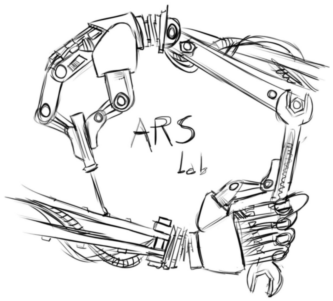
Il Sistema di Input/Output

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it



Architettura degli Elaboratori

Il Sistema di Input/Ouput

- Una caratteristica fondamentale di un computer è la possibilità di interagire con il mondo esterno (operatore umano, ambiente, etc.)
- Queste operazioni sono effettuate attraverso le **interfacce di Input/Output**
- Esse sono dispositivi hardware che mettono a disposizione opportune funzionalità e/o collegamenti elettrici con l'esterno
- Interagiscono con la CPU attraverso i collegamenti del **bus di sistema**

CPU e Interfacce di I/O

- Il modo con cui una CPU, ed il relativo software, vedono una interfaccia di I/O è attraverso l'uso di istruzioni che fanno riferimento a **indirizzi specifici**
- Esistono due modalità per l'accesso da parte delle istruzioni alle interfacce di I/O:
  - I/O mappato in memoria (spazio condiviso con la memoria di sistema)
  - Spazio di indirizzamento separato e uso di istruzioni ad-hoc

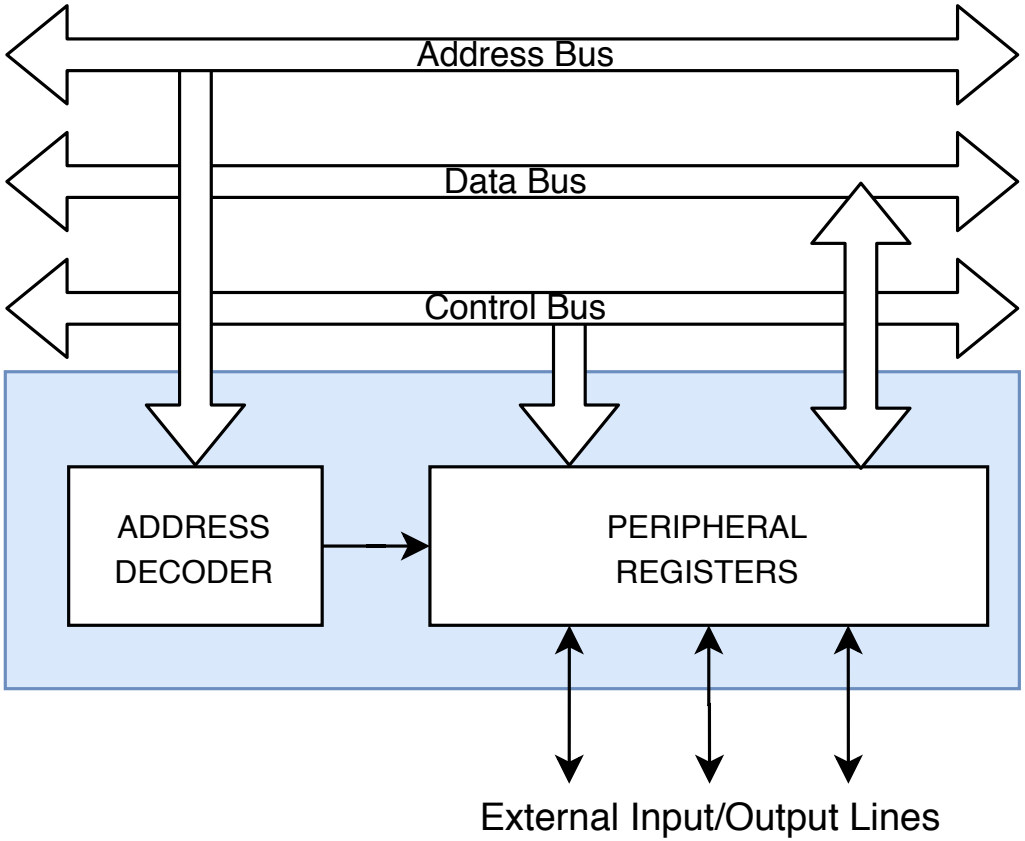
I/O Mappato in Memoria

- Una parte della memoria (stabilita a design-time) viene riservata per l'Input/Output
- Un set di indirizzi specifico dell'area di memoria è assegnato all'Input/Ouput
- Ogni interfaccia di I/O viene "allocata" (staticamente o dinamicamente) su uno o più indirizzi dell'area riservata
- L'accesso all'interfaccia è effettuato attraverso delle operazioni di LOAD/STORE che utilizzano l'indirizzo assegnato all'interfaccia stessa

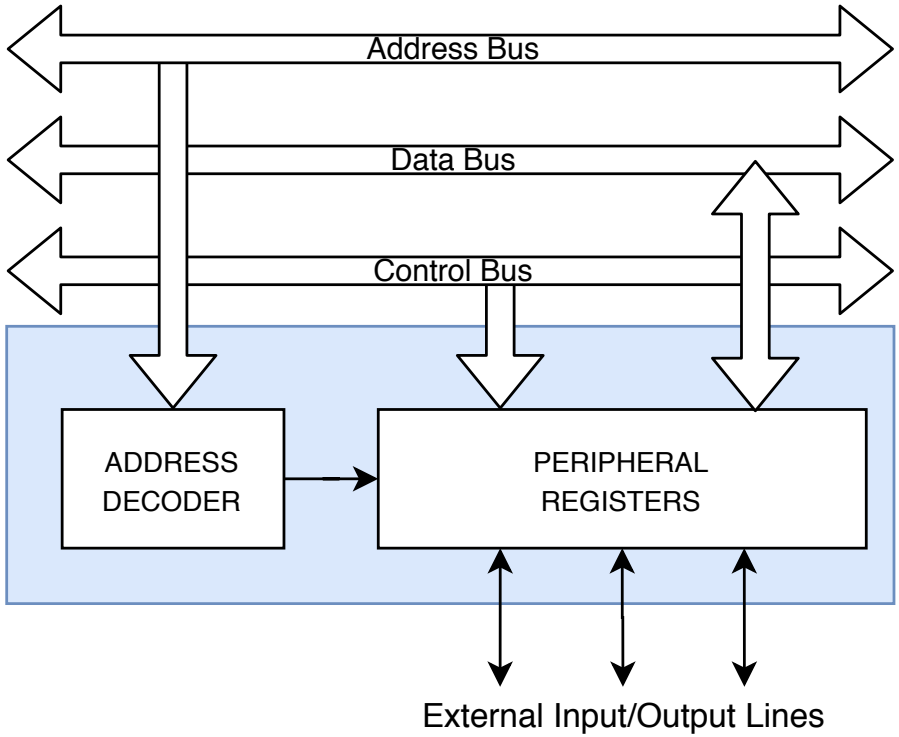
I/O su Spazio Separato

- Un set di indirizzi separato da quello della memoria centrale è assegnato all'Input/Output
- Ogni interfaccia di I/O viene "allocata" (staticamente o dinamicamente) su uno o più indirizzi di tale area di I/O
- L'accesso all'interfaccia è effettuato attraverso delle operazioni speciali di IN/OUT (differenti dalle LOAD/STORE) che utilizzano l'indirizzo assegnato all'interfaccia stessa

Bus di Sistema e Input/Output

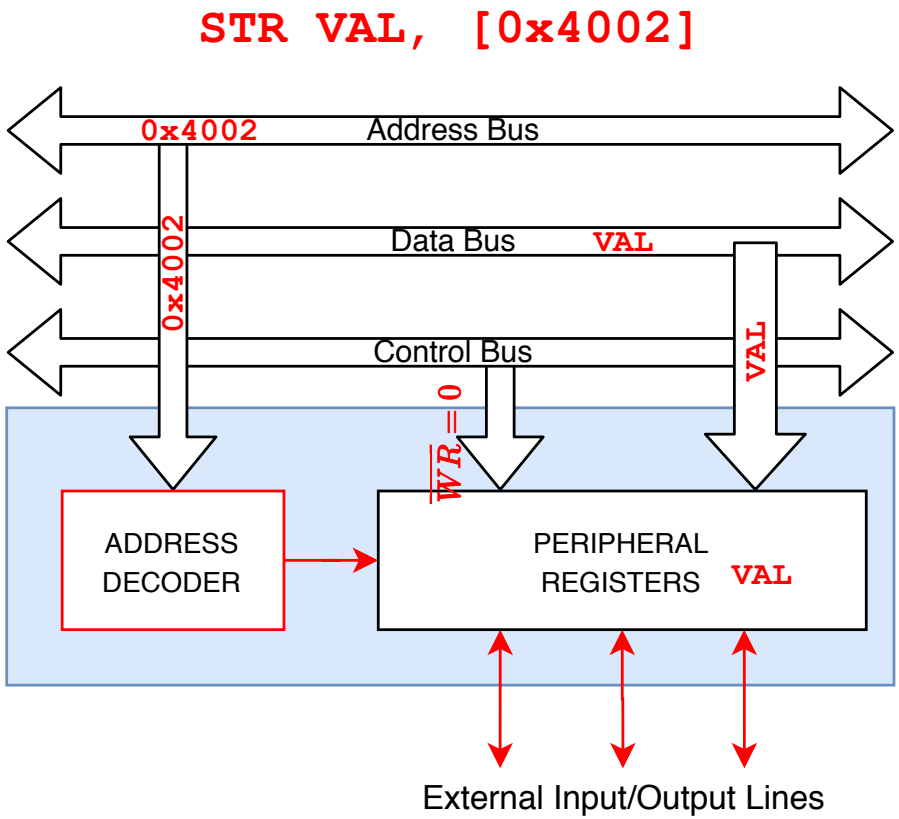


I/O Mappato in Memoria e Control Bus



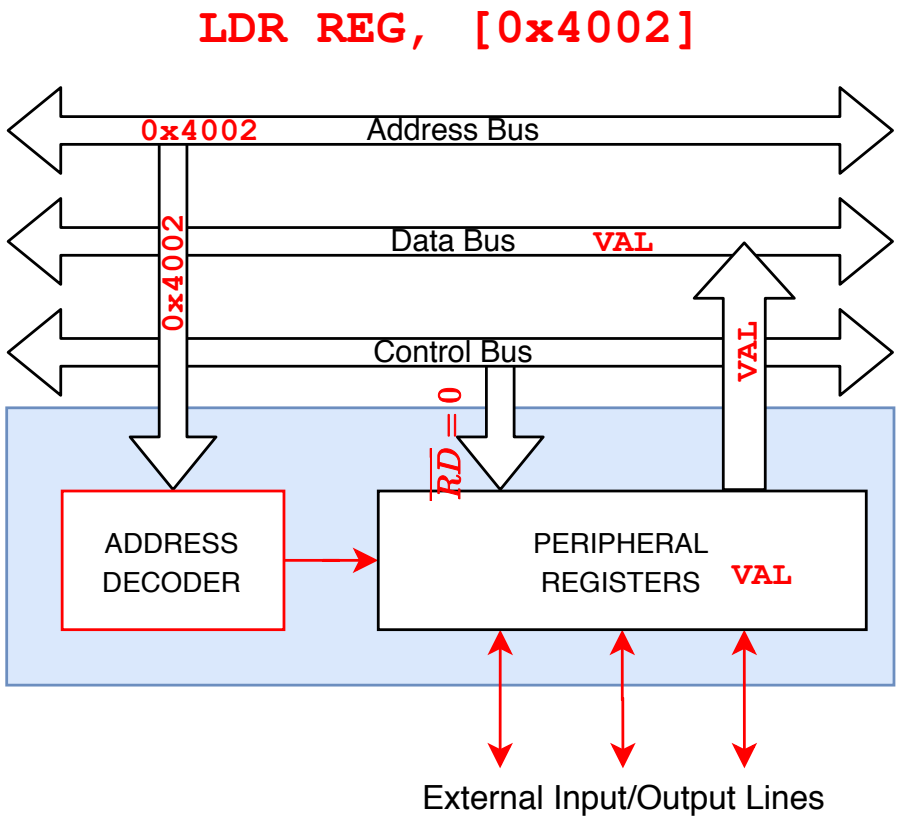
- Il Control Bus espone due linee che indicano l'operazione che si sta effettuando:
  - $\overline{RD}$, quando è 0 indica un'operazione di READ (o LOAD)
  - $\overline{WR}$, quando è 0 indica un'operazione di WRITE (o STORE)

Bus di Sistema e Input/Output



- La CPU pone l'indirizzo **0x4002** sull'Address Bus
- Il decoder della periferica lo riconosce
- La CPU pone il dato sul Data Bus
- La CPU "attiva" il segnale $\overline{WR}$
- L'hardware della periferica memorizza ("riceve") il dato

Bus di Sistema e Input/Output



- La CPU pone l'indirizzo **0x4002** sull'Address Bus
- Il decoder della periferica lo riconosce
- La CPU "attiva" il segnale $\overline{RD}$
- L'hardware della periferica invia il dato sul Data Bus
- La CPU acquisisce il dato

Bus di Sistema, Input/Output e Sincronizzazione

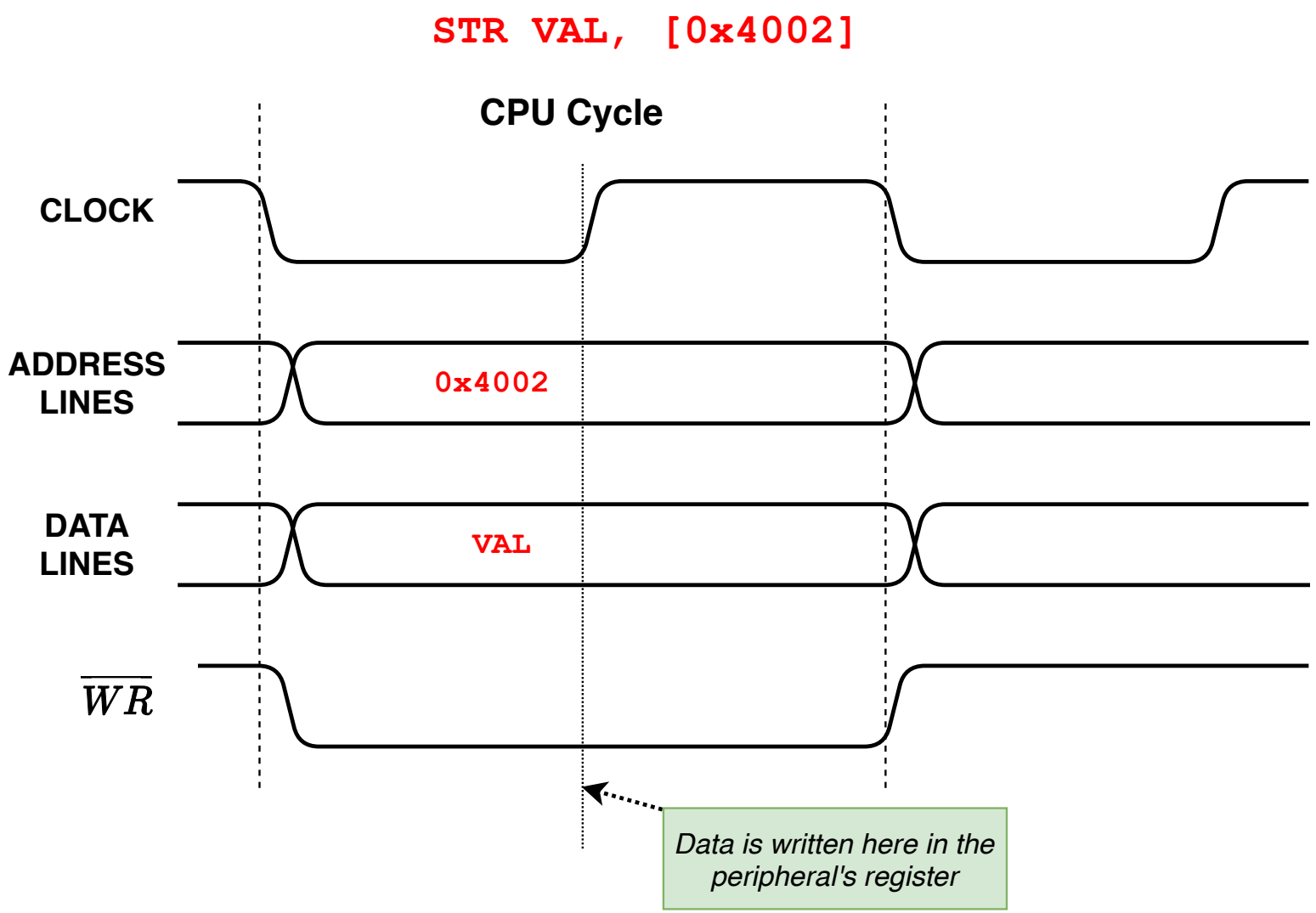
- Le operazioni tra CPU e periferica sono soggette ad uno scambio di segnali elettrici, in cui è importante sia l'aspetto logico (livello di tensione) che **temporale**
- Tali interazioni sono dunque regolate da un **protocollo** che specifica l'andamento tempolare dei segnali (cosa viene prima e dopo, quando accade un evento, quando accade l'altro evento, etc.)
- Il protocollo può essere di due tipi:
  - **sincrono**, è presente un segnale di clock e gli eventi avvengono solo durante i fronti
  - **asincrono**, non è presente un segnale di clock

BUS Sincrono

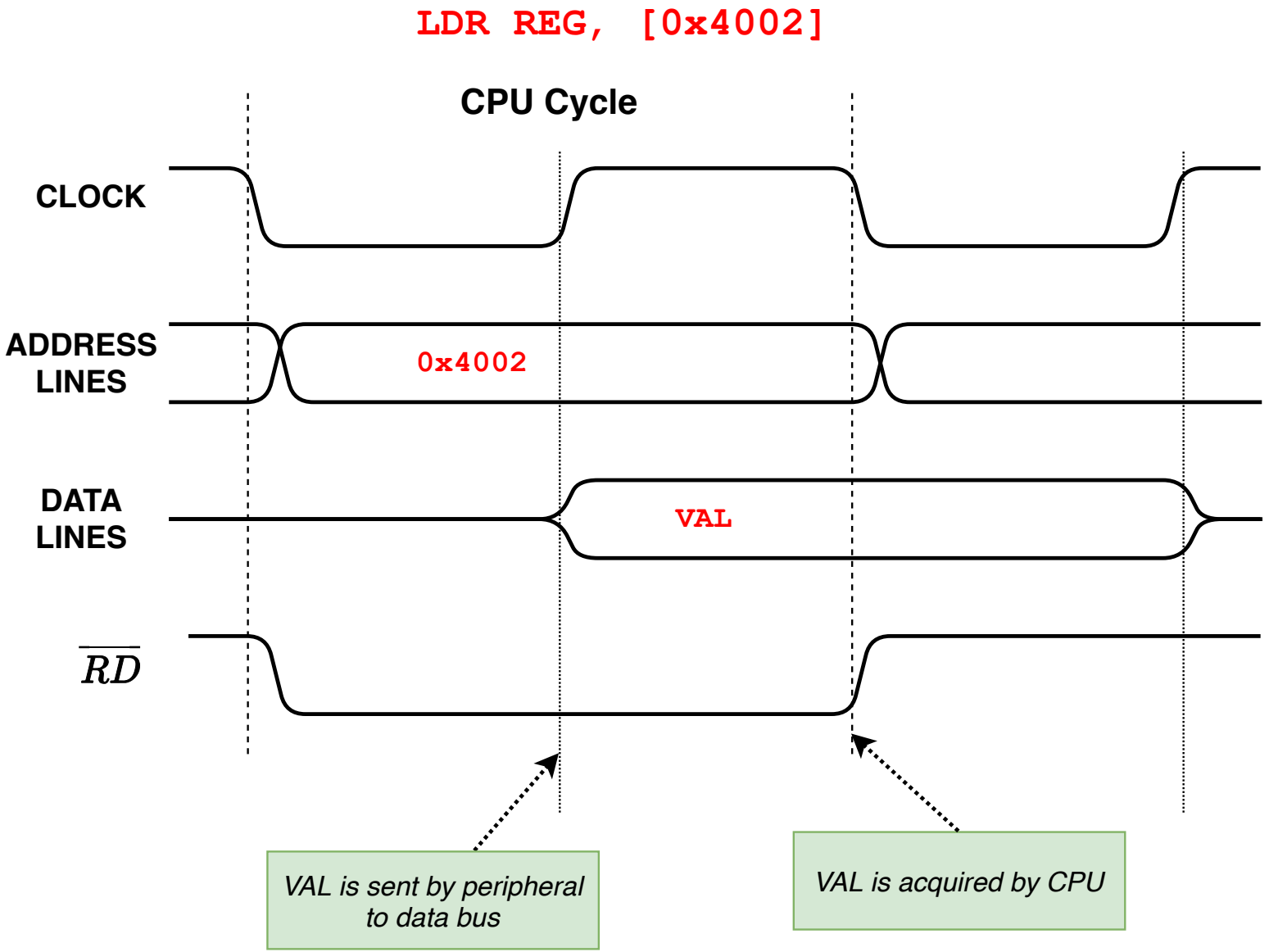
BUS Sincrono

- Nel Bus Sincrono le operazioni avvengono solo all'occorrenza dei fronti del **clock di sistema**
- In fase di progetto, si "assegna" uno dei due fronti alle operazioni della CPU e l'altro alla memoria/interfacce di I/O
- In questo modo è possibile garantire la sincronizzazione stretta nell'interazione tra CPU e sistema periferico

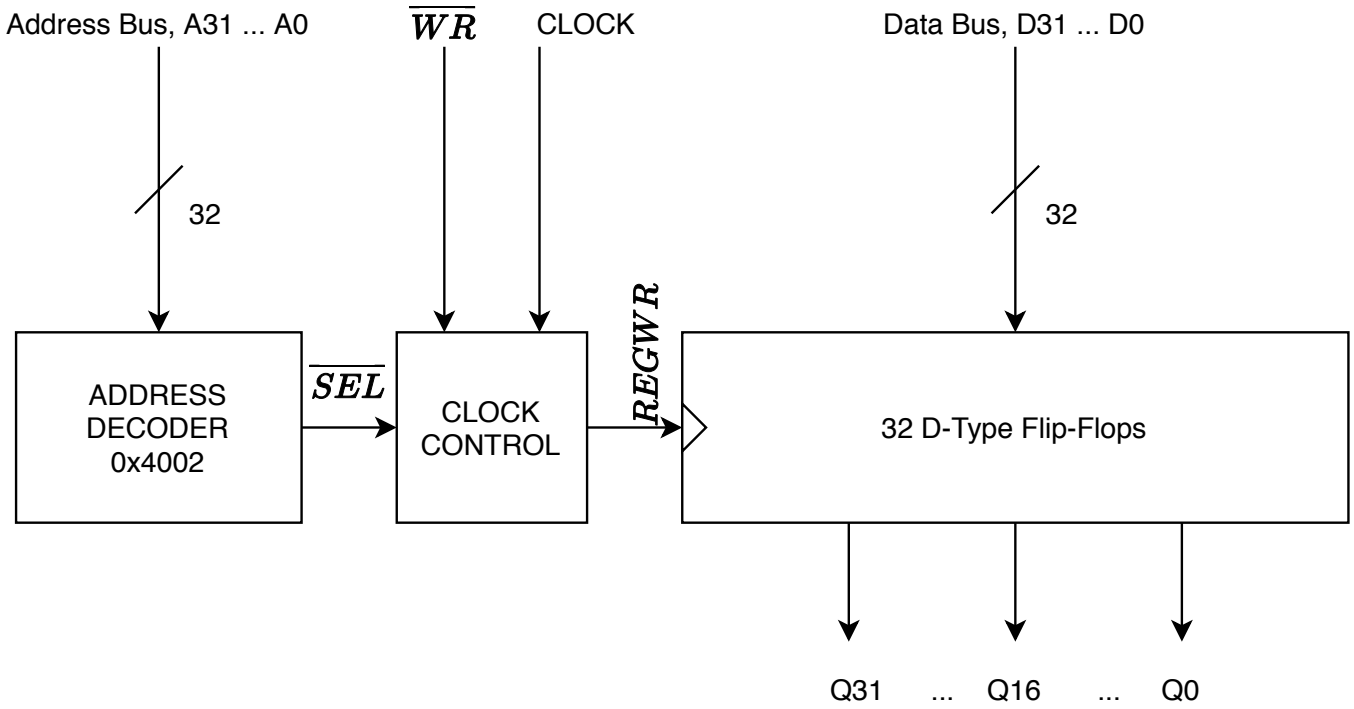
Bus Sincrono e Scrittura su I/O



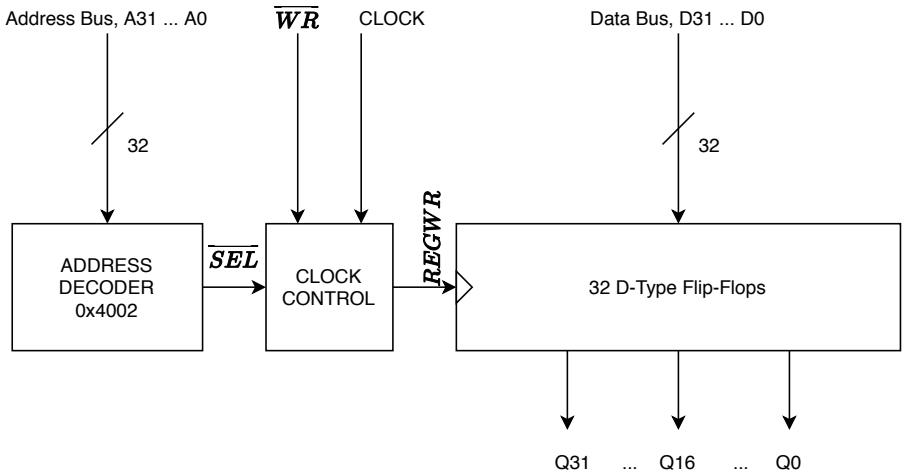
Bus Sincrono e Lettura da I/O



Esempio di Porta Digitale di Output



Esempio di Porta Digitale di Output

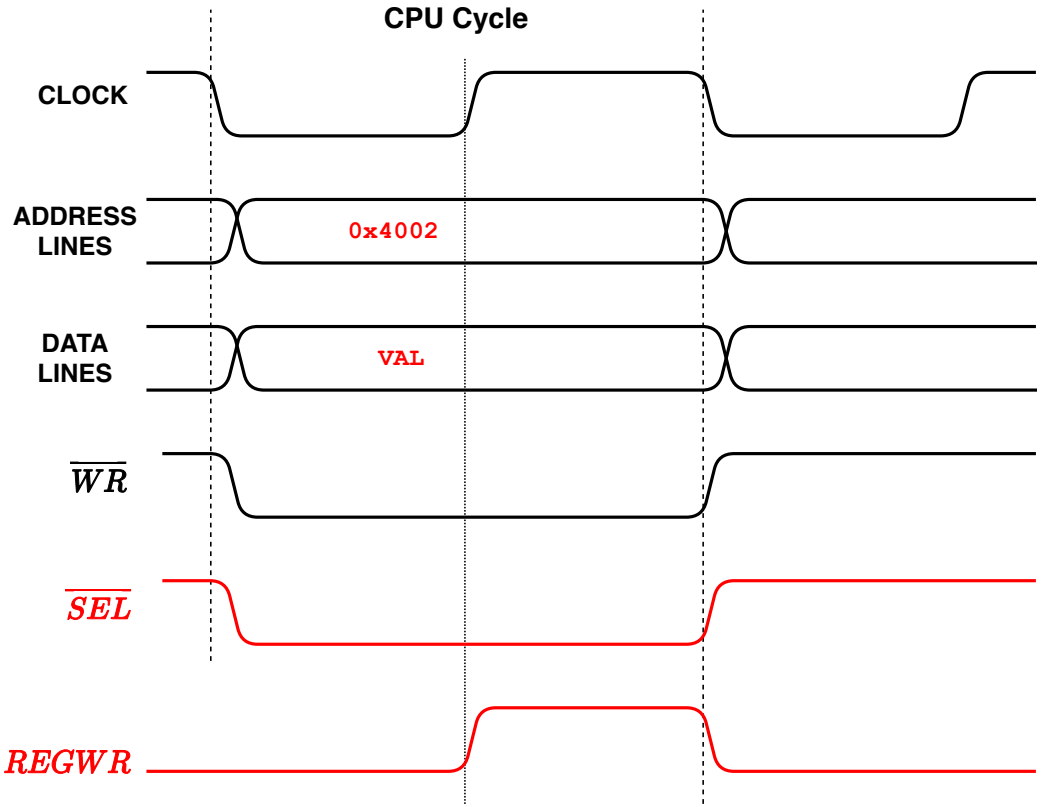


Address Decoder

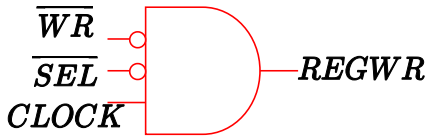
$$\overline{SEL} = 0, when \;\; address = 0x4002$$

$$\overline{SEL} = \overline{\overline{A_{31}} \ldots \overline{A_{15}} A_{14} \overline{A_{13}} \ldots \overline{A_3} A_2 A_1 \overline{A_0}}$$

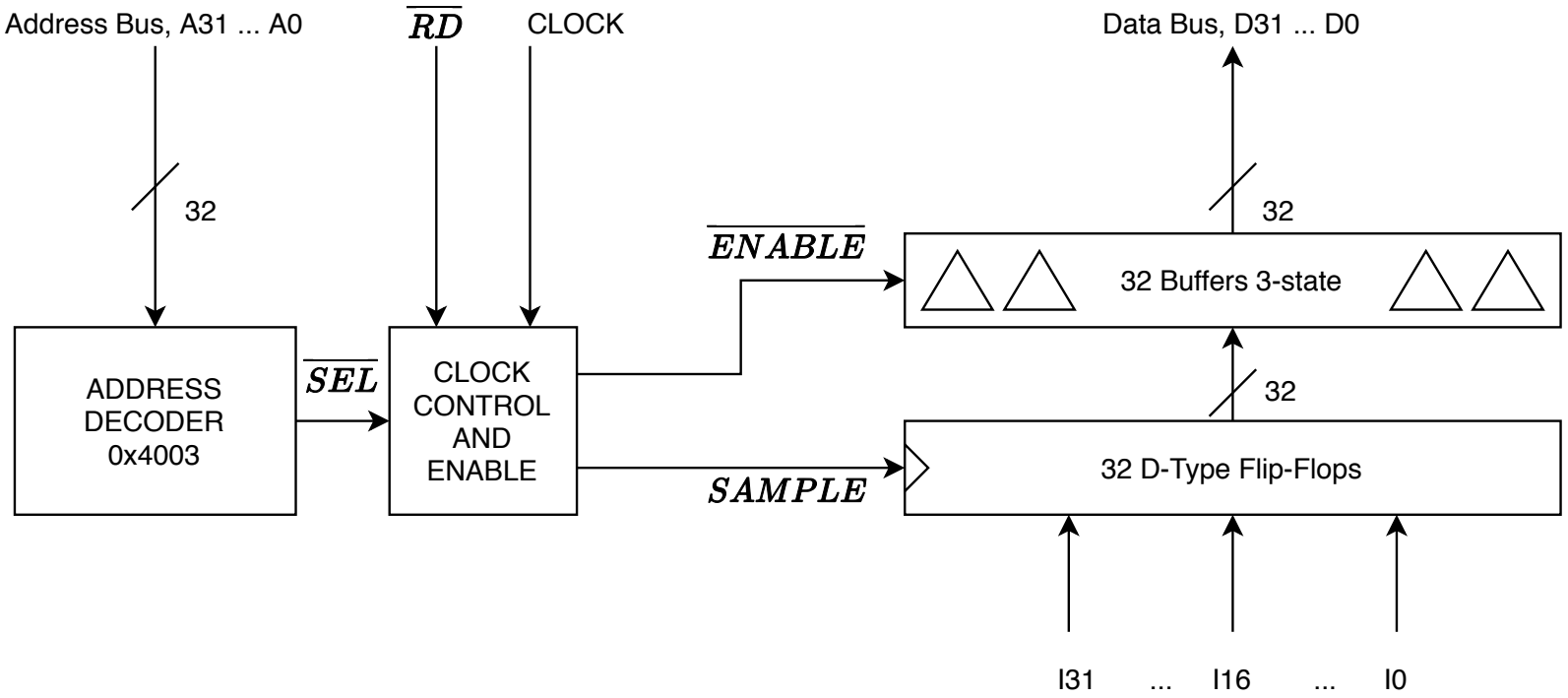
Esempio di Porta Digitale di Output



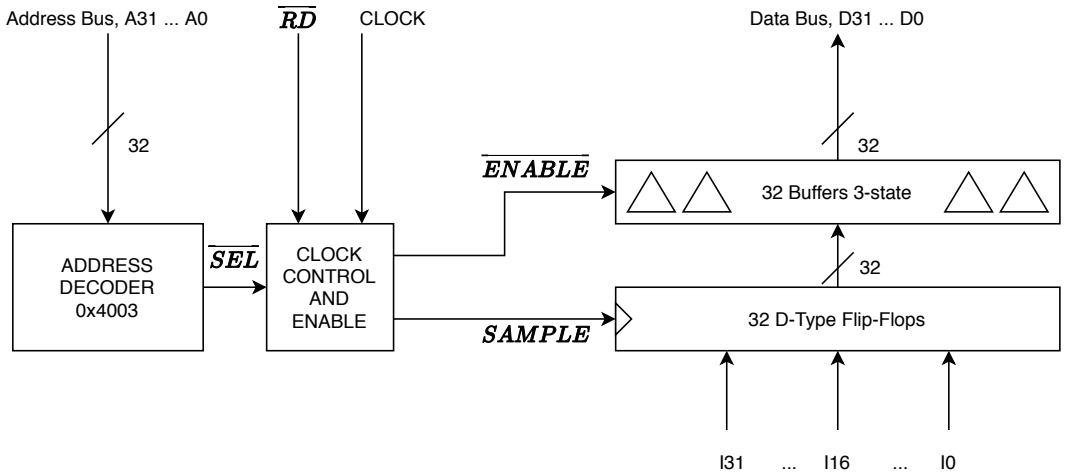
$$REGWR = CLOCK\ \overline{\overline{WR}}\ \overline{\overline{SEL}}$$



Esempio di Porta Digitale di Input



Esempio di Porta Digitale di Input

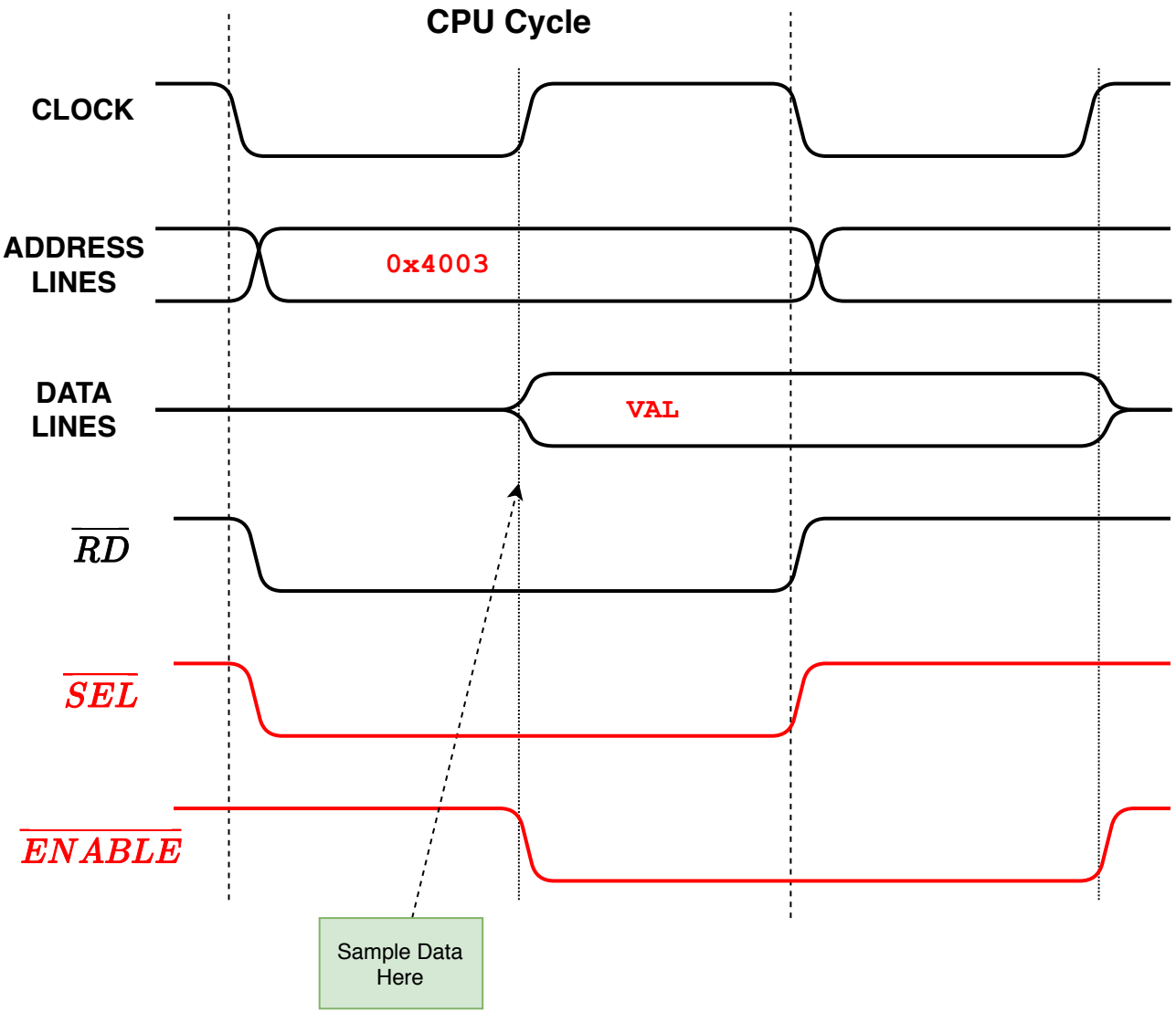


Address Decoder

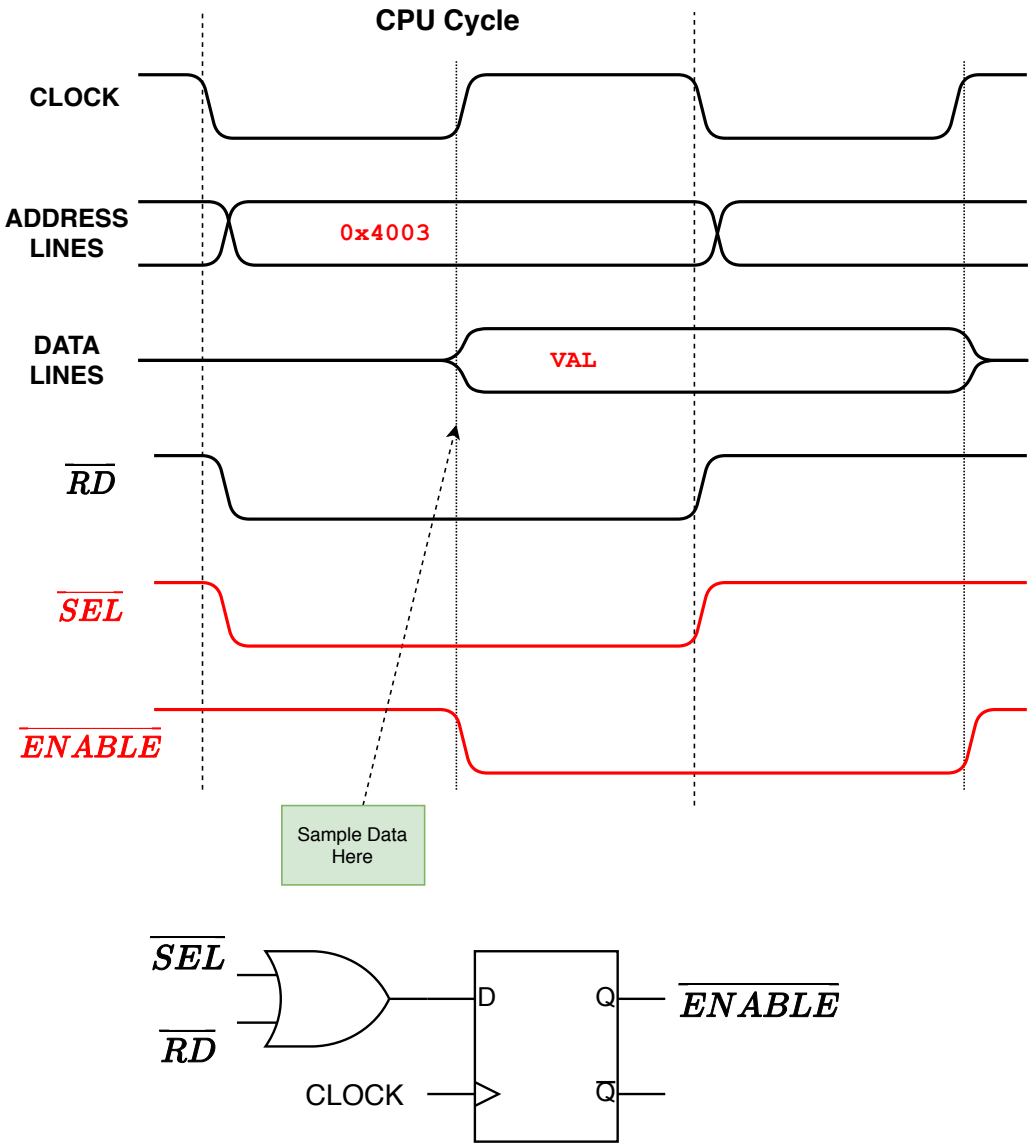
$$\overline{SEL} = 0, when \ \ address = 0x4003$$

$$\overline{SEL} = \overline{\overline{A_{31}} \ldots \overline{A_{15}} A_{14} \overline{A_{13}} \ldots \overline{A_3} \overline{A_2} A_1 A_0}$$

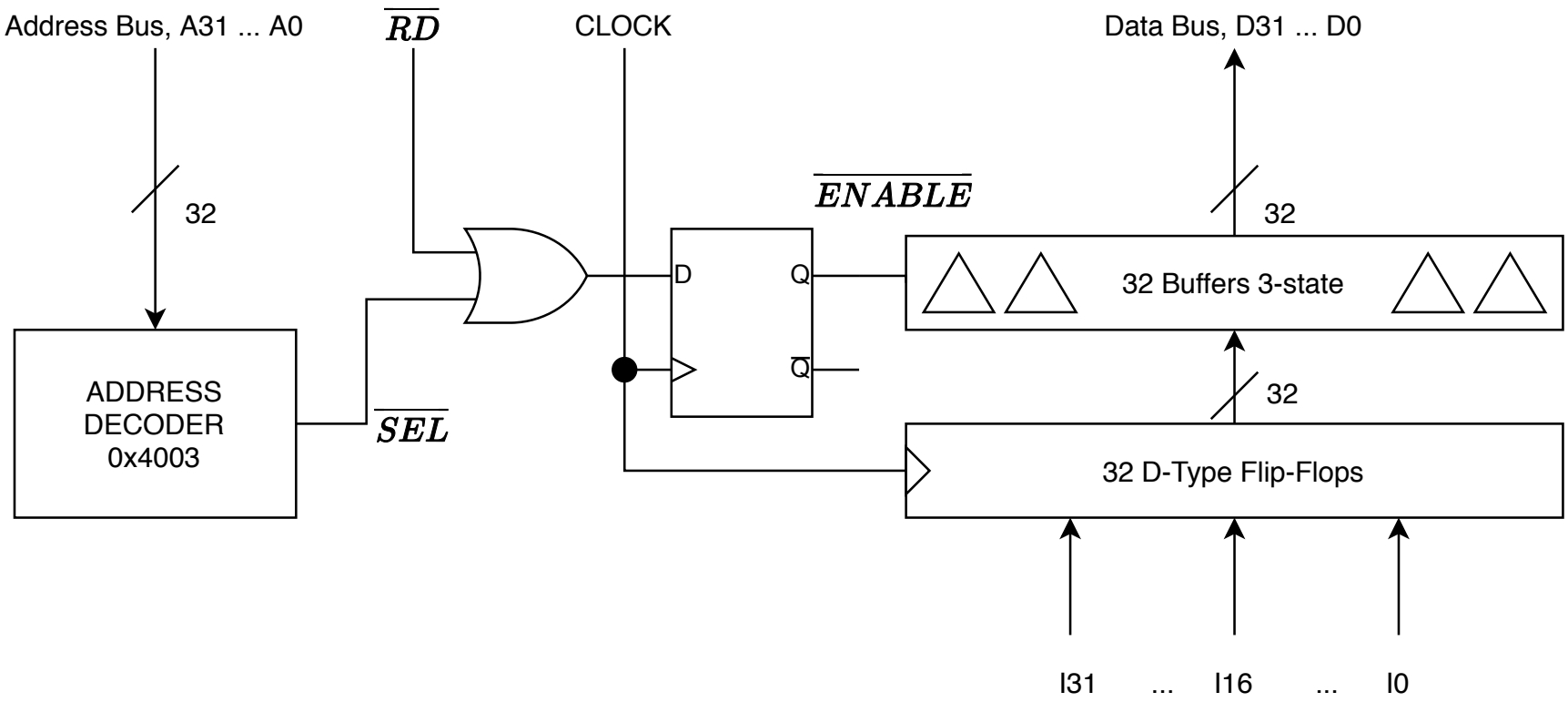
Esempio di Porta Digitale di Input—Timing



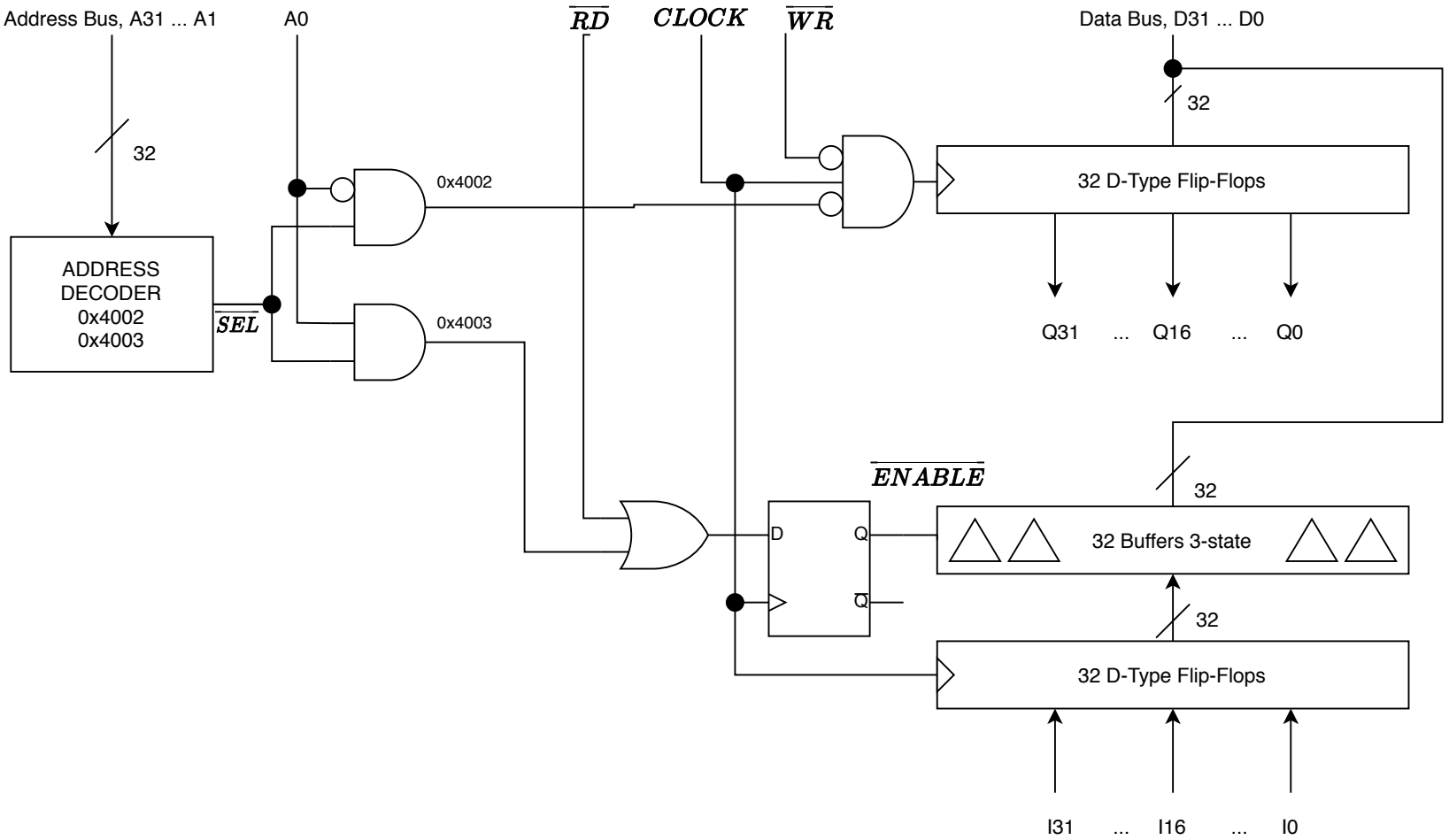
Esempio di Porta Digitale di Input—Segnale di Enable



Esempio di Porta Digitale di Input



Porta Digitale di Input/Ouput Completa

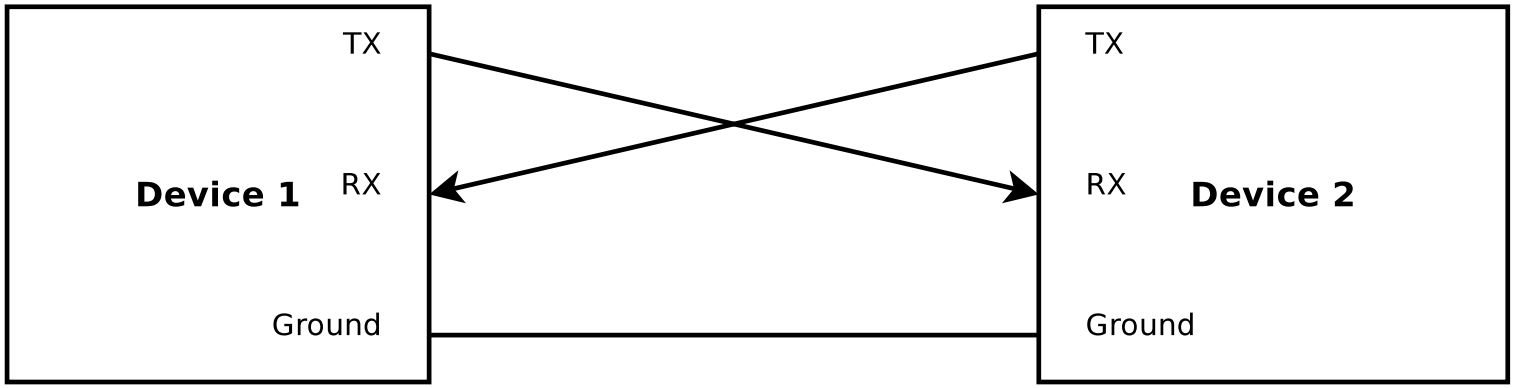


$$\overline{SEL} = \overline{\overline{A_{31}} \ldots \overline{A_{15}} A_{14} \overline{A_{13}} \ldots \overline{A_3} \overline{A_2} A_1}$$

L'Interfaccia Seriale (UART)

UART

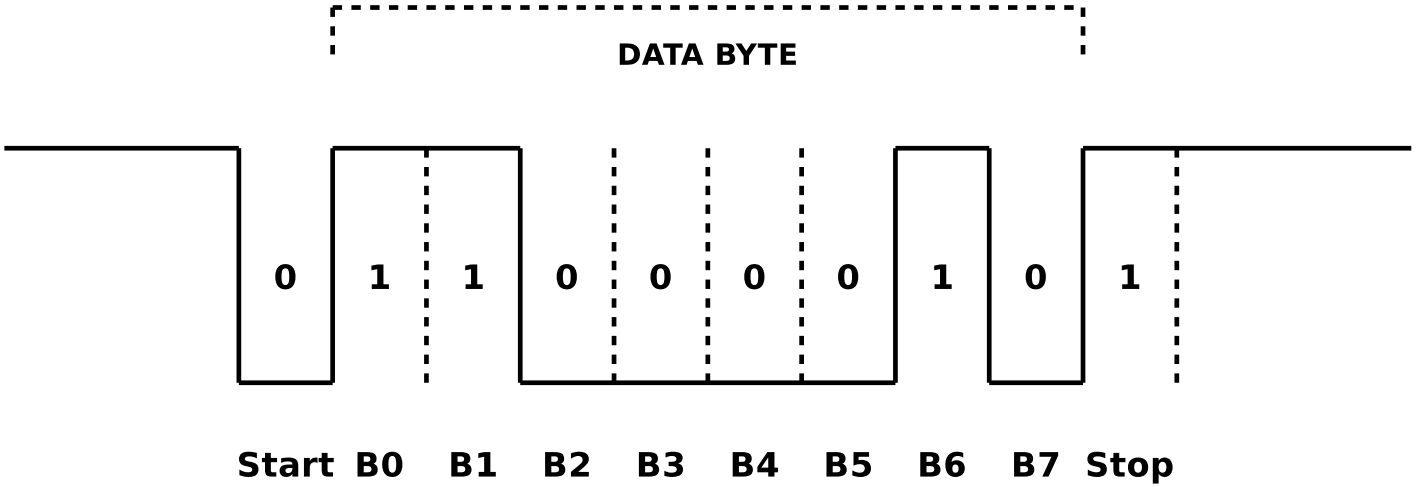
- UART = **Universal Asynchronous Receiver/Trasmitter**
- E' normalmente chiamata **porta seriale** (o **COM port**)
- E' una periferica per la comunicazione punto-punto tra due dispositivi a processore
- La comunicazione avviene **in seriale**, cioè un bit per volta
- L'interfaccia possiede due pin di comunicazione: **RX** e **TX**



UART: Caratteristiche elettriche e timing

- In assenza di trasmissione, la linea è a livello **logico "1"**
- Non appena il software avvia la trasmissione, scrivendo il dato in un apposito registro della periferica ...
- ... un primo bit **"0"** viene inviato, detto **start bit**
- successivamente il byte viene trasmesso un bit per volta a partire dal bit meno significativo (**LSB**)
- Uno o due **stop bits** (**Logical "1"**) concludono la trasmissione

Byte Trasmesso = "C", 0x43 = 0100 0011



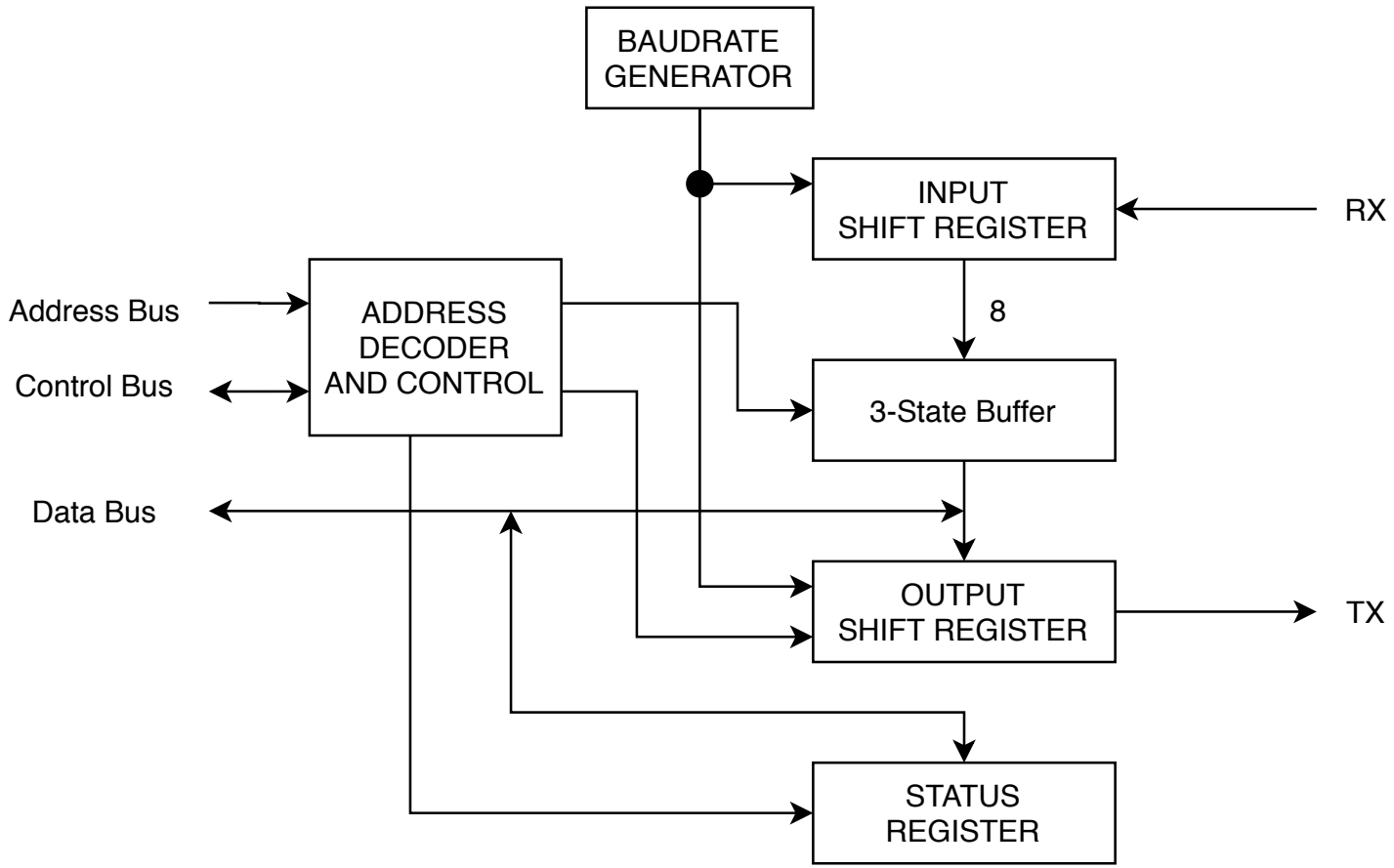
UART Parametri di Trasmissione

- **velocità di comunicazione**, in **bps = Bit Per Second** or **baud**
- **numero di bit per carattere**, normalmente **8**
- **presenza/assenza del bit di parità**, (usato per controllo d'errore), normalmente **assente**
- **numero di bit di stop**, normalmente **1**

L'impostazione **19200,8,N,1** significa:

- speed = 19200 bit-per-second;
- bits = 8;
- parity = None;
- stop bits = 1.

Schema Concettuale di una UART



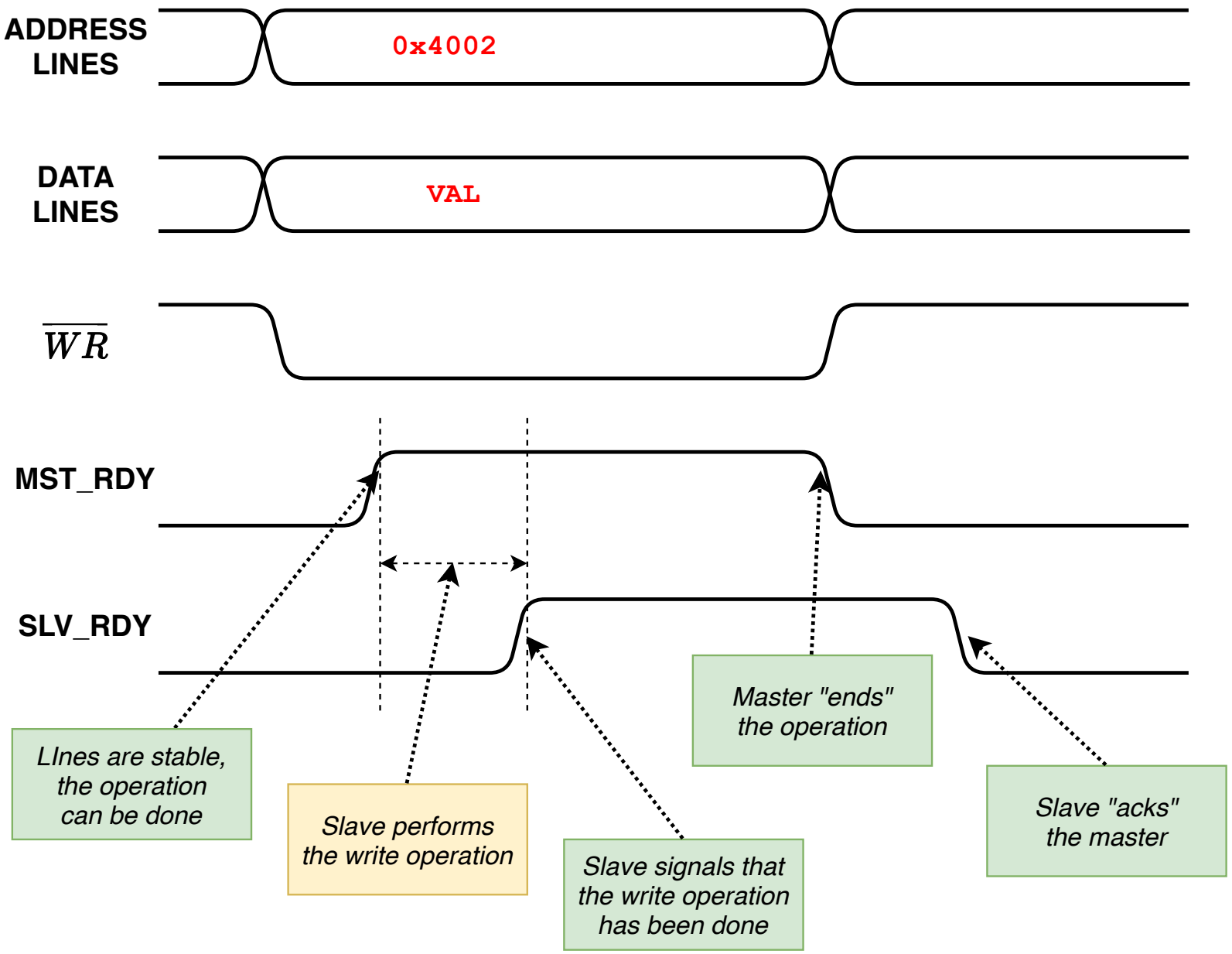
BUS Asincrono

Bus di Sistema Asincrono

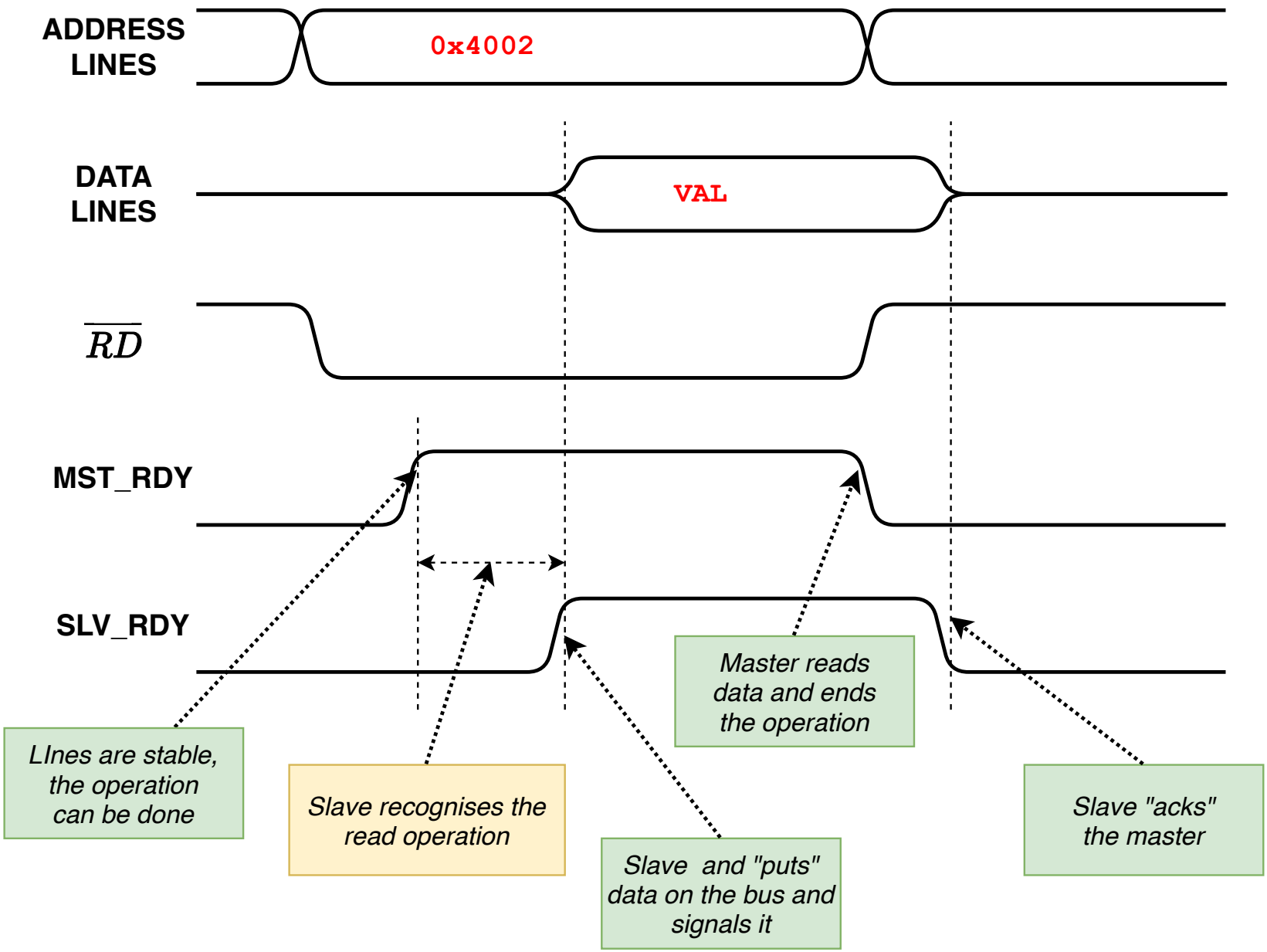
BUS Asincrono

- Nel Bus **asincrono** le operazioni avvengono solo all'occorrenza dei fronti di segnali speciali di **handshake**
- Il clock di sistema è sempre presente, tuttavia non regola le operazioni (sebbene possa essere usato per attivare i clock dei flip-flop delle periferiche)
- L'handshake avviene attraverso dei segnali sul Bus di controllo che regolano la segnalazione delle attività tra **CPU**—detta **master**—e **periferica**—detta **slave**
- Due segnali di controllo (tipicamente):
  - **Master-Ready** (da CPU a periferica) la CPU segnala che le informazioni su Bus dati/indirizzi sono valide
  - **Salve-Ready** (da periferica a CPU) la periferica segnala che l'operazione richiesta è stata effettuata

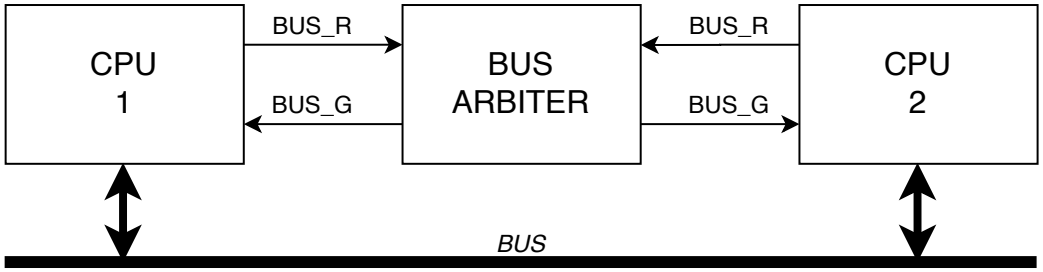
Operazione di Write su Bus asincrono



Operazione di Read su Bus asincrono



Multi-Master e Arbitraggio



Multi-Master

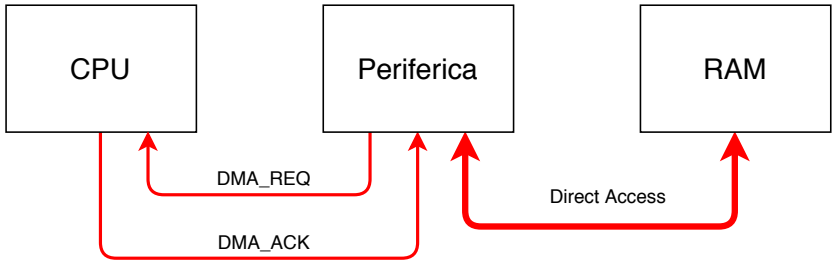
- Nei computer multi-CPU il bus è unico e condiviso da tutte le CPU
- Tuttavia solo una CPU per volta può accedere al Bus, altrimenti si verifica un conflitto
- Un circuito **Bus Arbiter** è presente per gestire l'accesso al Bus in modo mutuamente esclusivo
- Quando una CPU deve accedere al Bus attiva il proprio segnale di *BUS_R* (Bus Request)
- L'Arbiter risponde con un *BUS_G* (Bus Grant)
- La CPU accede al Bus

Direct Memory Access

Direct Memory Access (DMA)

- Alcune periferiche hanno la necessità di trasferire una quantità di dati molto più grande di una singola word
- Ad esempio, un'interfaccia di memoria di massa (hard disk) trasferisce un **intero blocco** di disco, il quale equivale a 512 byte
- Poichè il trasferimento word-by-word, gestito dal software e dalla CPU potrebbe essere lento, si adotta una tecnica detta **Direct Memory Access (DMA)**
- Nel DMA, la periferica ha la possibilità di accedere direttamente alla memoria di sistema prendendone il controllo
- E' dunque possibile che la periferica "depositi" o "prelevi" un intero blocco di dati dalla RAM, senza passare dal controllo della CPU

Direct Memory Access

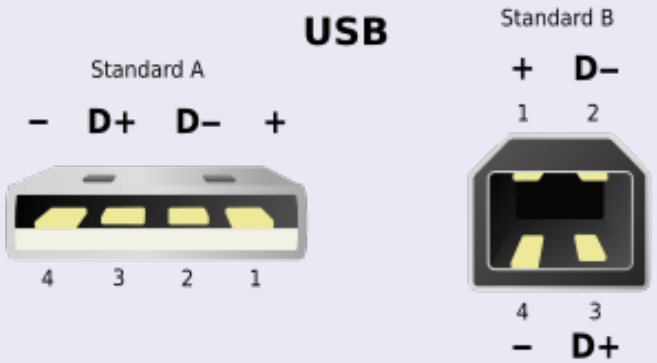


- Il Control Bus possiede due linee di handshake: *DMA_REQ* e *DMA_ACK*
- Quando una periferica vuole iniziare un trasferimento DMA, “attiva” la linea *DMA_REQ*
- La CPU conclude l’esecuzione dell’istruzione in corso e concede il Bus attivando, come risposta, la linea *DMA_ACK*
- La periferica riconose la riposta, prende il controllo del Bus ed effettua il trasferimento, al termine del quale “disattiva” la linea *DMA_REQ*, segnalando alla CPU che l’attività si è conclusa
- Opzionalmente, la periferica può generare un segnale di *IRQ* per far sì che il software possa gestire altre attività legate alla conclusione del trasferimento

Bus Standard

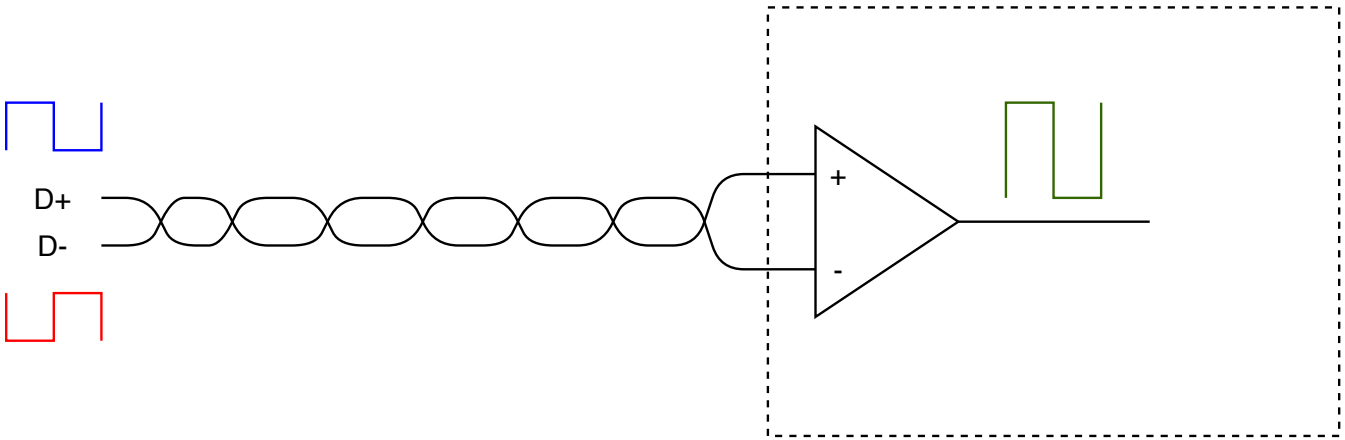
USB

- L'USB è un sistema standard di comunicazione per periferiche con caratteristiche di "hot plug"
- E' un sistema di comunicazione **seriale** ad alta velocità:
  - **USB1**: 1.5 Mbit/s, 12 Mbit/s
  - **USB2**: 480 Mbit/s
  - **USB3**: 5 Gbit/s
- Il connettore ha quattro poli:
  - **Alimentazione +/-**
  - **Segnale dati** differenziale D+ , D-



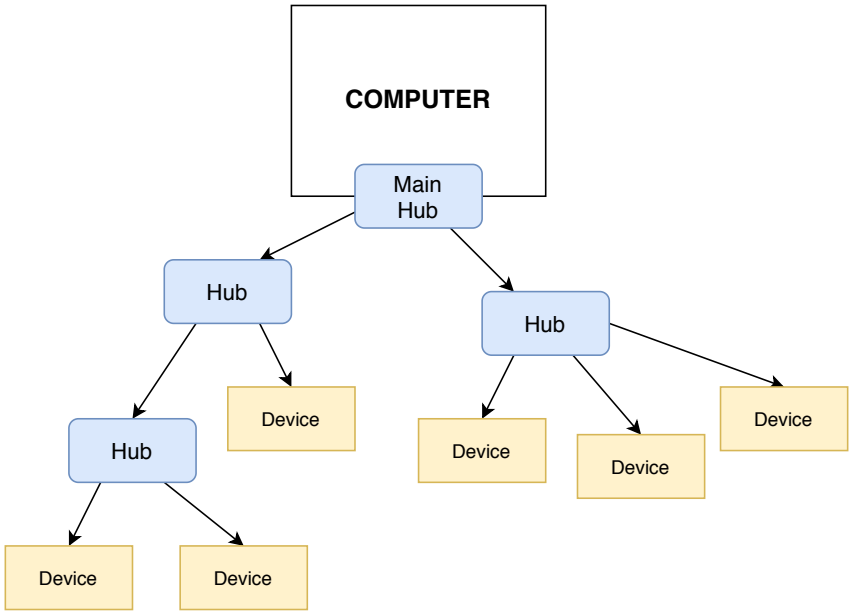
Trasmissione Differenziale

- I segnali $D+$ e $D-$ trasportano il treno di bit seriale
- Dal punto di vista elettrico, il segnale su $D-$ è sempre l'**opposto** di $D+$
- In ricezione, un opportuno circuito (amplificatore operazione) effettua una **differenza elettrica** tra i due segnali e genera il treno di bit da inviare agli shift register di ricezione
- Qualora ci sia un disturbo sulle linee elettriche, esso di ripercuoterà **identico** sia su $D+$ che su $D-$, pertanto l'operazione di differenza potrà "cancellarlo"



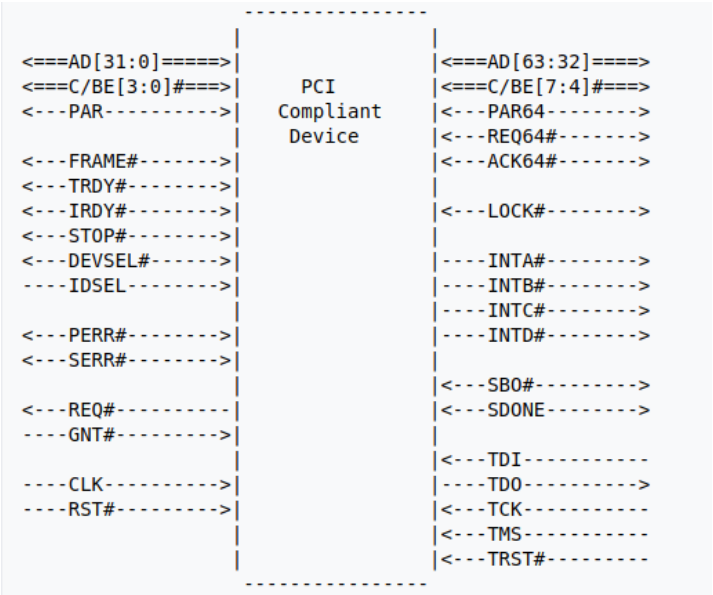
Architettura

- **Device**: dispositivo periferico
- **Hub**: concentratore di comunicazione
- **Accesso ai Device**: polling periodico da hub radice verso le foglie
- **Tipologia di traffico**: asincrono e isocrono



Peripheral Component Interconnect

- Il bus **PCI (Peripheral Component Interconnect)** è un bus standard sviluppato da Intel
- E' un bus parallelo con 124 linee elettriche, operante a 33 MHz e consente il trasferimento di dati alla velocità di 132 MBytes/sec
- E' progettato per consentire il trasferimento di dati CPU/periferica a **blocchi** (burst/raffica)



(Fonte: Wikipedia)

Peripheral Component Interconnect

- Possiede 64 linee in cui sono multiplexati sia gli indirizzi che i dati
- Il controllo avviene attraverso 4 linee, denominate C/BE (Command/Byte-Enable), in cui viene specificata l'operazione che si intende eseguire sul Bus

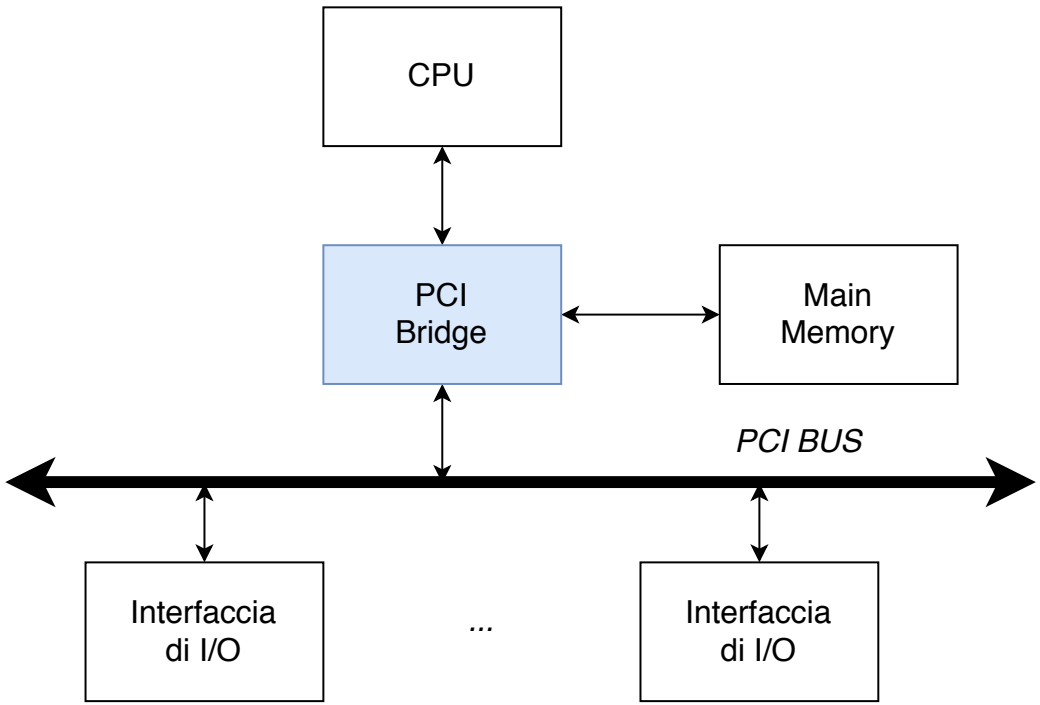
C/BE[3:0]#	Command Types
0000	Interrupt Acknowledge
0001	Special Cycle
0010	I/O Read
0011	I/O Write
0100	Reserved
0101	Reserved
0110	Memory Read
0111	Memory Write
1000	Reserved
1001	Reserved
1010	Configuration Read
1011	Configuration Write
1100	Memory Read Multiple
1101	Dual Address Cycle
1110	Memory Read Line
1111	Memory Write and Invalidate

Peripheral Component Interconnect

- Lo standard supporta il “plug-and-play”
- Definisce uno spazio di indirizzamento standard (PCI Configuration Space) in cui i registri di dell’hardware hanno funzionalità unificate
- Due sono le informazioni principali che permettono di identificare la scheda:
  - **Vendor ID**
  - **Device ID**
- La fase di configurazione prevede l’identificazione della scheda (Device ID) e l’assegnazione di un set di indirizzi di I/O
- Ogni scheda non ha pertanto un set di indirizzi predefinito dall’hardware, ma viene assegnato dinamicamente con il supporto del sistema operativo

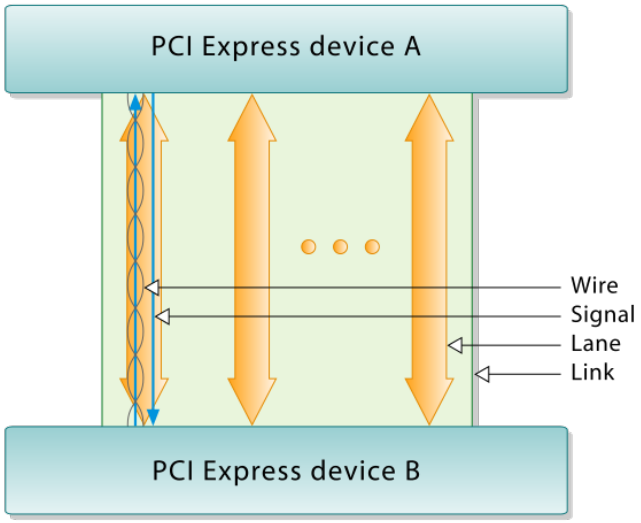
Architettura Hardware

- Le CPU non supportano direttamente il bus PCI, pertanto è necessario sempre un “bridge” che gestisca la comunicazione



PCIe

- Il **PCI-Express** è un bus standard che concettualmente discende dal PCI, ma adotta un modello hardware totalmente differente
- Il bus è seriale e la comunicazione avviene attraverso una serie di **lanes**
- Ogni "lane" è un canale di comunicazione seriale le cui linee di tramissione e ricezione sono separate e sono differenziali
- La velocità di trasmissione è dell'ordine dei Gbit/s

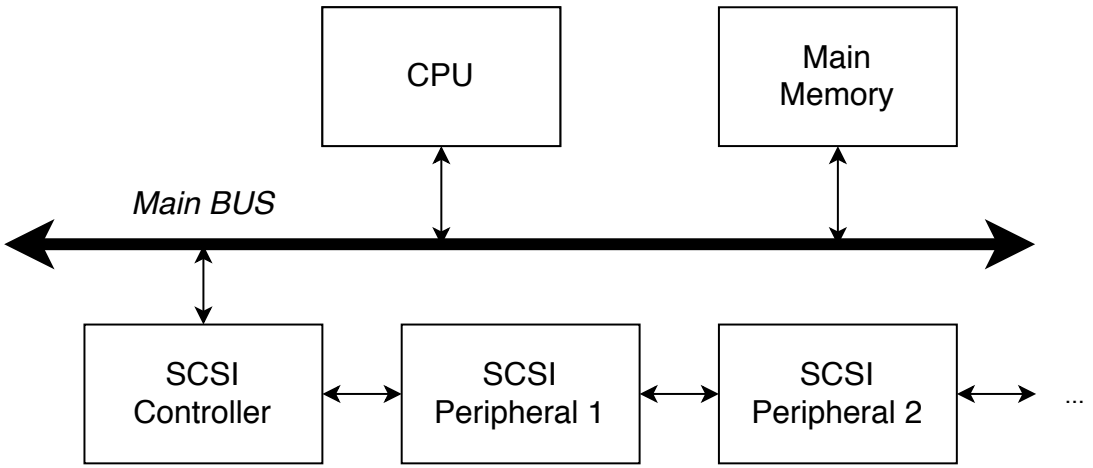


(Fonte: Wikipedia)

Small Computer Standard Interface

SCSI

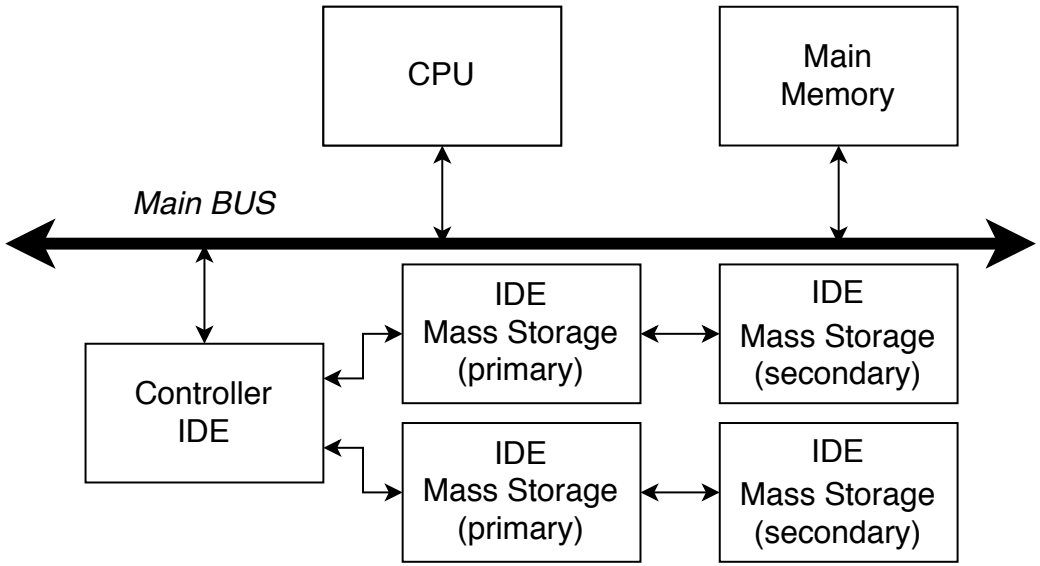
- L'interfaccia **SCSI** è stata introdotta come standard parallelo per interconnettere periferiche senza limitazioni
- E' basata sulla presenza di un **Controller SCSI** che si connette al bus del PC, dal quale poi, attraverso un cavo parallelo, vengono connesse le periferiche in "cascata"
- I tipici dispositivi SCSI sono memorie di massa, scanner, stampanti, dispositivi di acquisizione dati



Bus IDE (o ATA)

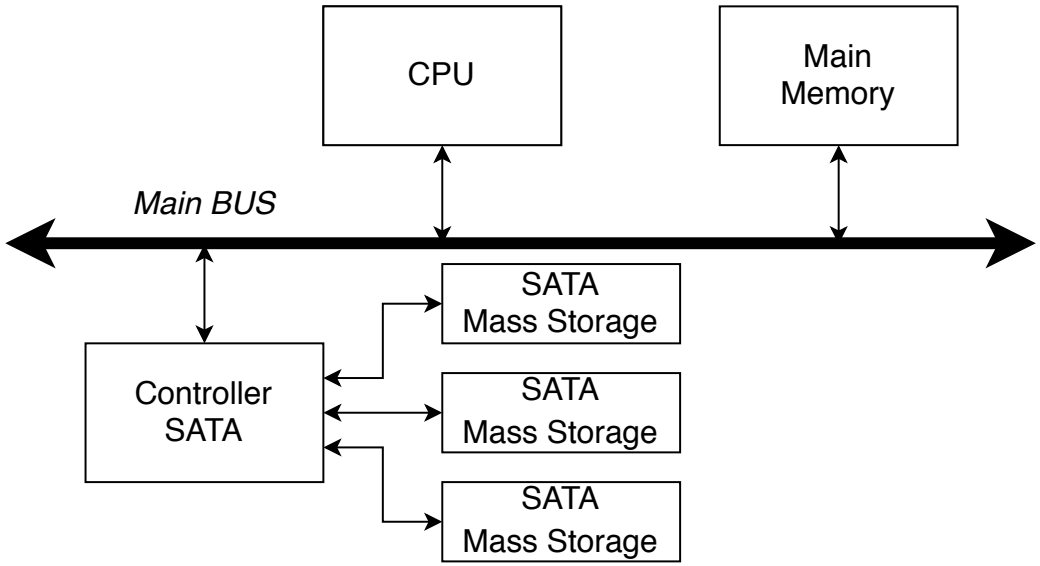
IDE / ATA

- L'interfaccia **IDE** è la prima interfaccia per memoria di massa introdotta sui PC di attuale generazione
- E' un'interfaccia parallela per il trasferimento di blocchi di dati
- Offre diversi connettori, ognuno dei quali può connettere, in cascata, due dispositivi periferici, un **primary device** e un **secondary device**



Serial ATA

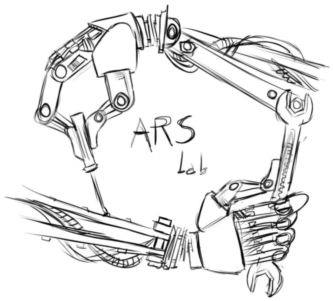
- L'interfaccia **SATA** è uno standard di ultima generazione per il collegamento di memorie di massa
- E' un'interfaccia seriale diffrenziale ad altissima velocità:
  - **SATA 1.0**: 1.5 Gbit/s
  - **SATA 2.0**: 3 Gbit/s
  - **SATA 3.0**: 6 Gbit/s



Il Sistema di Input/Output

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it



Architettura degli Elaboratori