Multi-Processor Systems

Pirouz Bazargan Sabet

Sorbonne Université - LIP6

Pirouz.Bazargan-Sabet@lip6.fr

Outline

- System Bus
- An example of on-chip bus : Pi-Bus

Introduction

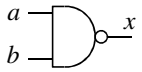
What is a System Bus ?

- A set of wires connecting the different components of a system
- A mean of communication between these components

Introduction

Why do we use the word *Bus* ?

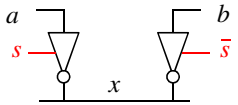
- A *Bus* does **not** mean a set of wires
- On a wire there is one and only one emitter

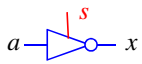


Introduction

Why do we use the word *Bus* ?

- On a *Bus* we may connect several emitters





Tri-State emitter

Si $s = 0$ *x undefined*

Si $s = 1$ $x = \bar{a}$

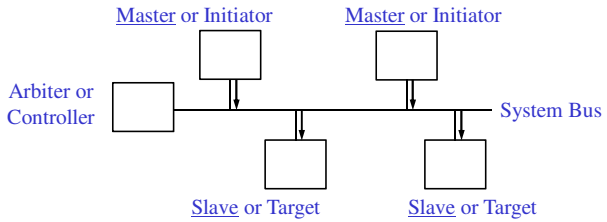
Introduction

What is a System Bus ?

- A set of wires connecting the different components of a system
- A mean of communication between these components
- 3 types of components may be connected on a *Bus*

Introduction

What is a System Bus ?



Introduction

What is a System Bus ?

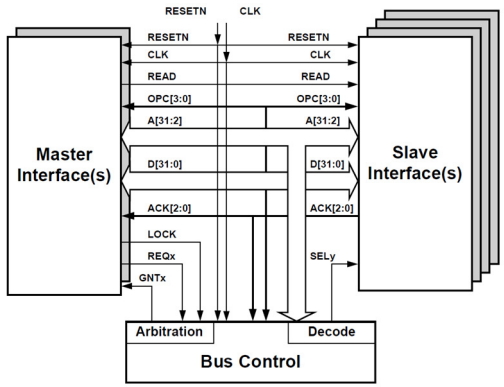
- *Transfer* : a data exchange on the bus
- *Transaction* : A complete sequence of exchanges on the bus following the bus protocol starting with the connection of the master till its disconnection.

Pi – Bus

- Pi-Bus is a synchronous on-chip system bus
- An European project aiming at the definition of a standard on-chip bus (OMI – 1994-2005)

Advanced RISC Machines
INMOS
Matra MHS
Philips
Siemens

Pi – Bus



Pi – Bus

Name	Emitter	Receiver	N bits	Signification
Ck	Ext	CMS	1	*Clock* : Synchronization signal
ResetN	Ext	CMS	1	*Reset* : Reset signal – negative logic
Req	M	C	1*M	*Request* : Request to use the bus
Gnt	C	M	1*M	*Grant* : Permission to use the bus
A	M	CS	30	*Address*
Opc	M	S	4	*Opcode* : Byte selection
Sel	C	S	1*S	*Select* : Selection of the slave
D	MS	MS	32	*Data*
Read	M	S	1	*Read* : Operation type – read or write
Ack	S	MC	3	*Acknowledge* : Slave's acknowledgment
Lock	M	C	1	*Lock* : Still need the bus
Tout	C	MS	1	*TimeOut* : Too long ! – Transaction aborted

Pi – Bus

Opc	M	S	4	*Opcode* : Byte selection

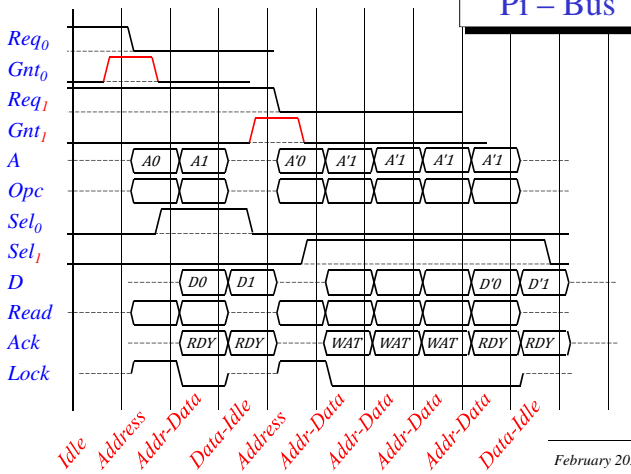
WDU	0010	*Word Undefined*
HW0	1000	*Half Word 0*
HW1	1010	*Half Word 1*
BY0	1100	*Byte 0*
BY1	1101	*Byte 1*
BY2	1110	*Byte 2*
BY3	1111	*Byte 3*
NOP	0000	*No Operation*

Pi – Bus

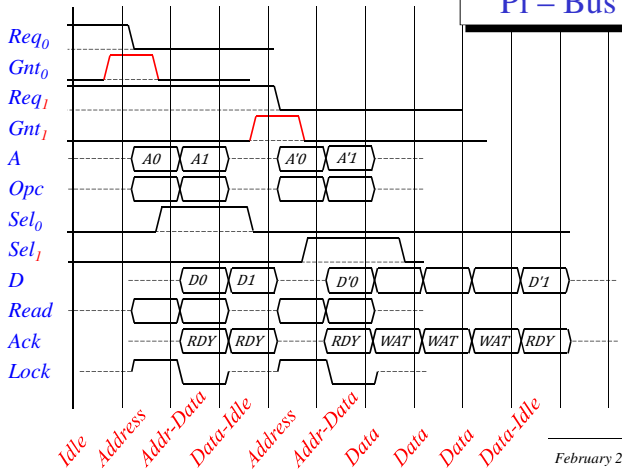
Ack	*S*	*MC*	**3**	*Acknowledge* : Slave's acknowledgment

RDY	`011`	*Ready*
WAT	`000`	*Wait*
RTR	`100`	*Retract*

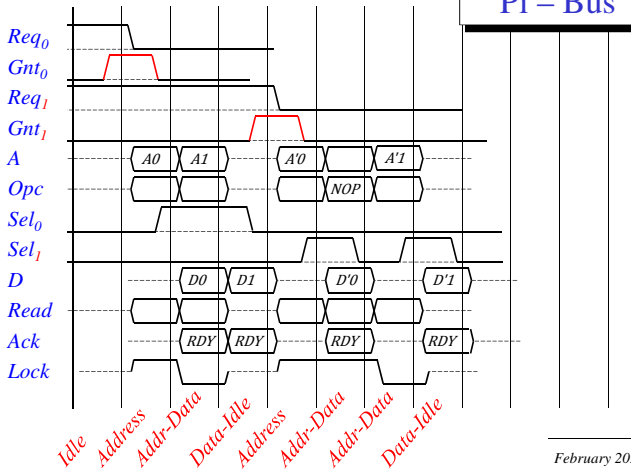
Pi – Bus



Pi – Bus

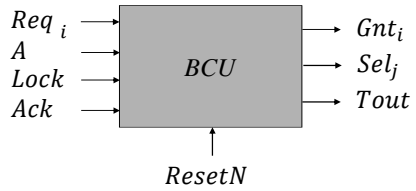


Pi – Bus



Pi – Bus

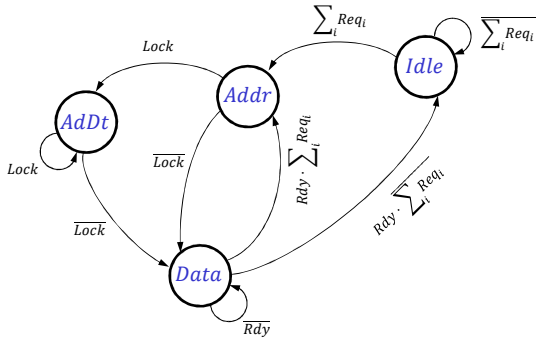
Bus controller



Pi – Bus

Bus controller

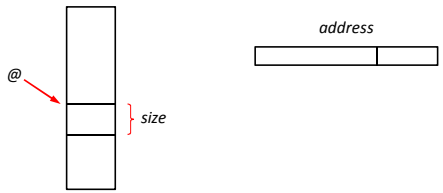
assuming $\overline{Rdy} = Wat$



Pi – Bus

Bus controller Selecting the Slave

- A slave is mapped on a portion of the memory space : first address and size (power of 2)



Pi – Bus

Bus controller Selecting the Slave

- A slave is mapped on a portion of the memory space : first address and size (power of 2)
- The most significant bits of the address (send by the Master) is decoded by the Controller to select the Slave

Pi – Bus

Bus controller — Arbitration : selecting the Master

- When several Masters wants to access the bus, the controller has to select the one than can use the bus
- The selection is based on an order of *priority*

Pi – Bus

Bus controller — Arbitration : selecting the Master

Fixed order

- Easy to implement
- Unfair

Pi – Bus

Bus controller — Arbitration : selecting the Master

Random order

- Requires a truly random selection to give each Master the same probability to be selected – random number generator
- Hard to implement

Pi – Bus

Bus controller — Arbitration : selecting the Master

First arrived first served

- Requires a FiFo
- Several Masters may send a request in the same cycle

Pi – Bus

Bus controller Arbitration : selecting the Master

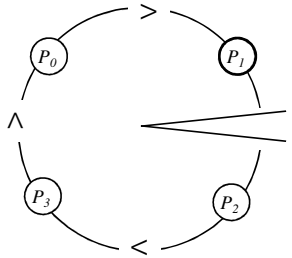
Wrapped fixed order (Round robin)

- Set a fixed *pseudo-order* between the Masters : $P_0 > P_1 > \cdots > P_n > P_0$
- The Master that has been selected gets the least priority
- Fair
- No starving problem

Pi – Bus

Bus controller Arbitration : selecting the Master

Wrapped fixed order (Round robin)



Pirouz Bazargan Sabet *February 2021*