3 Byte Memory

OUTLINE:

- **Designing an external 3 Byte Memory**
- **De-multiplexing the ATmega103 PORTS in to more devices**

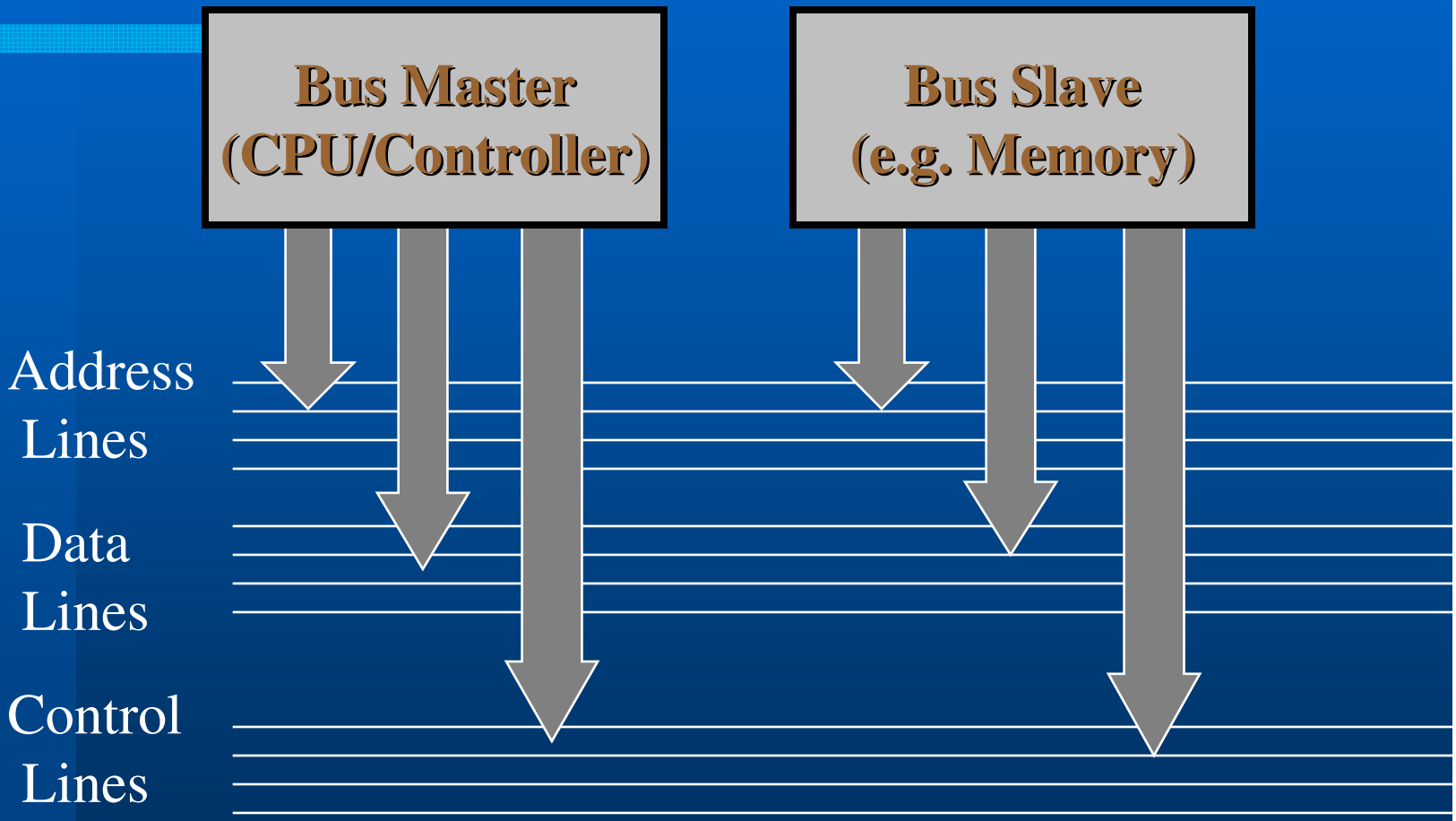
Costas Foudas, Imperial College,
Rm: 508, x47590

Computer Bus

- The processor of a computer communicates with the other computer modules via a device called: Bus.
- Several Bus architectures exist in the market such as PCI; cPCI; VME….
- All bus architectures include a control bus, a data bus and an address bus.

Costas Foudas, Imperial College,
Rm: 508, x47590

The BUS of a computer:



Typical Write Sequence

- The master places the address of the memory where the data is to be written on the bus and qualifies it using the control lines.
- The master signals using the control lines that this is a write sequence.
- The master places the data to be written on the bus and qualifies them using the control lines
- The slave compares the address on the bus with its own address. If the write refers to him, he takes the data and signals using the control lines that he is done (acknowledges the data)

Typical Read Sequence

- The master places the address of the memory from where the data is to be red on the bus and qualifies it using the control lines.
- The master signals using the control lines that this is a read sequence.
- The master signals that he is ready to accept data using the control lines.
- The slave compares the address on the bus with its own address. If the read refers to him, he places the data on the bus and signals using the control lines that he is done (acknowledges the data). At the end the Master Latches the data.

Parallel Transfer

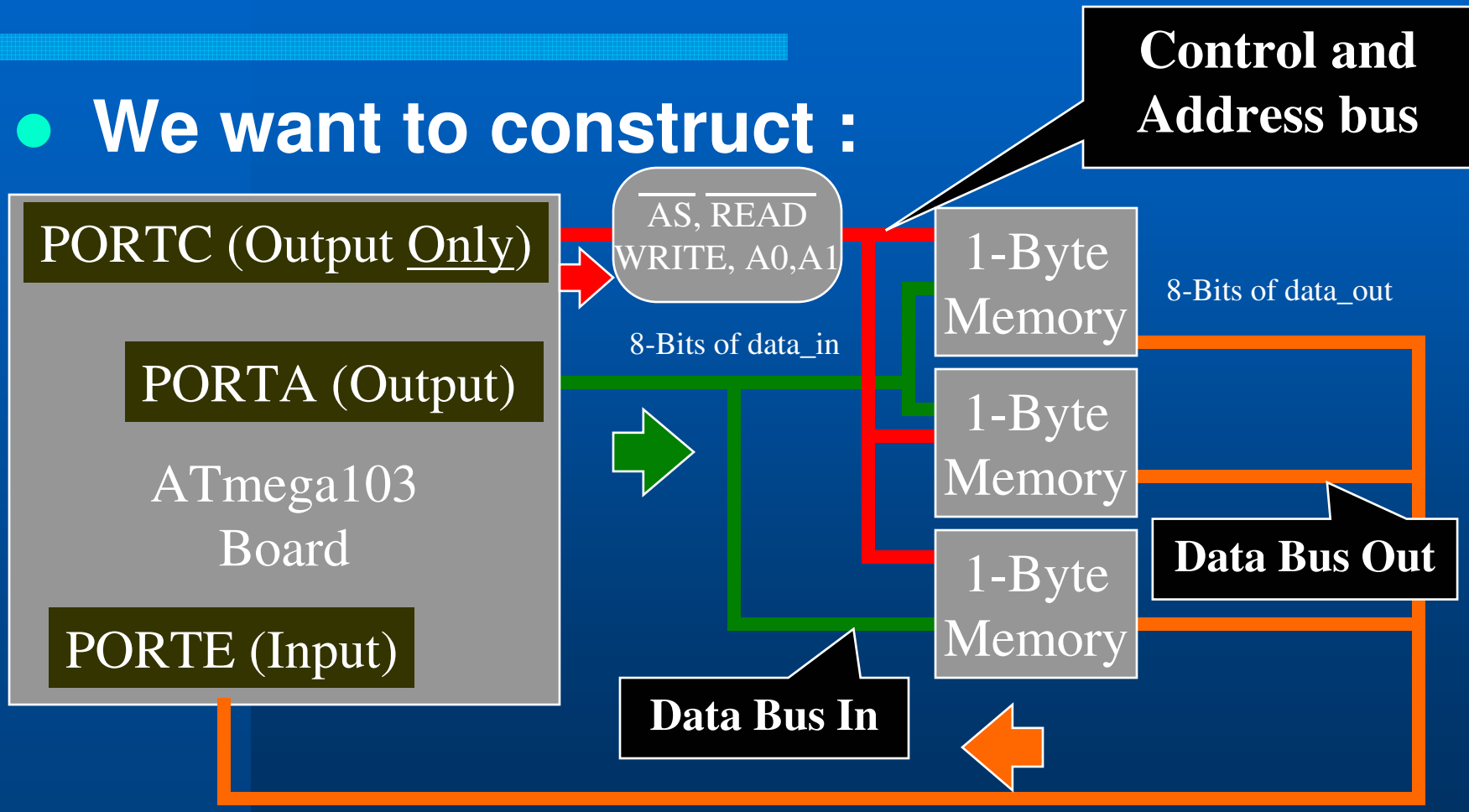
- In a bus all bits of a byte or word are transferred simultaneously.
- If we have a byte wide bus and a transfer frequency of 1MHz then we have a bus speed of 1Mbytes/sec
- Hence, this is called Parallel Data Transfer (oppose to Serial Data Transfer that we will learn next)

Electronics Exercise

Make a 3 Byte memory (SRAM) that you can read and write from the ATmega103 board.

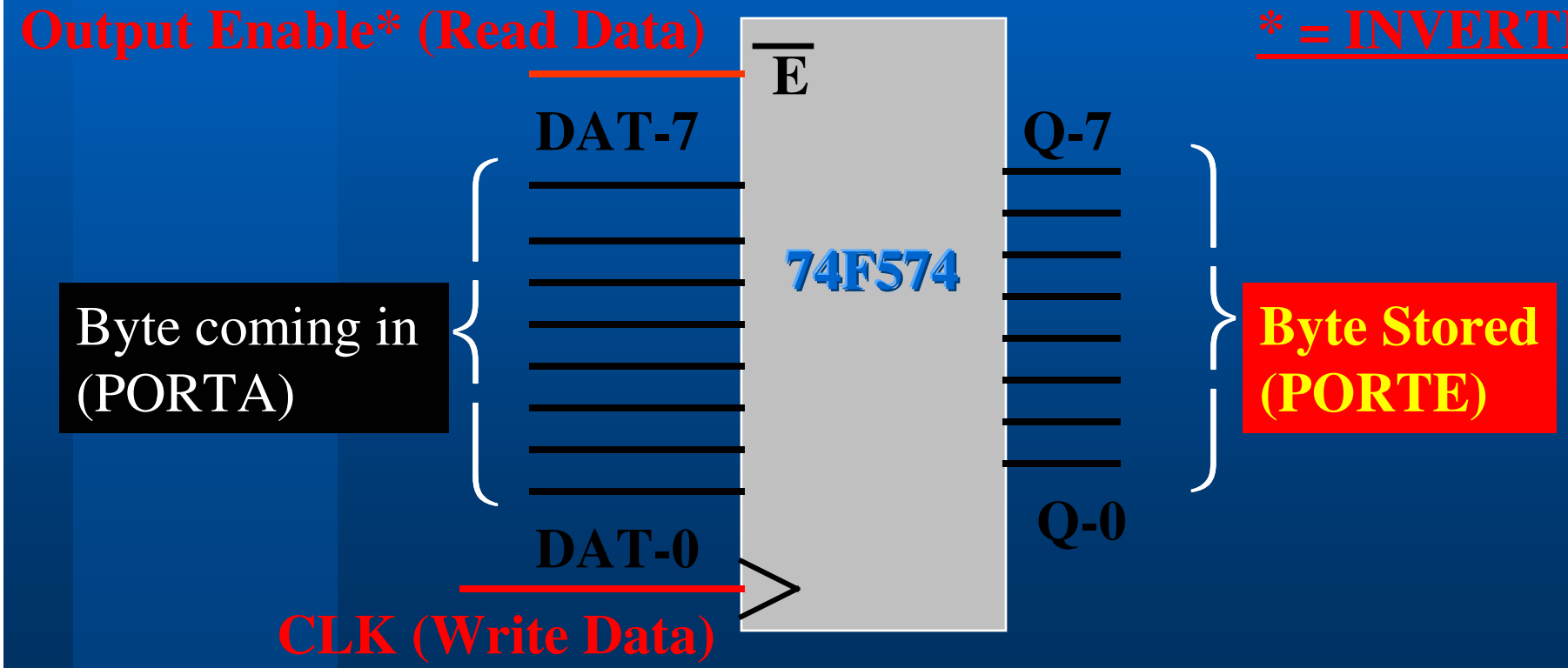
High Level Design :2 Byte Memory

- **We want to construct :**



Guess what is the 1 Byte memory ?

- **It exists in one package :**

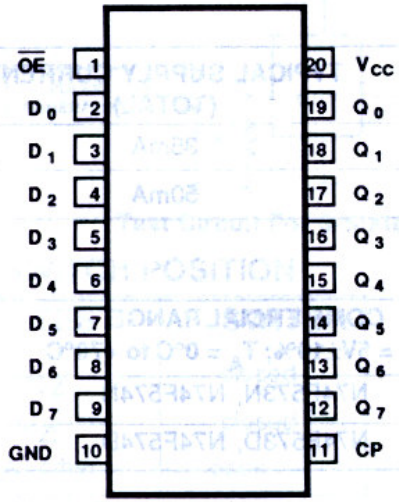


The data sheets of the 74F574

Register

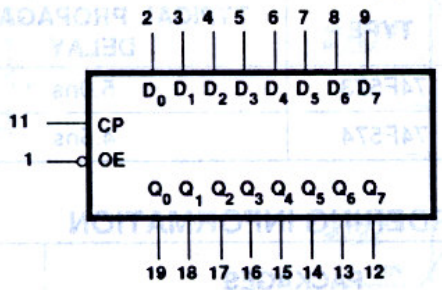
PIN CONFIGURATION

74F574



LOGIC SYMBOL

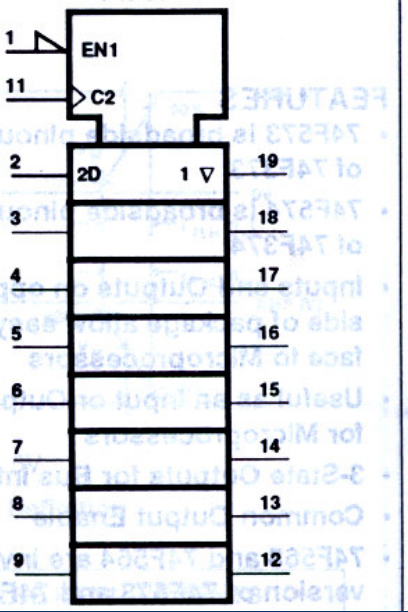
74F574



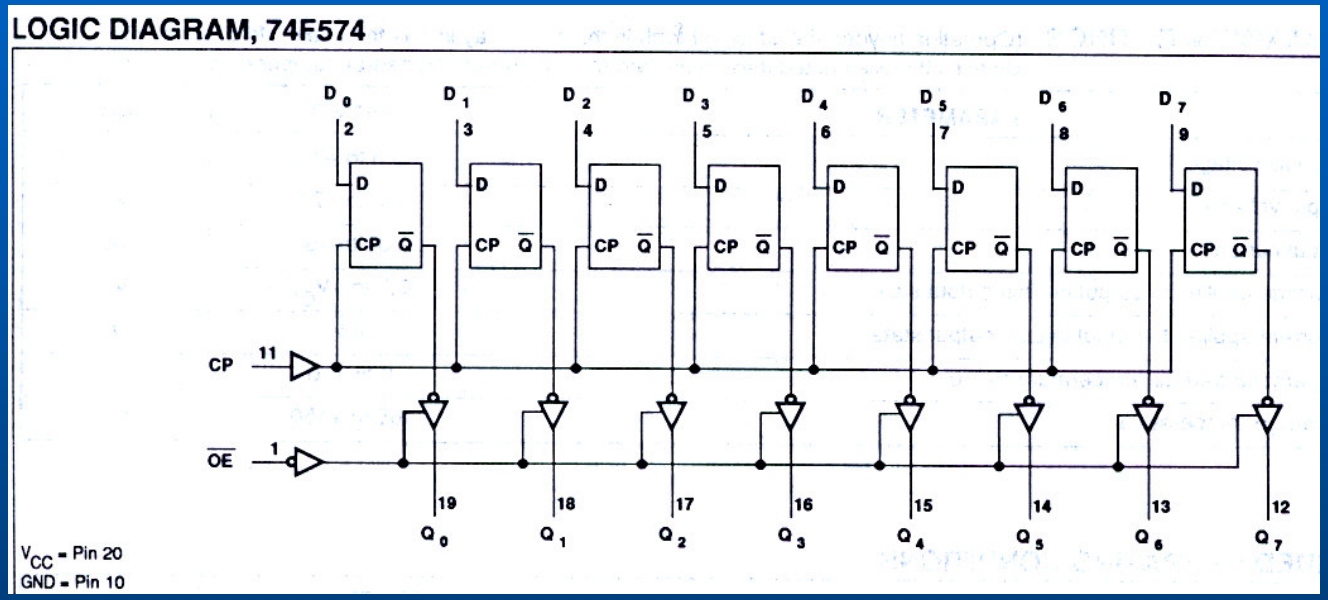
V_{CC} = Pin 20
GND = Pin 10

LOGIC SYMBOL(IEEE/IEC)

74F574



What does the 74F574 do ?



The outputs of the 574 are tri-state : They Can be high ‘1’, low ‘0’, and DISCONNECTED (HIGH IMPEDANCE STATE).

The 74F574 Truth Table

FUNCTION TABLE, 74F574

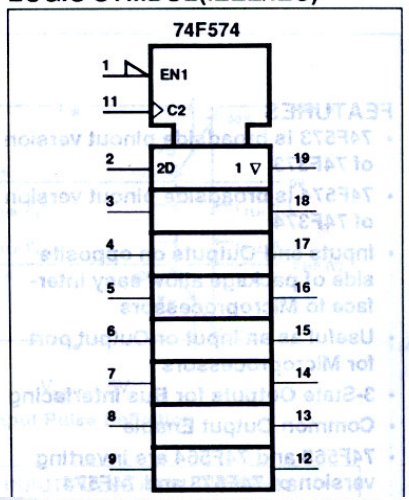
INPUTS			INTERNAL	OUTPUTS	OPERATING MODE
$\overline{OE}$	CP	D_n	REGISTER	Q_0 - Q_7	
L	↑	l	L	L	Load and read register
L	↑	h	H	H	
L	↕	X	NC	NC	Hold
H	↑	D_n	D_n	Z	Disable outputs
H	X	X	X	Z	

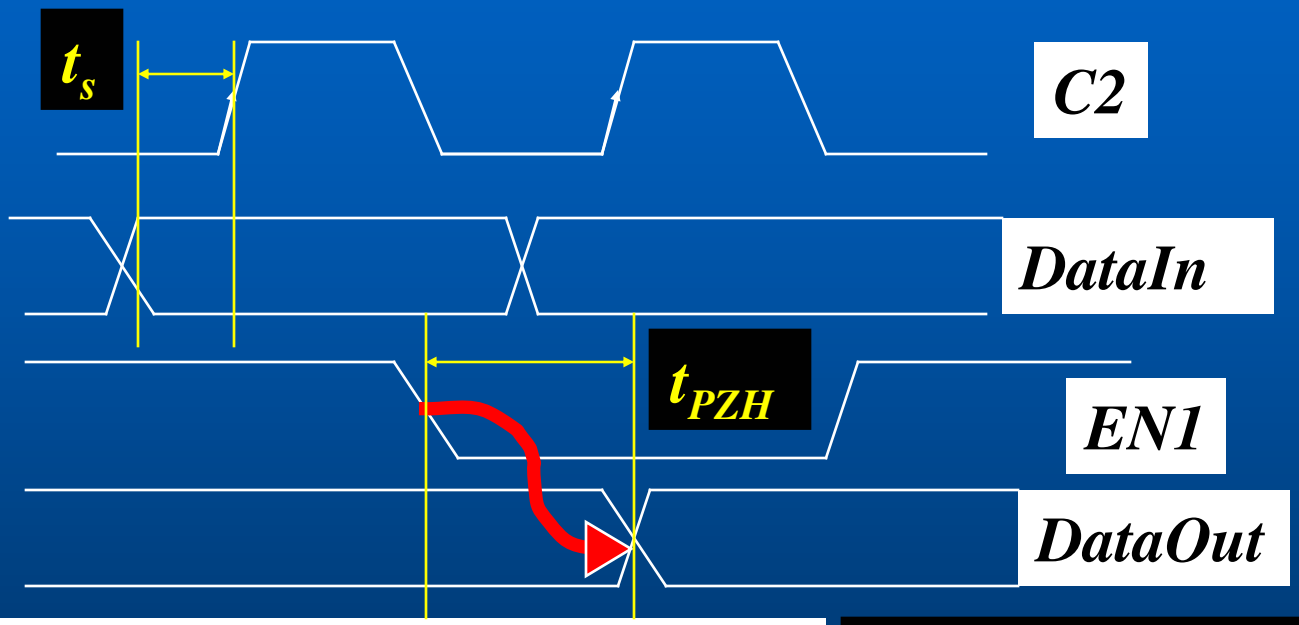
H = High voltage level
h = High voltage level one set-up time prior to the Low-to-High clock transition
L = Low voltage level
l = Low voltage level one set-up time prior to the Low-to-High clock transition
NC = No change
X = Don't care
Z = High impedance "off" state
↑ = Low-to-High clock transition
↕ = Not a Low-to-High clock transition

The Data Sheets of the 74F574 I

Register

LOGIC SYMBOL(IEEE/IEC)





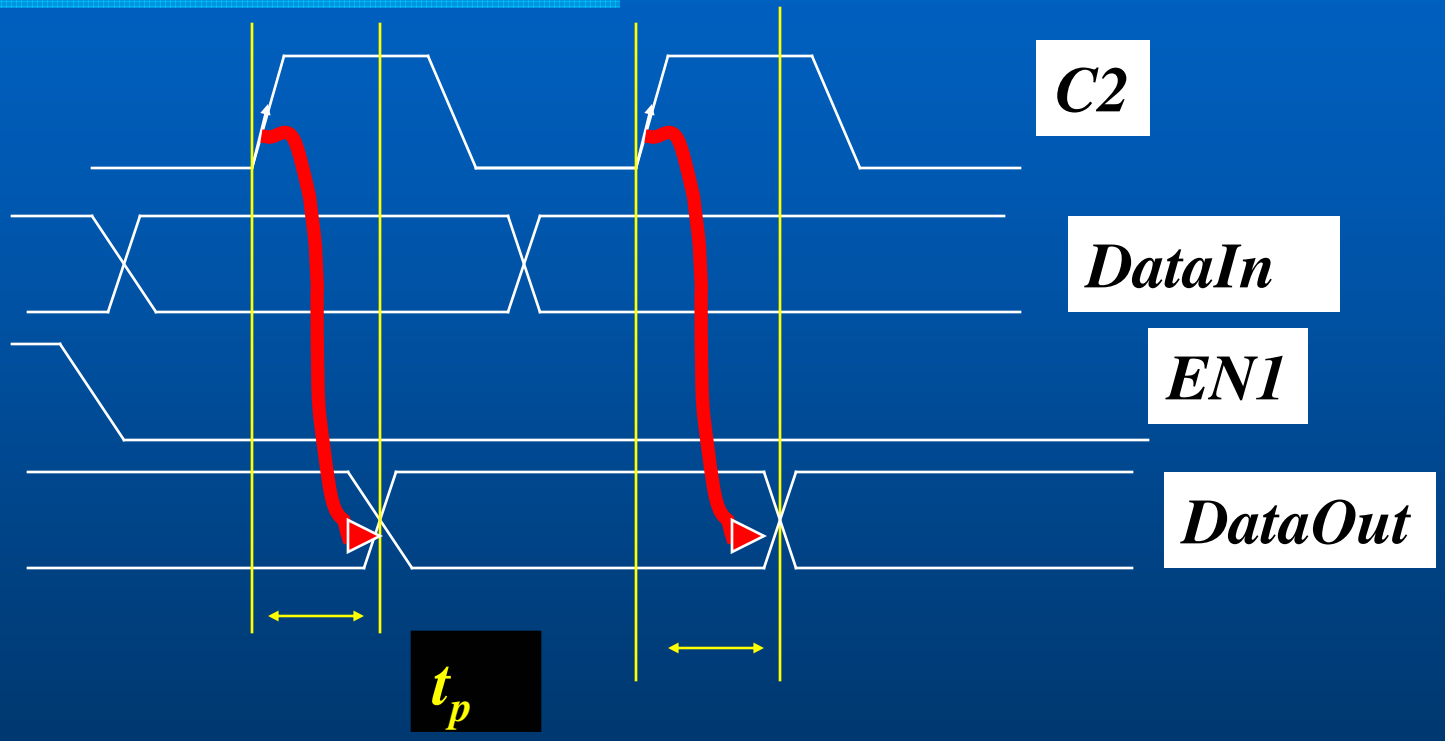
FUNCTION TABLE, 74F574

INPUTS			INTERNAL	OUTPUTS	OPERATING MODE
$\overline{OE}$	CP	D_n	REGISTER	Q_0 - Q_7	
L L	↑ ↑	l h	L H	L H	Load and read register
L	↕	X	NC	NC	Hold
H H	↑ X	D_n X	D_n X	Z Z	Disable outputs

T_s = setup time

t_{PZH} = Output Enable time

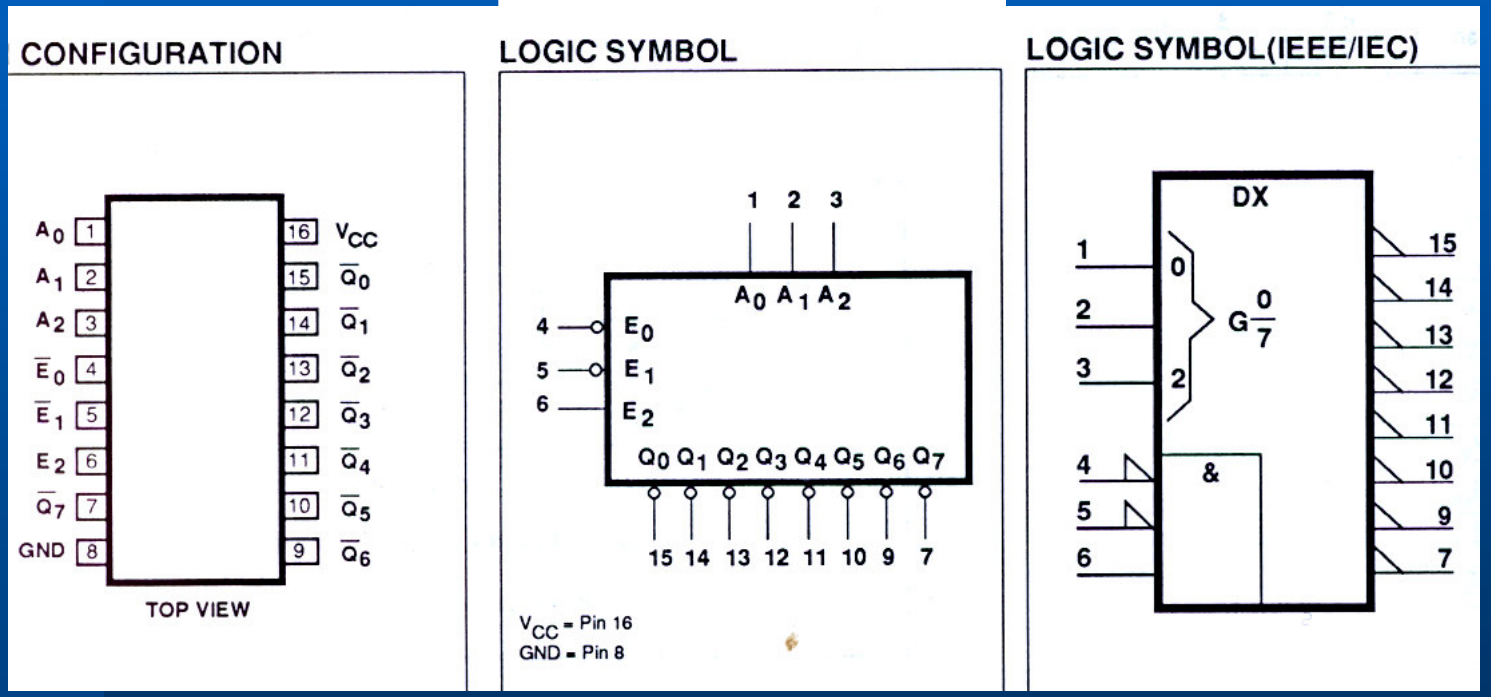
The Data Sheets of the 74F574 II



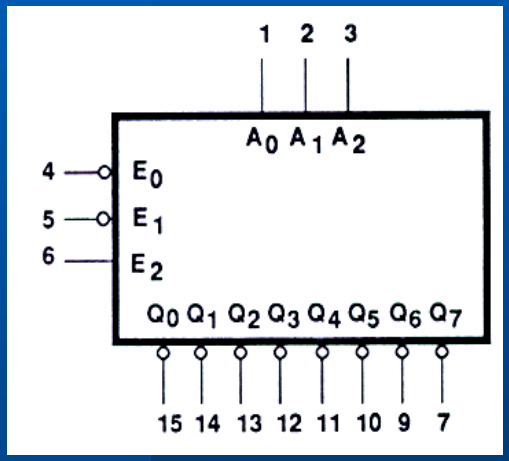
T_p = propagation delay

The data sheet for the 74F138

Address Decoder



What does the 74F138 do (II) ?



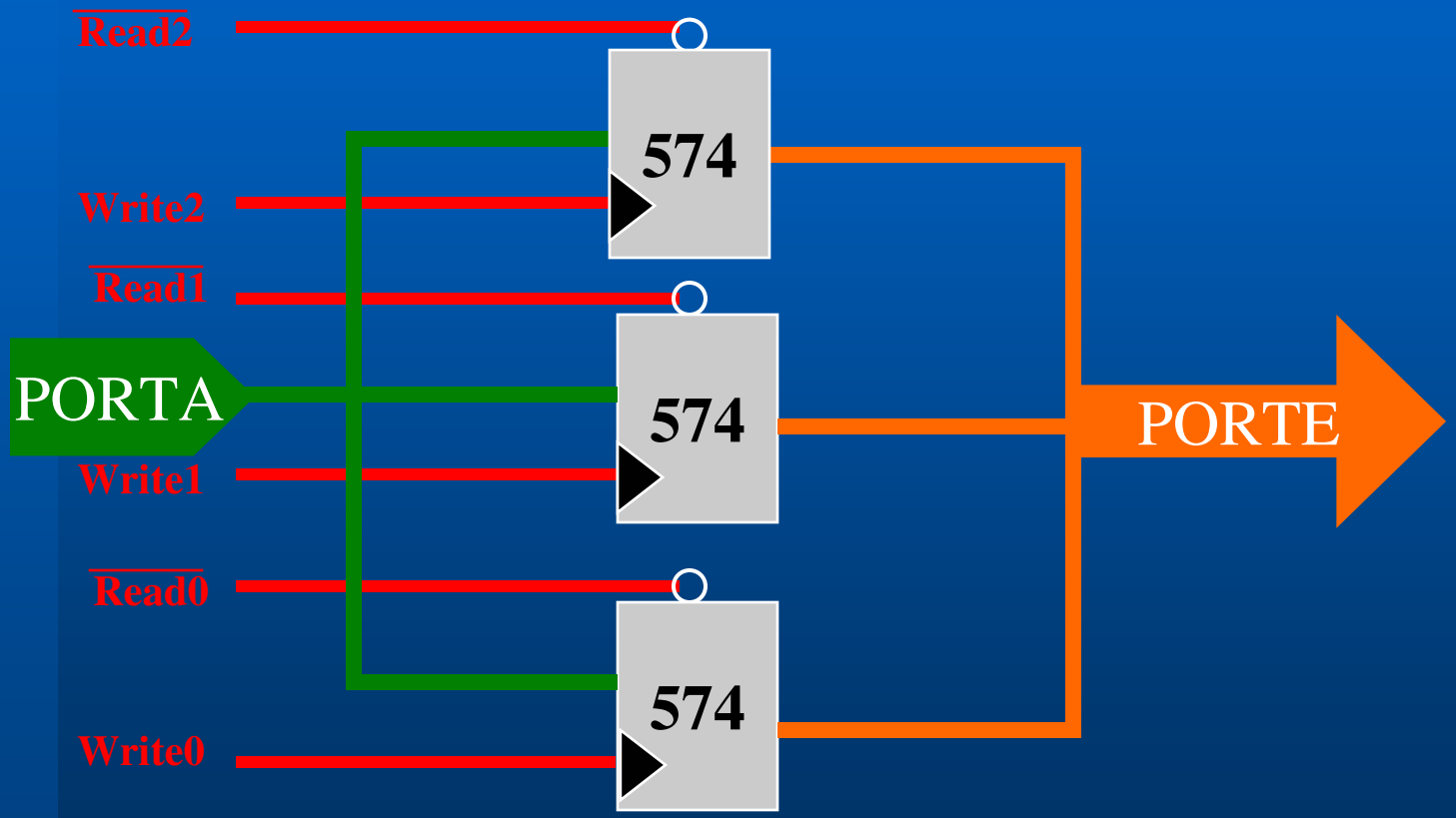
DECODER FUNCTION TABLE

INPUTS						OUTPUTS							
$\overline{E}_0$	$\overline{E}_1$	E_2	A_0	A_1	A_2	$\overline{Q}_0$	$\overline{Q}_1$	$\overline{Q}_2$	$\overline{Q}_3$	$\overline{Q}_4$	$\overline{Q}_5$	$\overline{Q}_6$	$\overline{Q}_7$
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

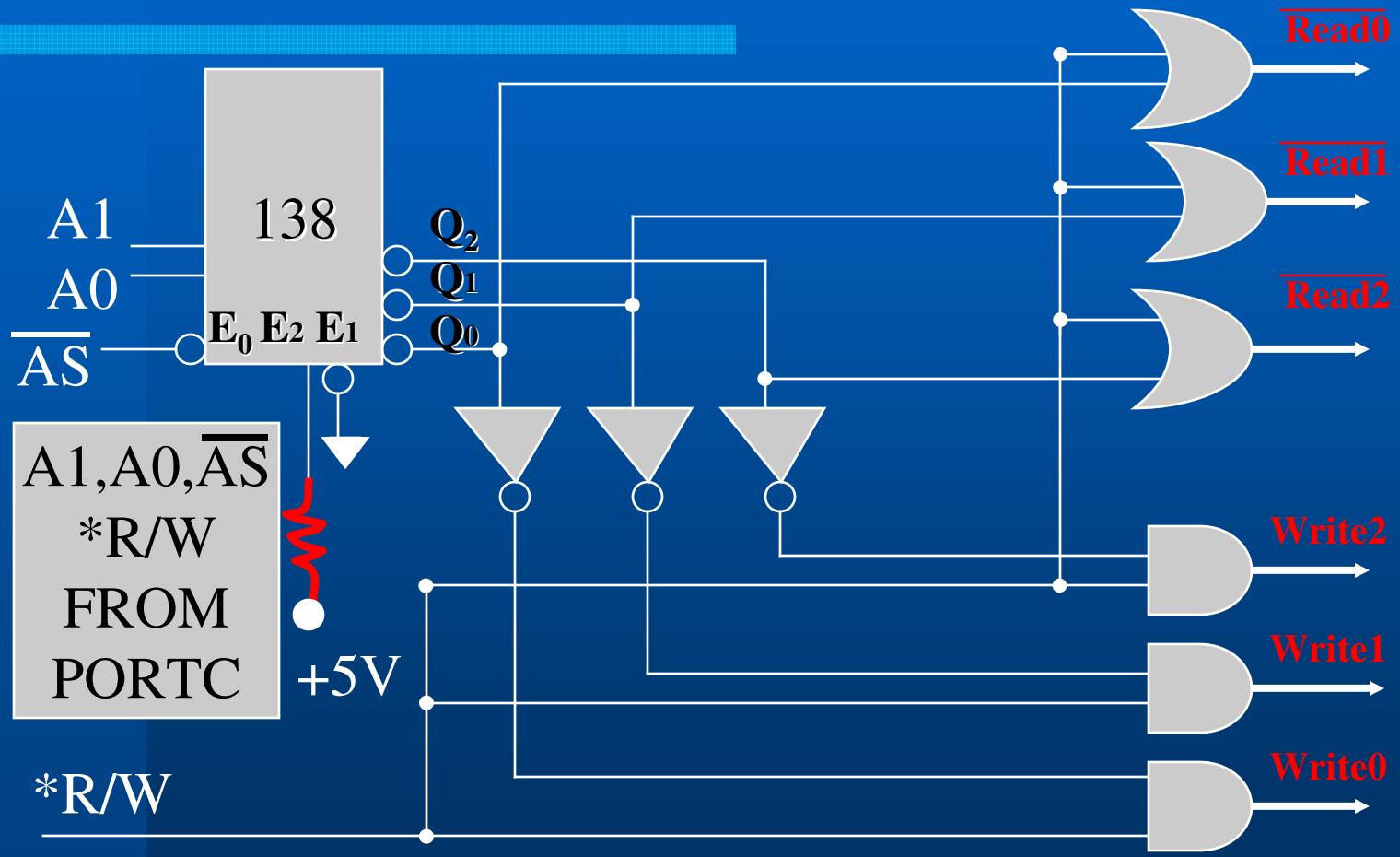
H = High voltage level
L = Low voltage level
X = Don't care

Costas Foudas, Imperial College,
Rm: 508, x47590

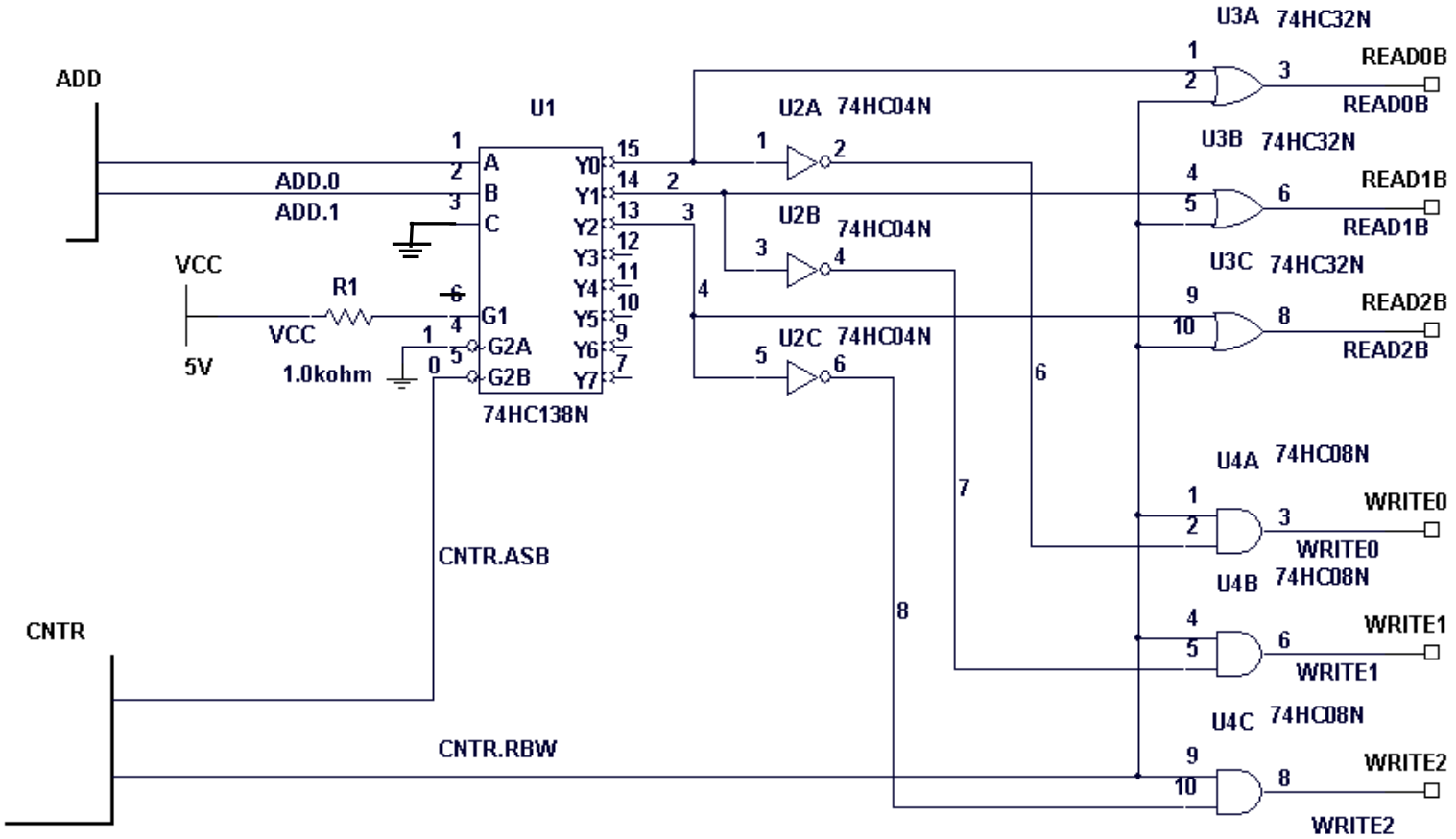
Basic Design idea



Control Bus

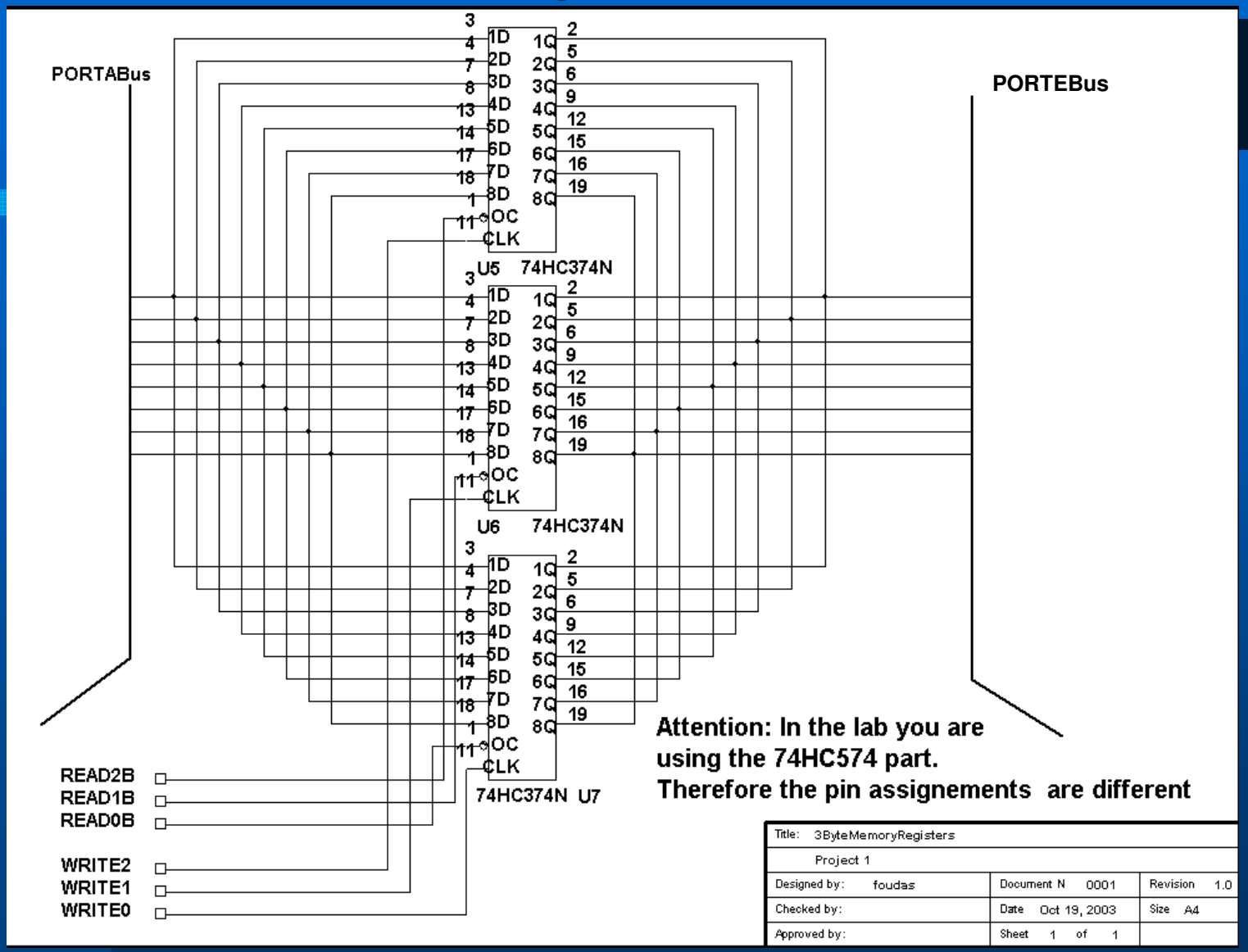

A1
A0
$\overline{AS}$
138
E_0 E_2 E_1
Q_2
Q_1
Q_0
A1,A0,$\overline{AS}$
*R/W
FROM
PORTC
+5V
*R/W
$\overline{Read0}$
$\overline{Read1}$
$\overline{Read2}$
Write2
Write1
Write0

Control Block and Bus

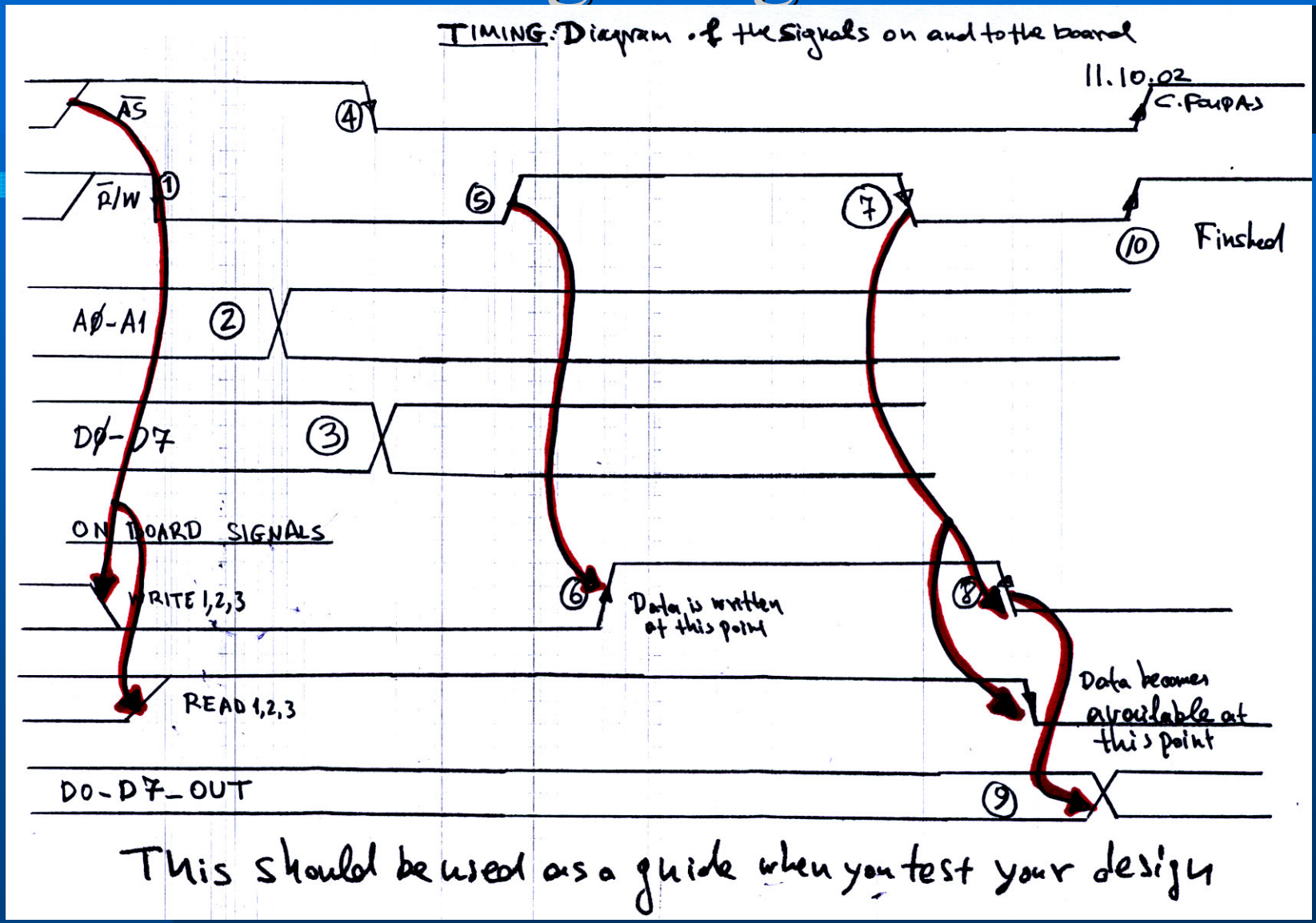


Title: 3 Byte Memory Control		
3d Year MicroProcessor Lab; Imperial College London		
Designed by: Foudas	Document N 0001	Revision 1.0
Checked by:	Date Oct 19, 2003	Size A4
Approved by:	Sheet 1 of 1	

The Registers

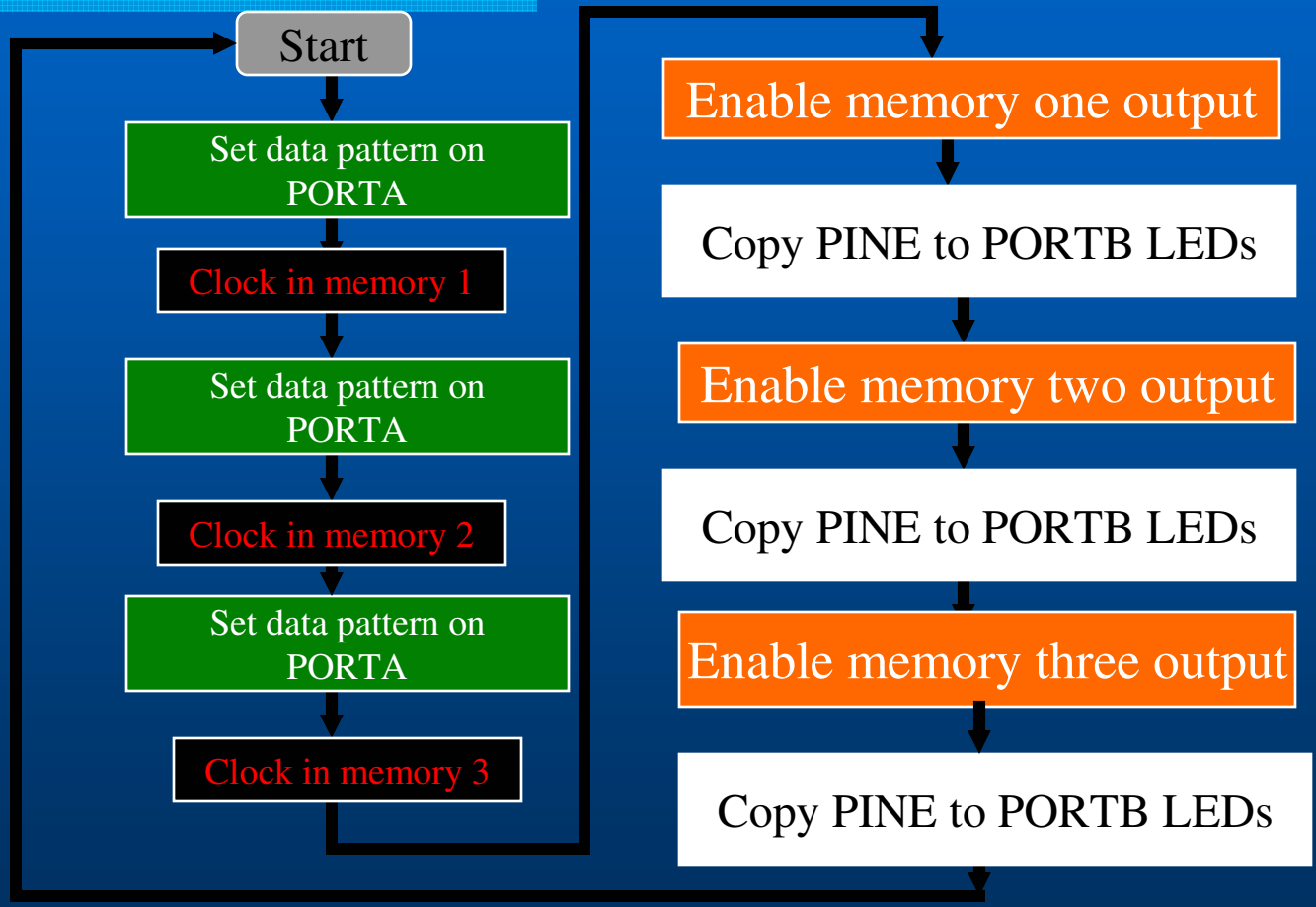


Timing Diagram:



Costas Foudas, Imperial College,
Rm: 508, x47590

The Memory Test Program



Task Plan:

(1) Construct the 3 byte memory and connect it to your ATmage103 Board (always through the special cable-resistor-pack unit).

(2) Do it in steps: **First the Control**, then the first byte and second and finally third.

(3)Write a program with a main section and driver subroutines that read, write and loop data through your memory board.

Some Comments:

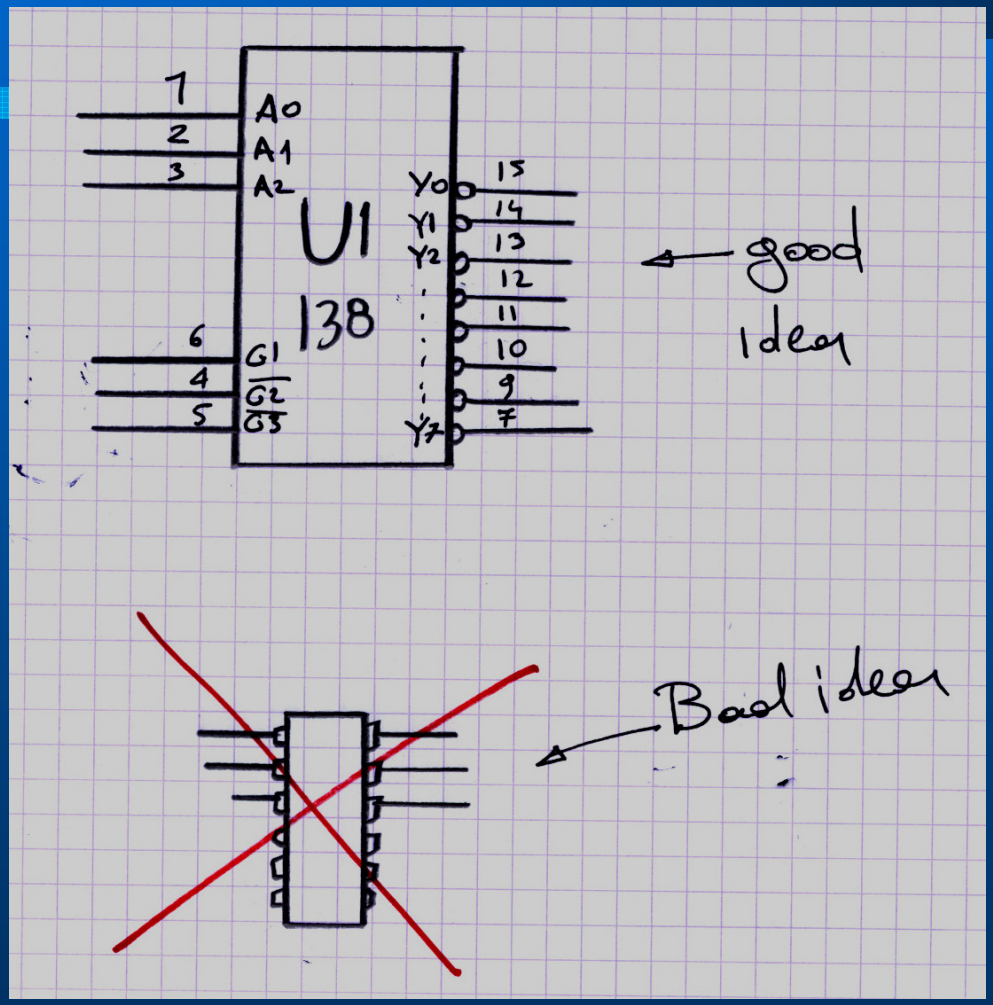
(1) Make a detailed schematic of your device with all ICs (logic diagrams) and the IC pin assignments.

(2) Make sure that you understand your design BEFORE you start building it.

(3) Build the device in pieces which you can check using your software. **Don't build the entire design before testing.**

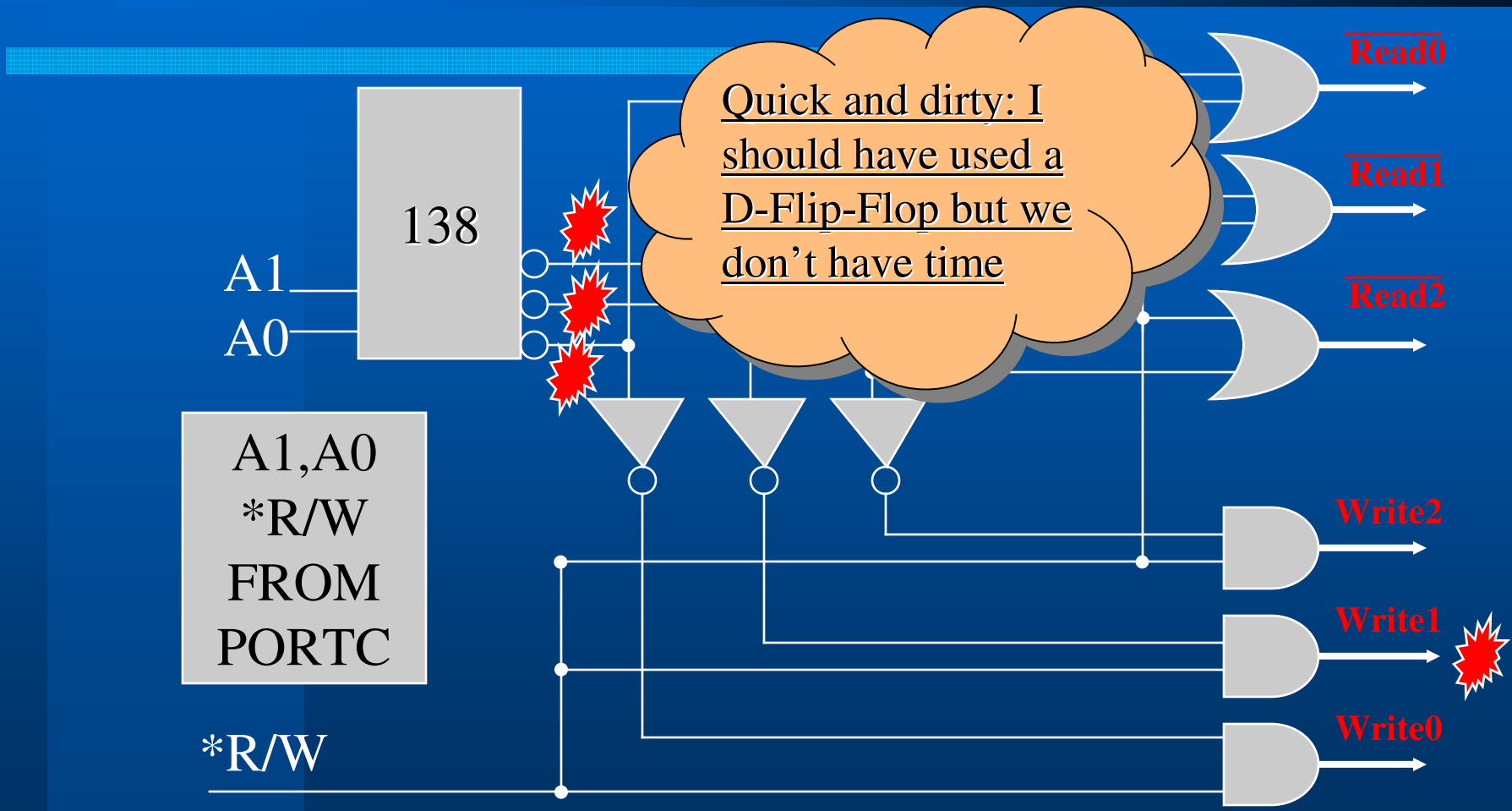
(4) If it does not work use the oscilloscope and the DVM to check what is going on and debug your device.

How to Draw Schematics:



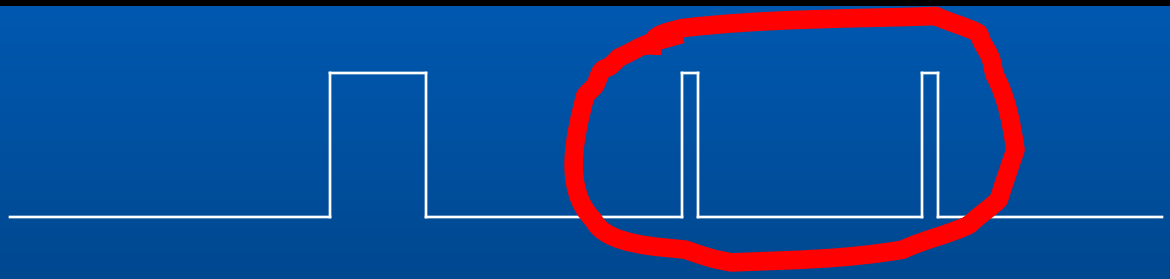
Costas Foudas, Imperial College,
Rm: 508, x47590

The 138 funny output pulses:



Some Comments:

(1) If you try to clock your registers with a pulse that looks like:



Write Pulse

Accidental glitches due to the 138

Then you should expect that you will write 3 times (once when you want) and twice more when/where you DON'T want.

More Comments:

(1) Start by setting the Address Strobe* high and the R*/W high .
(2) Set the correct address and data on your bus.
(3) Bring *R/W for 250 nsec High and then back Low.
(4) Before you connect the outputs together perhaps use LED/Resistors to check if the data is there…
(5) Try reading the data with the ATMEL.

This should result to a successful write of your data at the register you want them