Memory Interface

CECS 460

CSULB

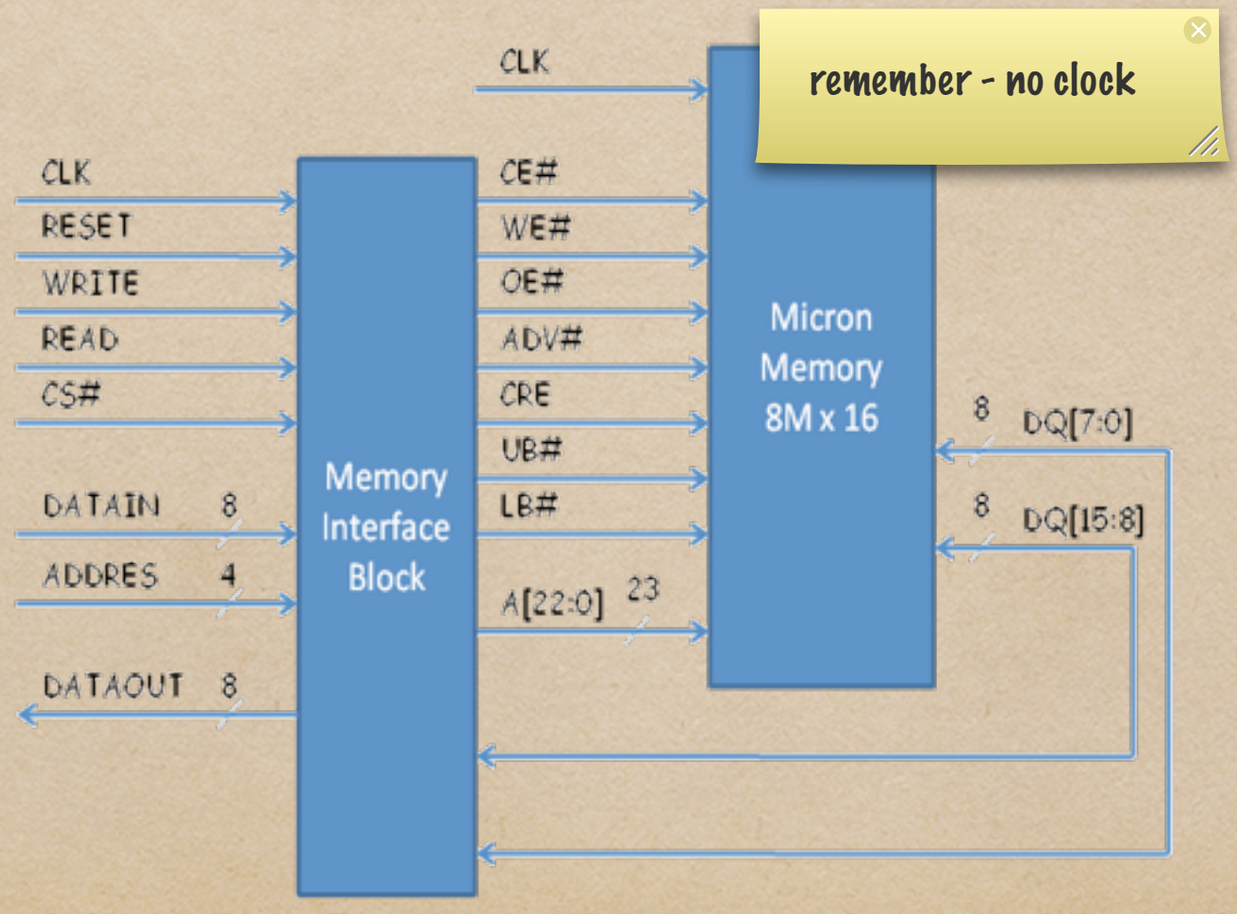
John Tramel

Project Description

- A two-step approach makes the implementation of the MIB a simpler problem to solve
  - First we will make sure we can communicate with the external memory model by simulation
  - Here we will make sure that we can write and read the memory interface making sure we understand the switching characteristics of all of the signals in the interface
  - Second we will design the Memory Interface Block (MIB) usable by our project

Where we want to go

- The final implementation looks like the block diagram below. Our MIB will communicate with the Micron Memory located on our board
- Although the device is a synchronous device, when it is powered up it defaults to an asynchronous mode - that is what we will use

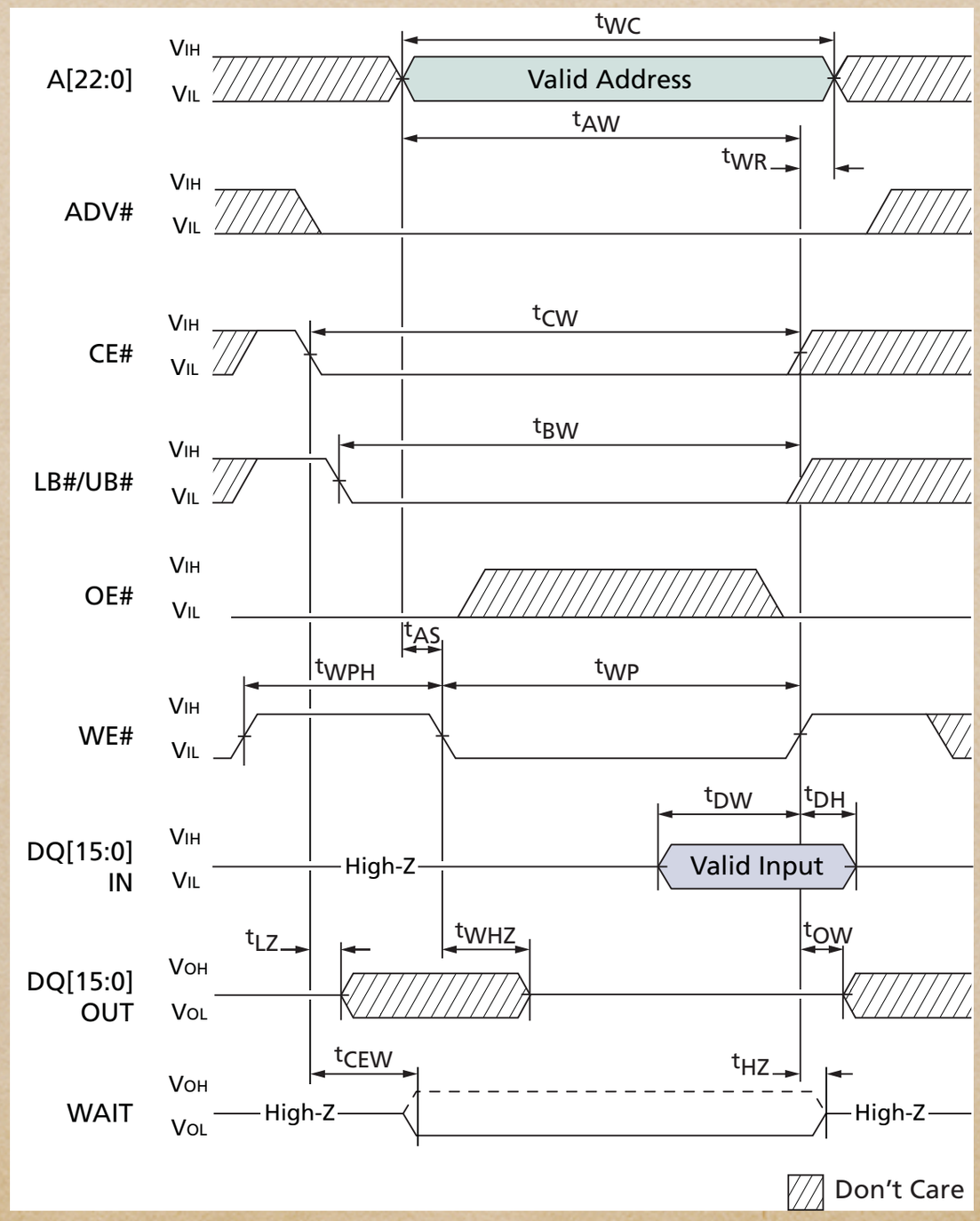


Memory Interface Signals

Signal	Width	In/Out	Description	
CLK	1	Input	System Input	← Always '0'
RESET	1	Input	System Input	
WRITE	1	Input	Write Pulse to write to Memory Interface Block	
READ	1	Input	Read Pulse to read from Memory Interface Block	
CS#	1	Input	Chip Select to select Memory Interface Block	
DATAIN	8	Input	Data to Memory Interface Block	
ADDRESS	4	Input	Address to Memory Interface Block	
DATAOUT	8	Output	Data Back to Processor	
CE#	1	Output	Chip Select of Micron Memory	
WE#	1	Output	Write Enable to Micron Memory	
OE#	1	Output	Output Enable to Micron Memory	
ADV#	1	Output	Always LOW	← Always '0'
CRE	1	Output	Always LOW	← Always '0'
UB#	1	Output	Upper Byte Enable	← Always '0'
LB#	1	Output	Lower Byte Enable	← Always '0'
A	23	Output	Address to Micron Memory	
DQ	16	Inout	Bidirectional Data to/from Micron Memory	

Asynchronous Write

Figure 41: WE#-Controlled Asynchronous WRITE



Asynchronous Write Timing Requirements

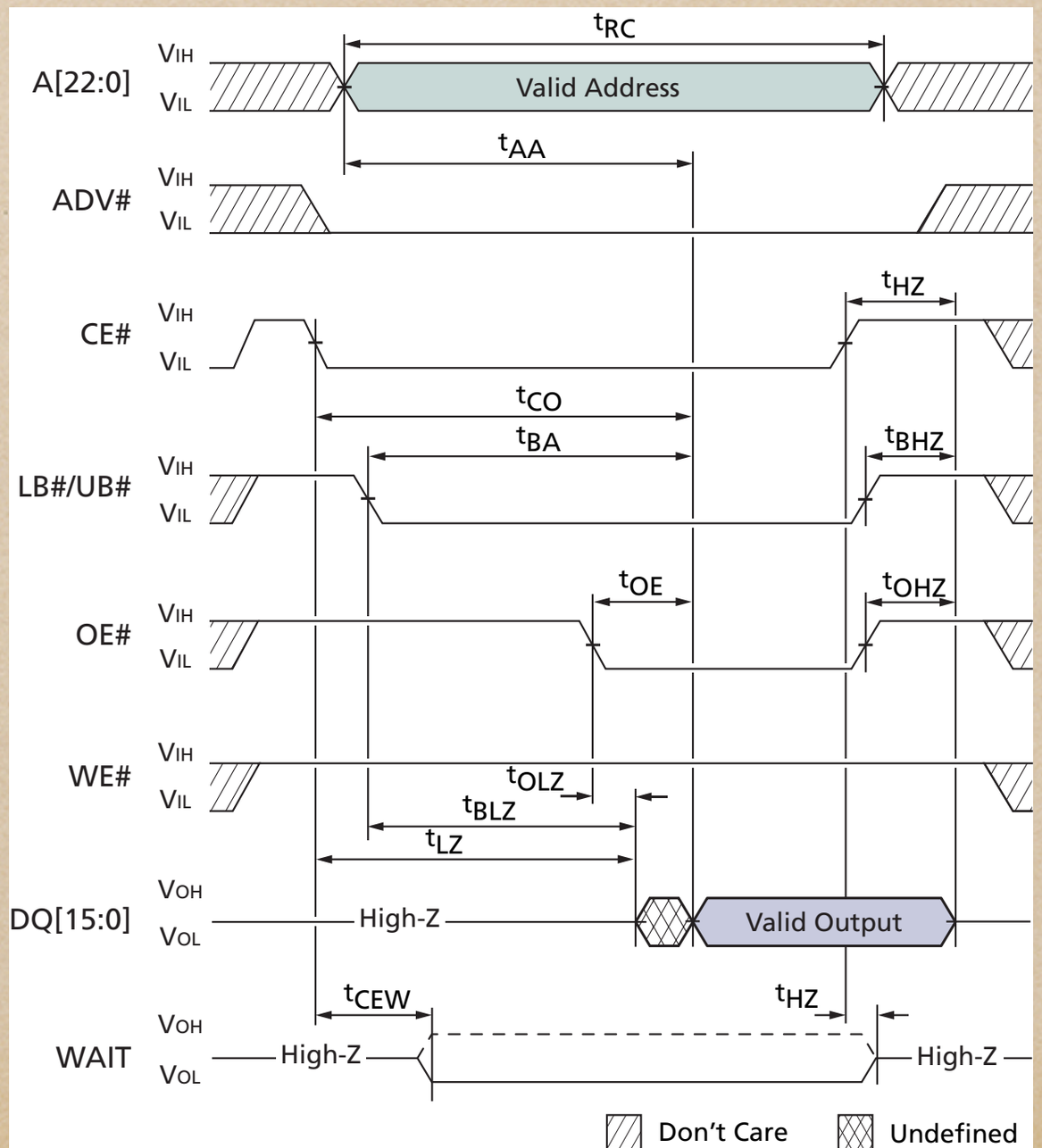
Table 16: Asynchronous WRITE Cycle Timing Requirements

Parameter	Symbol	70ns		85ns		Unit	Notes
		Min	Max	Min	Max		
Address and ADV# LOW setup time	^{t}AS	0		0		ns	
Address HOLD from ADV# going HIGH	^{t}AVH	2		2		ns	
Address setup to ADV# going HIGH	^{t}AVS	5		5		ns	
Address valid to end of WRITE	^{t}AW	70		85		ns	
LB#/UB# select to end of WRITE	^{t}BW	70		85		ns	
CE# LOW to WAIT valid	^{t}CEW	1	7.5	1	7.5	ns	
CE# HIGH between subsequent async operations	^{t}CPH	5		5		ns	
CE# LOW to ADV# HIGH	^{t}CVS	7		7		ns	
Chip enable to end of WRITE	^{t}CW	70		85		ns	
Data HOLD from WRITE time	^{t}DH	0		0		ns	
Data WRITE setup time	^{t}DW	20		20		ns	
Chip disable to WAIT High-Z output	^{t}HZ		8		8	ns	1
Chip enable to Low-Z output	^{t}LZ	10		10		ns	2
End WRITE to Low-Z output	^{t}OW	5		5		ns	2
ADV# pulse width	^{t}VP	5		7		ns	
ADV# setup to end of WRITE	^{t}VS	70		85		ns	
WRITE cycle time	^{t}WC	70		85		ns	
WRITE to DQ High-Z output	^{t}WHZ		8		8	ns	1
WRITE pulse width	^{t}WP	45		55		ns	3
WRITE pulse width HIGH	^{t}WPH	10		10		ns	
WRITE recovery time	^{t}WR	0		0		ns	

Notes:
1. Low-Z to High-Z timings are tested with the circuit shown in Figure 27 on page 36. The High-Z timings measure a 100mV transition from either V_{OH} or V_{OL} toward $V_{CC}Q/2$.
2. High-Z to Low-Z timings are tested with the circuit shown in Figure 27 on page 36. The Low-Z timings measure a 100mV transition away from the High-Z ($V_{CC}Q/2$) level toward either V_{OH} or V_{OL}.
3. WE# LOW time must be limited to ^{t}CEM (4µs).

Asynchronous Read

Figure 30: Asynchronous READ



Asynchronous Read Timing Requirements

Timing Requirements

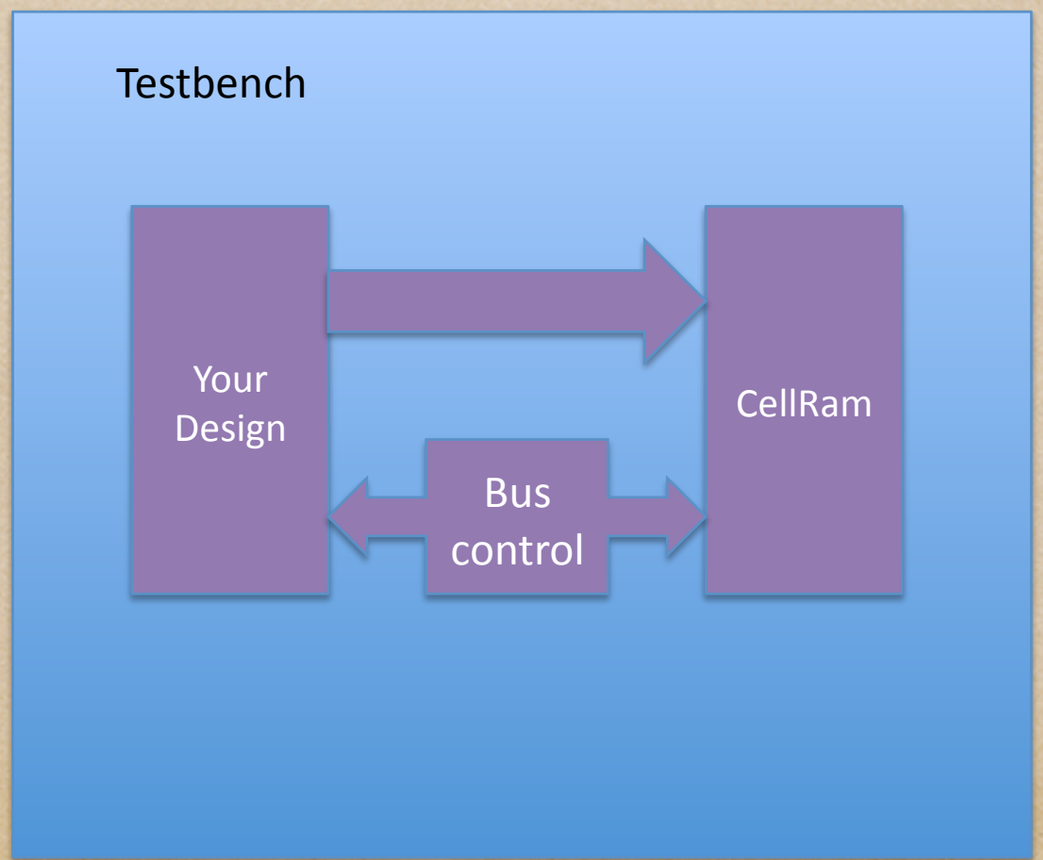
Table 14: Asynchronous READ Cycle Timing Requirements
All tests performed with outputs configured for default setting of one-half drive strength, (BCR[5:4] = 01b)

Parameter	Symbol	70ns		85ns		Unit	Notes
		Min	Max	Min	Max		
Address access time	^{t}AA		70		85	ns	
ADV# access time	$^{t}AADV$		70		85	ns	
Page access time	^{t}APA		20		25	ns	
Address hold from ADV# HIGH	^{t}AVH	2		2		ns	
Address setup to ADV# HIGH	^{t}AVS	5		5		ns	
LB#/UB# access time	^{t}BA		70		85	ns	
LB#/UB# disable to DQ High-Z output	^{t}BHZ		8		8	ns	1
LB#/UB# enable to Low-Z output	^{t}BLZ	10		10		ns	2
Maximum CE# pulse width	^{t}CEM		4		4	µs	3
CE# LOW to WAIT valid	^{t}CEW	1	7.5	1	7.5	ns	
Chip select access time	^{t}CO		70		85	ns	
CE# LOW to ADV# HIGH	^{t}CVS	7		7		ns	
Chip disable to DQ and WAIT High-Z output	^{t}HZ		8		8	ns	1
Chip enable to Low-Z output	^{t}LZ	10		10		ns	2
Output enable to valid output	^{t}OE		20		20	ns	
Output hold from address change	^{t}OH	5		5		ns	
Output disable to DQ High-Z output	^{t}OHZ		8		8	ns	1
Output enable to Low-Z output	^{t}OLZ	3		3		ns	2
Page READ cycle time	^{t}PC	20		25		ns	
READ cycle time	^{t}RC	70		85		ns	
ADV# pulse width LOW	^{t}VP	5		7		ns	

Notes:
1. Low-Z to High-Z timings are tested with the circuit shown in Figure 27 on page 36. The High-Z timings measure a 100mV transition from either V_{OH} or V_{OL} toward $V_{CCQ}/2$.
2. High-Z to Low-Z timings are tested with the circuit shown in Figure 27 on page 36. The Low-Z timings measure a 100mV transition away from the High-Z ($V_{CCQ}/2$) level toward either V_{OH} or V_{OL}.
3. Page mode enabled only.

Phase One

- The first phase will be to write a module that writes and reads the memory
- The memory model is good through 0x200
- The purpose here is to implement an interface to the memory and run a simulation to ensure we meet the timing requirements
- Simulation is the final step of this phase



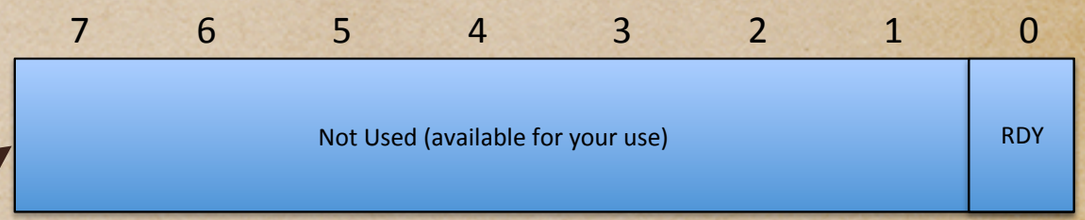
Phase Two

- The interesting part of our project is to have an 8-bit interface communicate with a 16-bit memory, 8M deep
- Our MIB will need to have a number of internal registers that will need to be memory mapped to the processor

Memory Map

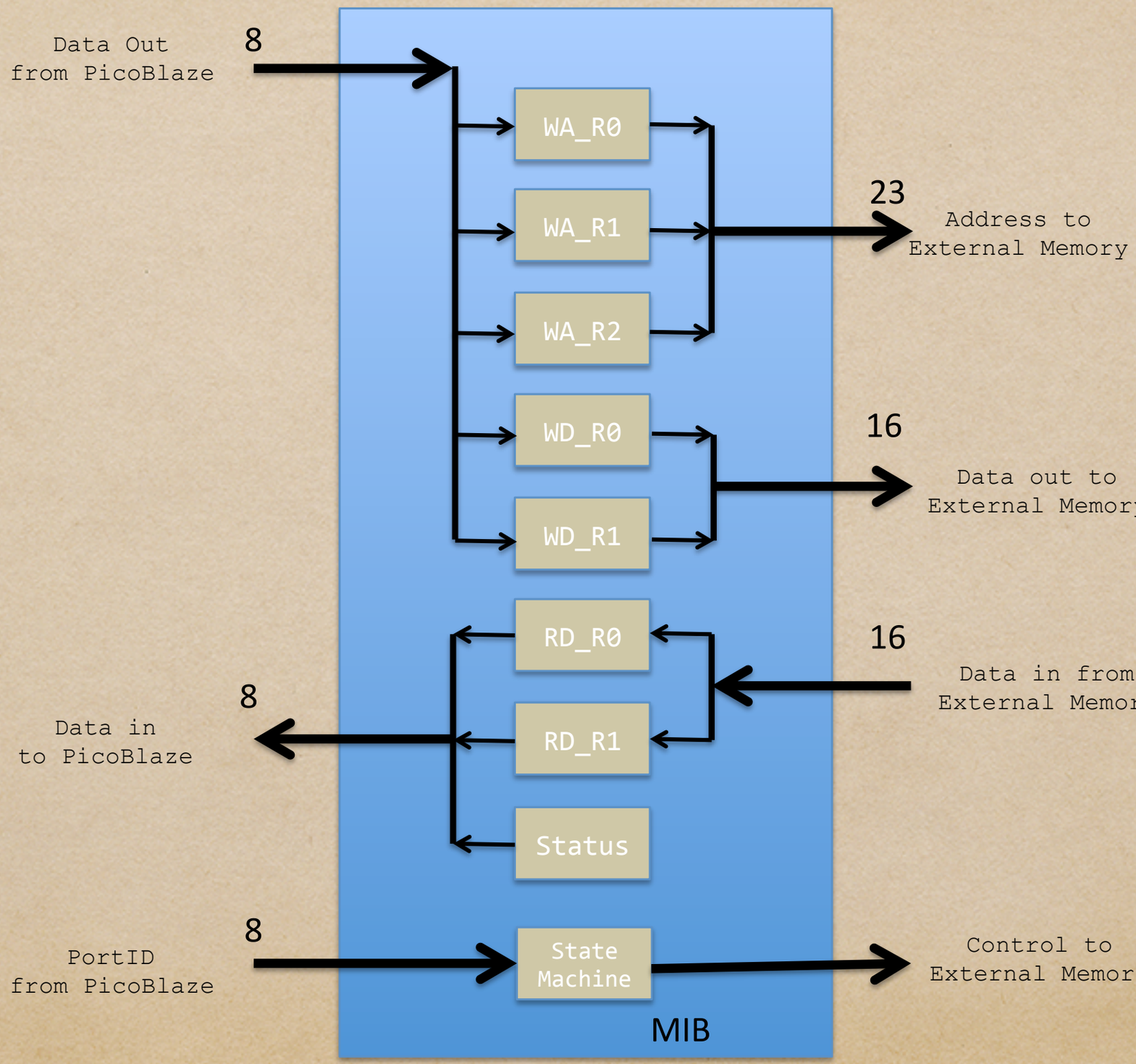
ADDRESS	OPERATION
0	No Operation
1	Write Address Register 0
2	Write Address Register 1
3	Write Address Register 2
4	Write Data Out Register 0
5	Write Data Out Register 1
6	Read Data In Register 0
7	Read Data in Register 1
8	Perform Memory Read
9	Perform Memory Write
A	Read MIB Status
B-F	No Operation

Status Register



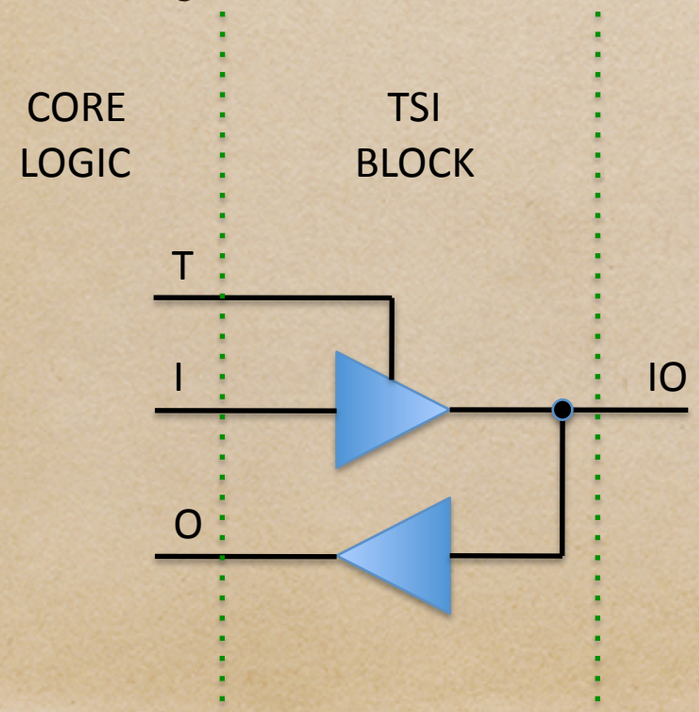
RDY = 1, MIB Ready (current operation complete)
RDY = 0, MIB NOT Ready (currently writing/reading)

Sample Block Diagram



IO Discussion

- All control lines and address lines are unidirectional from the FPGA to the memory device on the Digilent PCB
- The data lines are bidirectional and need to be treated differently - sometimes we will write the memory and sometimes we will read the memory over the same set of data lines
- Remember that inside of the FPGA (your design) the data busses are always unidirectional (no high impedance within the chip)



- The data bus will use the IOBUF from the Spartan 3E library
- The control and address will use the OBUF
- Refer to TSI lecture

Firmware Performance Description

- At startup you should print a banner and follow it with a newline and a prompt
- Every character that is received over the UART interface should still be echoed to the user. Now, starting at location 0, each byte should also be written to the Micron memory
- Remember the Micron memory interface should be written and read 16-bits at a time
- When an "*" is received from the user the data that has been written into the memory should be sent back to the user beginning with the first data byte received. Send a CR/LF first so that the dumped data begins at a new line
- After dumping the memory the controller should be ready to collect data again
- Every time you send a newline to the UART you should follow it with prompt
- When you detect a backspace (BS) you should automatically perform an auto delete to the display ensuring that you do not delete you prompt (no need to alter contents of memory)