ECE 571 – Intro to System Verilog

FINAL PROJECT

TEAM - 14

Under the guidance:

Mark Faust.

Team Composition:

Deepak Kumar Dupati

Krishna Priya Narra

Sai Tulasi Prasad Dumpala

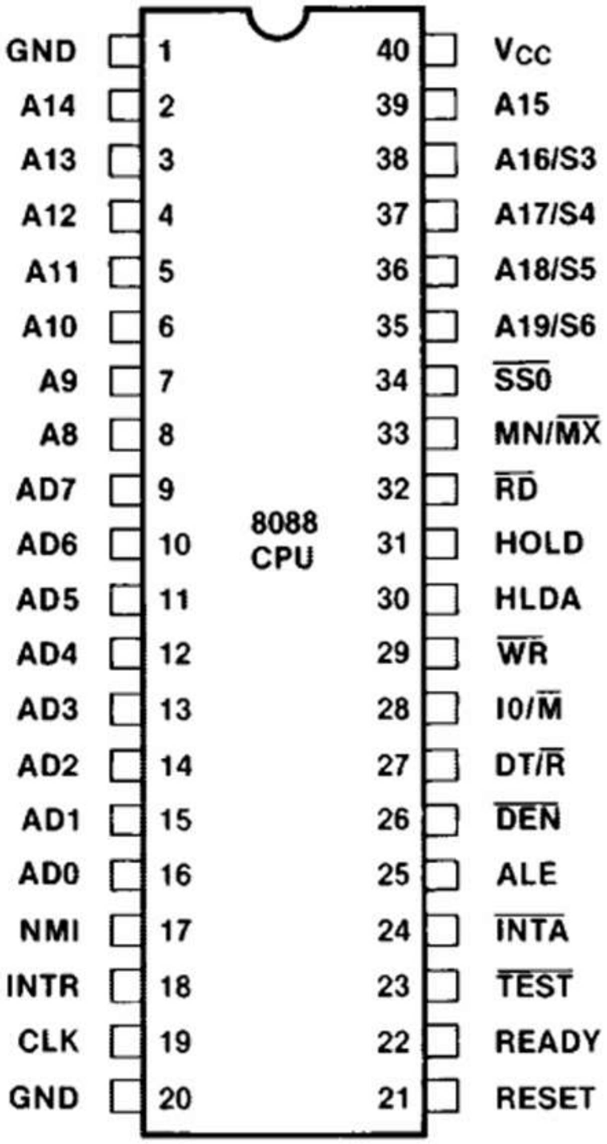
Vineeth Rao Vala

ANALYSIS OF THE PROJECT:

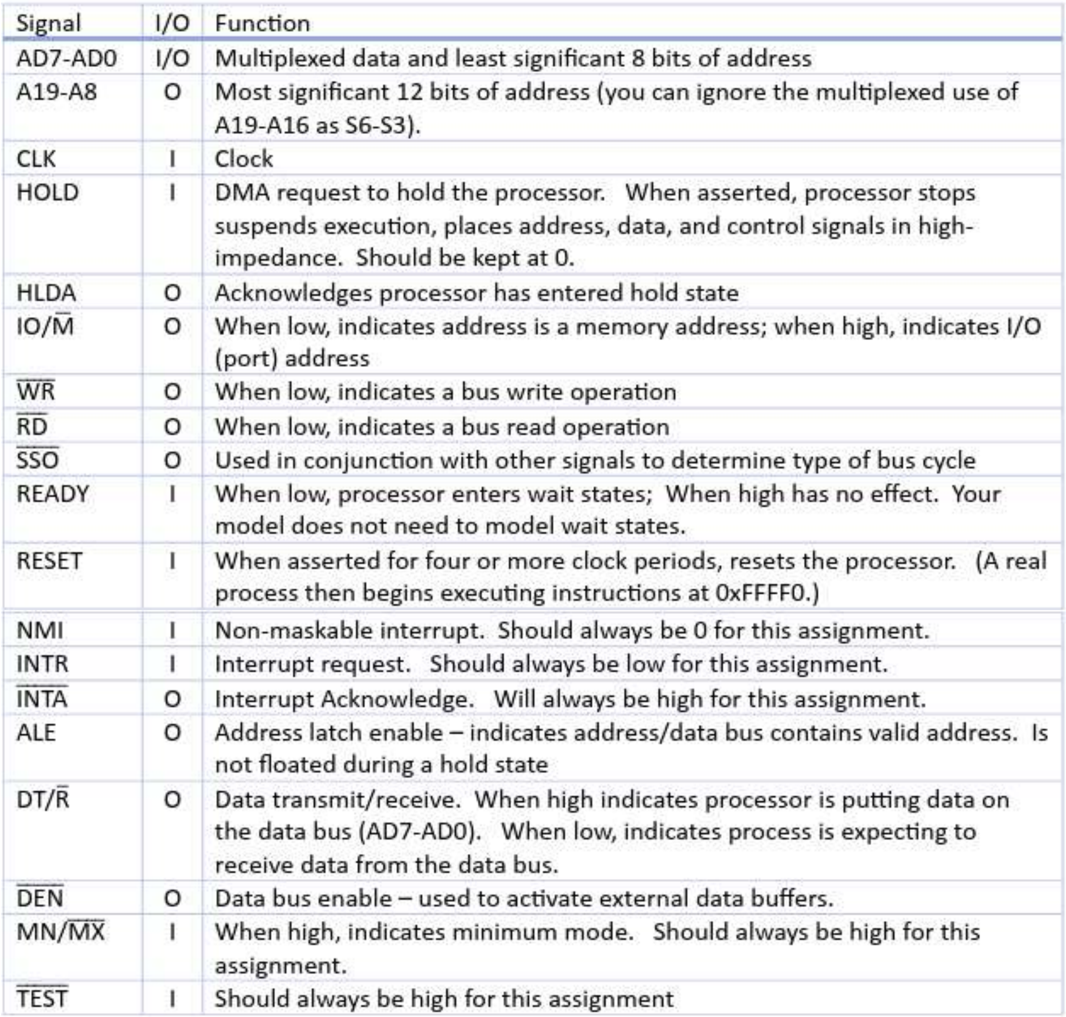
- Developed I/O and Memory modules and seamlessly integrated them with the Intel 8088 processor bus.
- Implemented the interface for the 8088 processor.
- Utilized `$readmemh` to initialize memory for design verification purposes.
- Analyzed functional models of the bus to facilitate integration with the I/O memory module and comprehend processor signals.
- Designed a Finite State Machine to depict memory read and write operations.
- Established and utilized interfaces effectively.
- Acquired proficiency in System Verilog and adopted standard industry methodologies.
- Executed read and write operations on both Memory and I/O modules, incorporating various test cases, including corner cases, to ensure comprehensive testing.

BRIEF INTRO ABOUT 8088:

PIN DIAGRAM OF 8088:

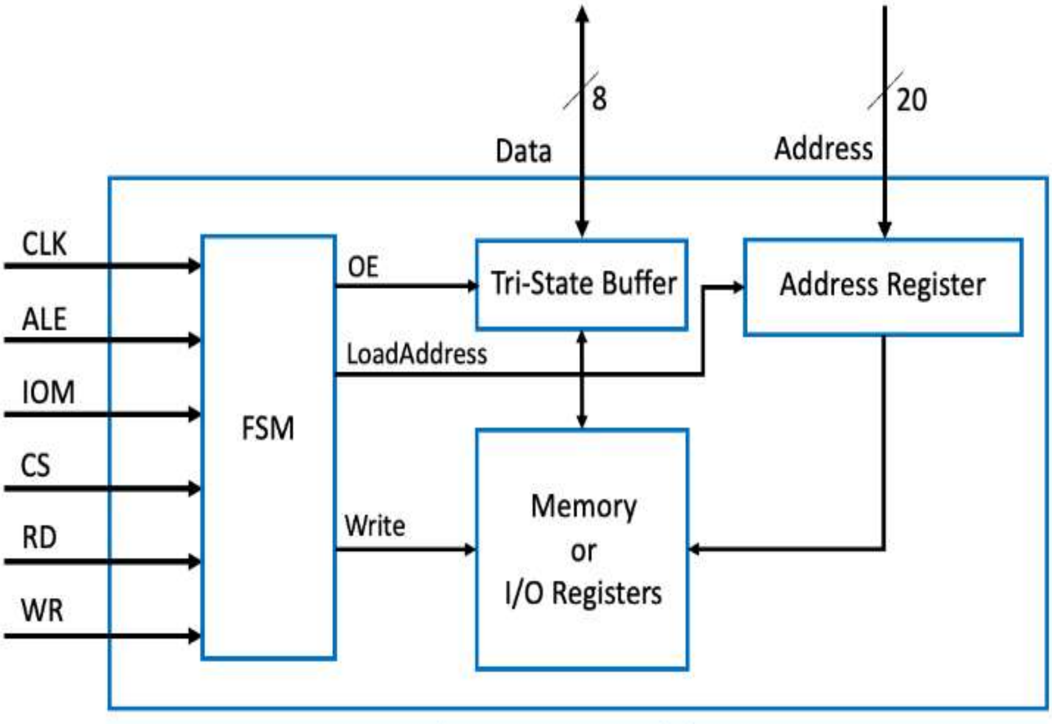


PIN DESCRIPTION OF 8088:



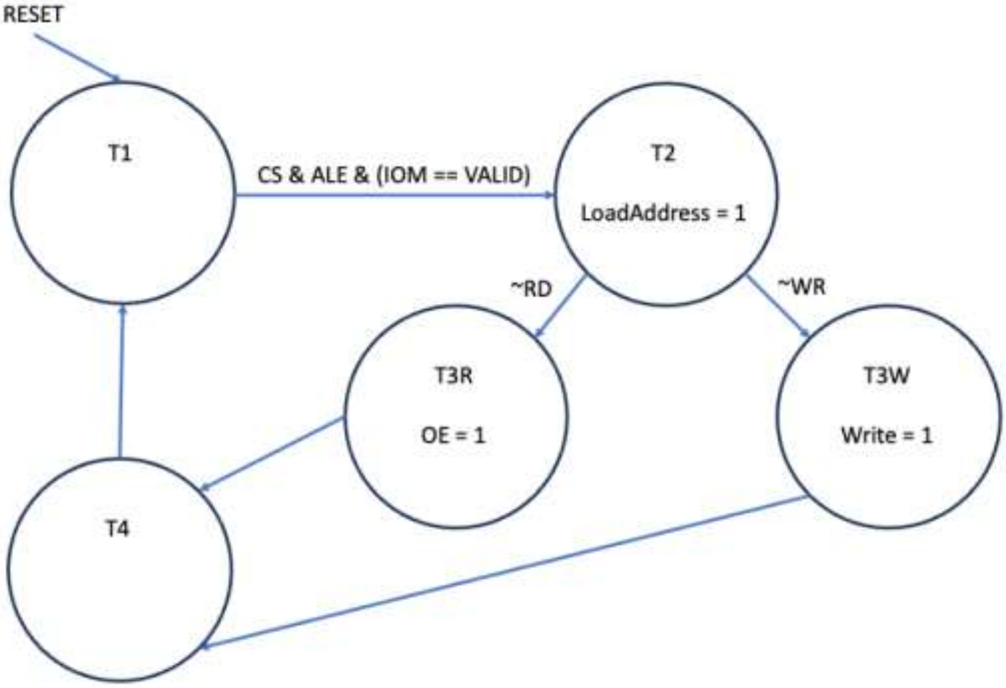
Signal	I/O	Function
AD7-AD0	I/O	Multiplexed data and least significant 8 bits of address
A19-A8	O	Most significant 12 bits of address (you can ignore the multiplexed use of A19-A16 as S6-S3).
CLK	I	Clock
HOLD	I	DMA request to hold the processor. When asserted, processor stops suspends execution, places address, data, and control signals in high-impedance. Should be kept at 0.
HLDA	O	Acknowledges processor has entered hold state
IO/$\overline{\text{M}}$	O	When low, indicates address is a memory address; when high, indicates I/O (port) address
$\overline{\text{WR}}$	O	When low, indicates a bus write operation
$\overline{\text{RD}}$	O	When low, indicates a bus read operation
$\overline{\text{SSO}}$	O	Used in conjunction with other signals to determine type of bus cycle
READY	I	When low, processor enters wait states; When high has no effect. Your model does not need to model wait states.
RESET	I	When asserted for four or more clock periods, resets the processor. (A real process then begins executing instructions at 0xFFFF0.)
NMI	I	Non-maskable interrupt. Should always be 0 for this assignment.
INTR	I	Interrupt request. Should always be low for this assignment.
$\overline{\text{INTA}}$	O	Interrupt Acknowledge. Will always be high for this assignment.
ALE	O	Address latch enable – indicates address/data bus contains valid address. Is not floated during a hold state
DT/$\overline{\text{R}}$	O	Data transmit/receive. When high indicates processor is putting data on the data bus (AD7-AD0). When low, indicates process is expecting to receive data from the data bus.
$\overline{\text{DEN}}$	O	Data bus enable – used to activate external data buffers.
MN/$\overline{\text{MX}}$	I	When high, indicates minimum mode. Should always be high for this assignment.
$\overline{\text{TEST}}$	I	Should always be high for this assignment

SKETCHING a BLACK BOX:



I/O or Memory module

STATE DIAGRAM FOR FSM:



VERIFICATION OF DESIGN:

- Trace files are created using a separate SV module, and memory is initialized using $readmemh.
- Different test cases are checked using busops.txt file, design is verified, and waveforms are observed.
- Parameters are declared for specifying the trace file to initialize the memory.
- Wave forms are observed using Questasim and the expected functioning of Read and Write operations is verified by confirming that the waveforms conform to the bus timings and the timings specified in busops.txt.

DIFFERENT TEST CASES:

Test Case 1:

100	M	R	0xFFFFF
200	M	W	0x33333
300	I	R	0x1F00F
400	I	W	0x200F0
401	I	R	0x300FF

Initially, we initialized the data for all addresses on two memory and IO modules. In the test scenario described above, we instantiated two memory and two IO modules along with the 8088 modules in the top module. We verified both Read and Write operations. For the **Write operation, data was written** into the Memory over 4 clock cycles. During the **Read operation, data was read** and transmitted to the data bus. Additionally, we included two consecutive Write and Read operations on the IO module at cycles 400 and 401, both of which were executed successfully.

Test Case 2:

150	I	R	0xFF1F
250	I	R	0x11DF

Memory modules are instantiated, and I/O modules are not instantiated. For the given case, the read operations are intended to be performed on I/O devices. So, the memory module did not respond, and the data bus was **tristaed**.

<u>Test Case 3</u>:

100	M	R	0xFF1F
200	M	W	0x15454
300	M	R	0x15454
400	M	R	0xFF0F
500	M	W	0xFFFFF
600	M	R	0xFFFFF
700	I	R	0xFF10
800	I	W	0xFF0F
900	I	R	0xFF0F
950	I	R	0x1FD1
1000	I	R	0x1FFF
1050	I	R	0x1FFF

In the described scenario, we conducted write and read operations for each memory and IO module, performing three reads and writes for each module. Furthermore, we specifically targeted the same address for both write and read operations to ensure the accuracy of data read and written to that specific address. This was done to **validate the correctness** of the write and read operations executed at the same address.

<u>Test Case 4:</u>

100	I	W	0xFF1F
200	I	W	0x11DF

In the described test scenario, only two Memory modules were instantiated to assess the execution of operations. Since the operations were designed for IO devices, the **Memory modules did not respond** to them and were effectively disregarded.

Test Case 5:

100	M	W	0xFFF1F
200	M	W	0x11DFF

In the described test scenario, only two I/O modules were instantiated to assess the execution of operations. Since the operations were designed for memory devices, the **I/O modules did not respond** to them and were effectively disregarded.

Test Case 6:

350	M	R	0xFF1FF
450	M	R	0x11DFF

I/O modules are instantiated, and memory modules are not instantiated. For the given case, the read operations are intended to be performed on memory devices. So, the I/O module did not respond, and the data bus was **tristaed.**

Different test cases are given, and the wave forms are observed using Questasim and the expected functioning of Read and Write operations is verified by confirming that the waveforms conform to the bus timings and the timings specified in busops.txt.

All the files from time to time are updated in GitHub:

https://github.com/Deepak-1159/SV_PROJECT_TEAM-14