

NMK206 - Computer Architecture

This course is *Computer Architecture*, offered by the Faculty of Electronics Engineering & Technology (FKTEN) for Electronic Engineering Technology programs.

Download [ModelSim 20.1.1.720 \(Intel FPGA Starter Edition\) setup \(SHA-1 checksum\)](#). Linux users can try [this \(SHA-1 checksum\)](#).

Video Guide(s)

[YouTube Playlist](#)

ModelSim: Installation Note: Available on YouTube . Note: Local copy available.	 ModelSim Installation
ModelSim: Create Project Note: Available on YouTube . Note: Local copy available.	 ModelSim: Create New Project
ModelSim: Simulate Logic Circuit Note: Available on YouTube . Note: Local copy available.	 ModelSim: Simulate Logic Circuit

Announcements

[202503023] Welcome to NMK206 info page (for 007 lab sessions only)!

Lab Session

I am using [Takahashi Method](#) for these slides (Actually, I broke that method by adding diagrams and long codes because I think my students need them). You will find them hard to understand if you do



not attend my sessions. So, that is the 'advantage' I gave to those who actually listen in class

- Lab Briefing
 - Slides
- Intro to CAD Tools and HDL
 - Slides
 - Online Session (Video)
- Verilog Basics
 - Slides
 - Online Session (Video)
- Combinational Logic
 - Online Session (Video) (P1:Partial... was stopped due to low number of students)

note: these are from 202324s2 academic session. they should be very similar this semester, but some details may be added or removed. i usually post an updated version at the end of the week.

- Combinational Logic
 - Slides (P1)
 - Slides (P2)
 - Slides (P3)
 - Slides (P4)
 - Slides (P5)
- Sequential Logic
 - Slides (P1)
 - Slides (P1x)
 - Slides (P2)
- State Machine
 - Slides
- Simple Digital System
 - Slides

Verilog Coding Rule

This is the coding rule that I impose on my students. You will be penalized during assessments if it is not adhered to.

1. One file for one module
 - **RULE:** 1 file 1 module
2. **File name must be the same as module name**
 - **RULE:** file name === module name (.v)
3. All circuit (module) must have a testbench (tb)
 - tb is a also module (so, separate file)
 - **RULE:** All module MUST have a testbench
 - **RULE:** Tb name === module name + _tb
4. Use Verilog95 module declaration
 - Port list contain names only (separate input/output declaration)
 - Port connection(s) MUST BE specified using ordered list
 - **RULE:** Port connection(s) by ordered list ONLY!
5. Modules for combinational logic should only use wire/assign statements

- reg/always reserved for sequential logic and testbench modules
 - **RULE:** comb. logic use assign/wire only!
6. Only basic logic gates are allowed
- Can only use AND/OR/INV in your logic implementation
 - XOR logic is allowed for **lab project only**
 - **RULE:** allowed operators AND, OR, INV
7. Assign statements can only have ONE binary operator
- Multiple bitwise inverts (~) are ok (they are unary operators)
 - **RULE:** 2-input logic gates ONLY
8. ALL nets (wire/reg) MUST BE declared.
- some compiler may allow using without declaration
 - for my assessments, they MUST BE declared
 - **RULE:** All signals @wire must be declared!

From:

<http://azman.unimap.edu.my/dokuwiki/> - Azman @UniMAP

Permanent link:

<http://azman.unimap.edu.my/dokuwiki/doku.php?id=archive:nmk206&rev=1743996700>

Last update: 2025/04/07 11:31

