



Quick Start to Hardware Simulation

TA: Ting Wang

d10943014@ntu.edu.tw

Date: Nov. 22, 2021



Outline

- ◆ Connect to server
- ◆ Basic Linux commands
- ◆ Editors
- ◆ Lab



EE2-231 Servers

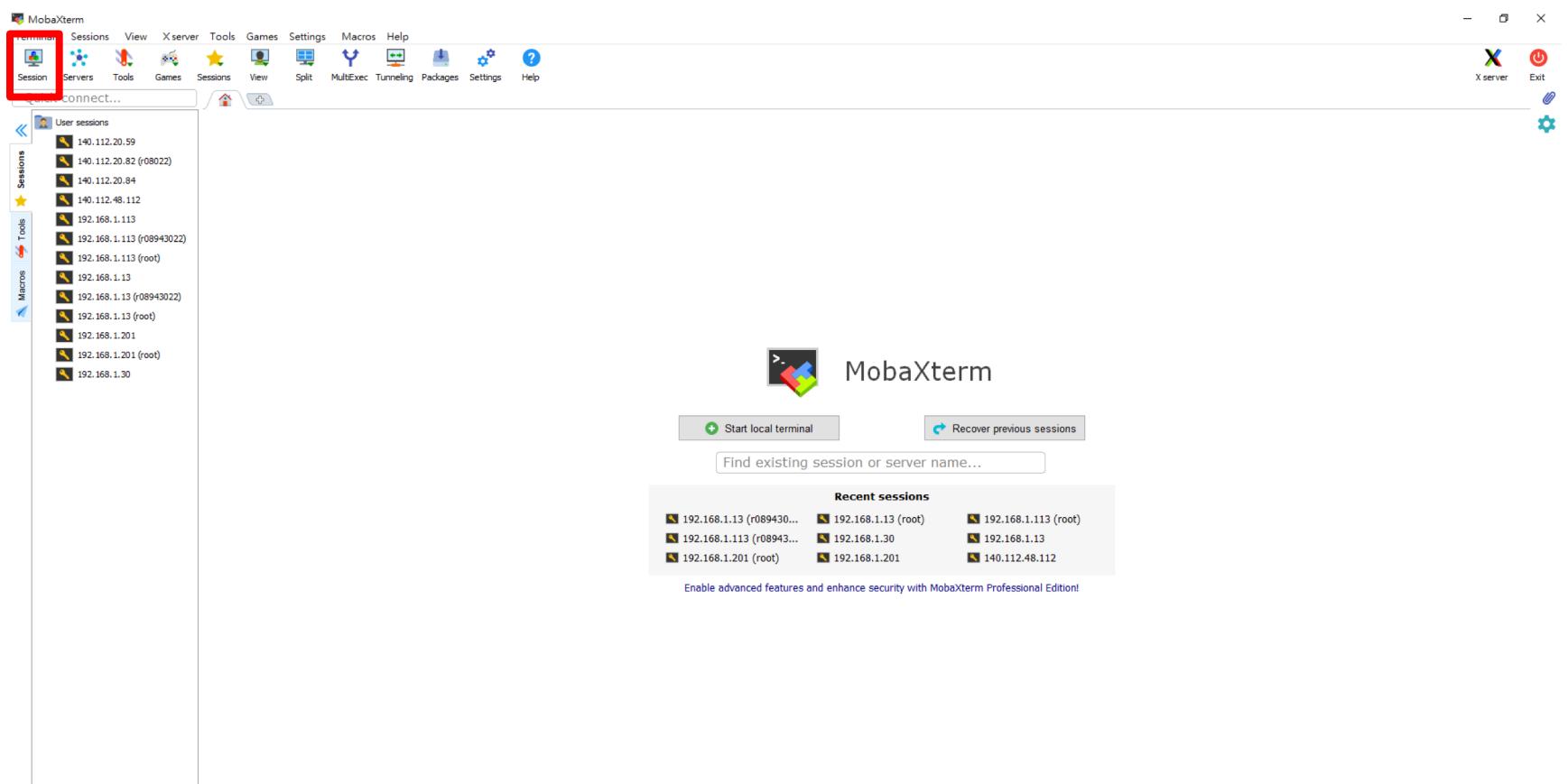
- ◆ <http://cad.ee.ntu.edu.tw>

* IP	Name	Type	CPU	CPU Clock	Memory	OS
59	cad16	IBM X3500	Intel Xeon 64	1.2 GHz * 24	40 GB	CentOS 6.10
60	cad17	IBM x260	Intel Xeon 64	1.2 GHz * 24	32 GB	CentOS 6.10
64	cad21	DELL R620	Intel Xeon 64	2 GHz * 32	48 GB	CentOS 6.9
66	cad23	DELL R620	Intel Xeon 64	2 GHz * 16	96 GB	CentOS 6.9
70	cad27	DELL R640	Intel Xeon Silver 4110	2.1 GHz * 32	128 GB	CentOS 6.10
71	ead28	IBM e336	Intel Xeon 64	3.2 GHz	5 GB	
72	cad29	Dell R730	Intel Xeon 64	2.3GHz * 40	336 GB	CentOS 6.10
73	cad30		Intel Xeon 64	1.6GHz * 16	12 GB	CentOS 6.9
74	cad31	DELL R620	Intel Xeon 64	2.2 GHz * 32	48 GB	CentOS 6.9
76	ead33	IBM X3400	Intel Xeon 64	2.4 GHz * 16	100 GB	CentOS 6.9
77	cad34	IBM X3650	Intel Xeon 64	1.6 GHz * 16	16 GB	CentOS 6.9
78	ead35	SUN V20z	AMD Opteron	2.4 GHz * 2	4 GB	
79	ead36	SUN V20z	AMD Opteron	2.4 GHz * 2	4 GB	
80	ead37	SUN V20z	AMD Opteron	2.4 GHz * 2	4 GB	
81	cad38	IBM X3650	Intel Xeon 64	2.4 GHz * 16	96 GB	RHEL 5
82	cad39	IBM X3650	Intel Xeon 64	1.2 GHz * 24	40 GB	CentOS 6.9
83	cad40	IBM X3550	Intel Xeon 64	2 GHz * 24	132 GB	CentOS 6.9
84	cad41	IBM X3550	Intel Xeon 64	2.1 GHz * 24	132 GB	CentOS 6
85	cad42	IBM X3500	Intel Xeon 64	2 GHz * 24	32 GB	CentOS 7
86	cad43	IBM X3550	Intel Xeon 64	2.6GHz * 32	148 GB	CentOS 5.11



MobaXterm (1/3)

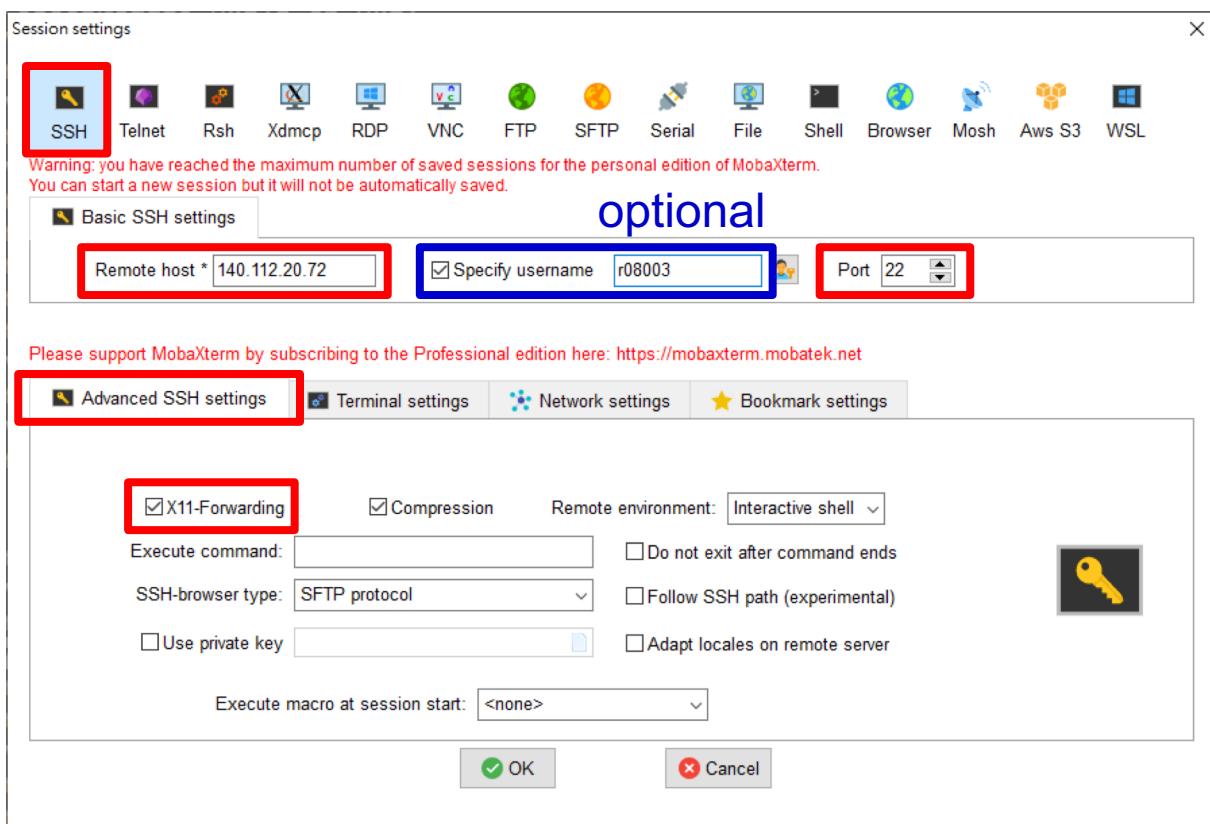
- ◆ Download here: <https://mobaxterm.mobatek.net/>
- ◆ MobaXterm includes SSH, X server, and SFTP
- ◆ Click session





MobaXterm (2/3)

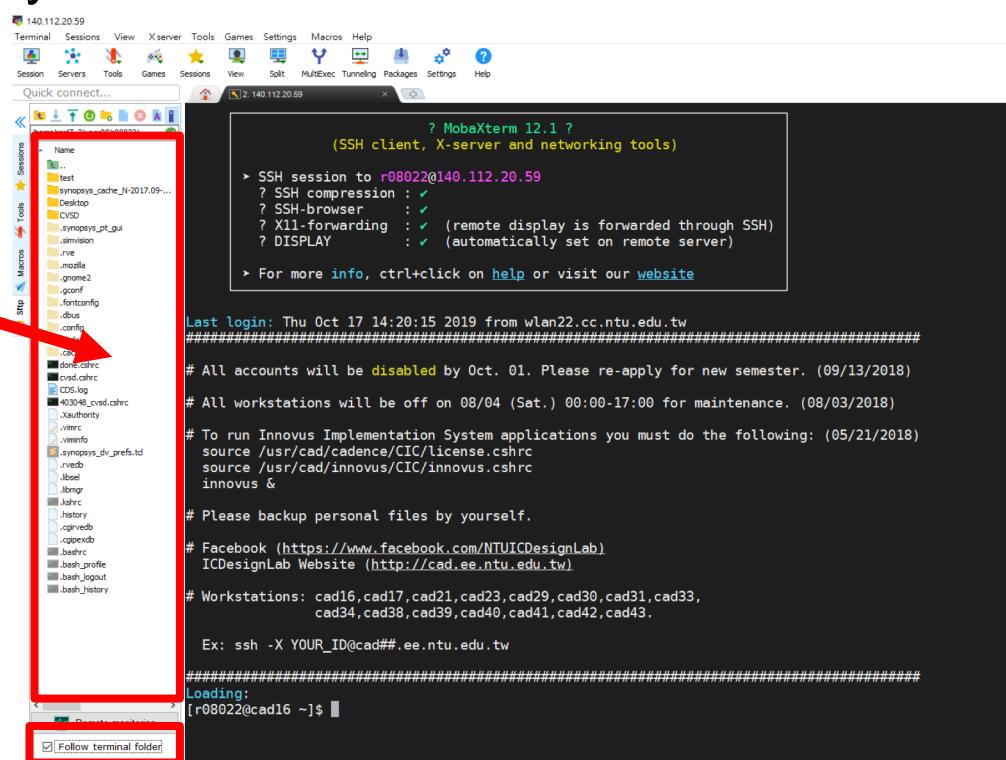
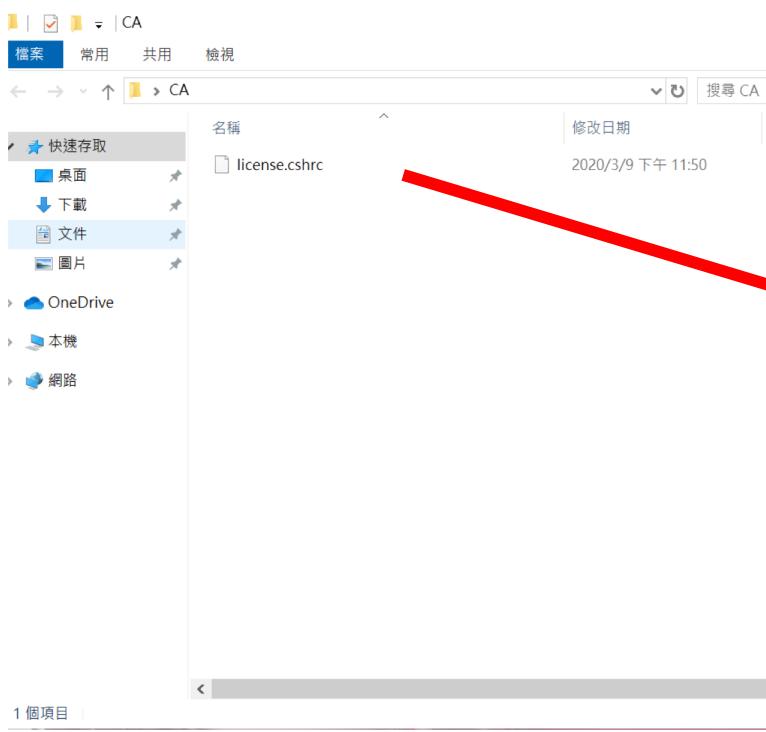
- ◆ Choose SSH
- ◆ Enter IP: 140.112.20.x and Port: 22
- ◆ Enable “X11-Forwarding” in advanced SSH settings





MobaXterm (3/3)

- ◆ After login
- ◆ Enable “Follow terminal folder”
- ◆ Upload license.cshrc to SFTP browser
 - ◆ Drag and drop files directly





For OSX User

- ◆ For X11, you can use XQuartz:
<https://www.xquartz.org/>
- ◆ For FTP, you can use filezilla:
<https://filezilla-project.org/>
 - ◆ Alternative: scp command
- ◆ To login to server, you may use terminal command:
ssh -x <account>@<IP>



Account Unavailable?

- ◆ Email TA: d10943014@ntu.edu.tw
 - ◆ In order to confirm your identity, **only NTU mail** is allowed
 - ◆ Please report your problem about the account **ASAP**

- ◆ Your email must include
 - ◆ Course name
 - ◆ Student ID
 - ◆ Name



Basic Linux Commands (1/6)

For files

- ◆ \$ ls: list your files
- ◆ \$ ls -l: list your files in long format
- ◆ \$ ls -a: list all the files, including the hidden files
- ◆ \$ cp <src-filename> <dest-filename>:
copy the file
- ◆ \$ mv <src-filename> <dest-filename>:
move and rename the file
- ◆ \$ rm <filename>: delete the file
- ◆ \$ more <filename>: show the content of the file



Basic Linux Commands (2/6)

For directory

- ◆ \$ mkdir <dirname>: make a new directory
- ◆ \$ cd <dirname>: change directory
- ◆ \$ cd ..: go up one directory level
- ◆ \$ cd ~: go to home directory
- ◆ \$ rm -r <dirname>: remove directory and all the files inside
- ◆ Some special symbol
 - ◆ ~ means home directory
 - ◆ . means current directory
 - ◆ .. means up one directory level directory
 - ◆ / means root directory (users cannot edit anything)



Basic Linux Commands (3/6)

- ◆ Let's see some examples
- ◆ Change filename from a.v into b.v
 - ◆ \$ mv a.v b.v
- ◆ Create folder called CA_HW
 - ◆ \$ mkdir CA_HW
- ◆ Copy a.v into folder CA_HW
 - ◆ \$ cp ./a.v ./CA_HW/
- ◆ Delete a.v
 - ◆ \$ rm a.v
- ◆ Delete the folder CA_HW
 - ◆ \$ rm -r CA_HW



Basic Linux Commands (4/6)

- ◆ Compress
 - ◆ \$ zip -r <zip_filename> <folder>
- ◆ Decompress
 - ◆ \$ unzip <zip_filename>



Basic Linux Commands (5/6)

◆ Processes

- ◆ \$ Ctrl+C: kill current process (only for the process is into infinite loop or cannot stop via other methods)
- ◆ \$ <command> &: run the command in background
- ◆ \$ jobs: list every process in background
- ◆ \$ Ctrl+Z: **suspend** and put the process in background
- ◆ \$ bg %<number>: let the number of process **run** background
- ◆ \$ fg %<number>: move the number of process to front ground

◆ Logout

- ◆ \$ exit
- ◆ \$ Ctrl+D (EOF)



Basic Linux Commands (6/6)

- ◆ To understand a command, use “man”
 - ◆ “man” command is abbreviation of manual
- ◆ For example: if you want to know how to use “ls” and its parameters
 - ◆ \$ man ls
 - ◆ \$ q to exit

```
LS(1)                               User Commands                               LS(1)
NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILEs (the current directory by default). Sort entries alphabetically if none
    of -cftuvSUX nor --sort.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
        print octal escapes for nongraphic characters

    --block-size=SIZE
        use SIZE-byte blocks. See SIZE format below

    -B, --ignore-backups
        do not list implied entries ending with ~
```



Editors

- ◆ There are so many powerful editors
 - ◆ Sublime text
 - ◆ Notepad++
 - ◆ Visual studio code
 - ◆ Vim
 - ◆ gedit
- ◆ If you already have a favorite editor, just use it
- ◆ If not?
 - ◆ Recommendation: Sublime Text



Introduction to Sublime Text (1/5)

- ◆ Sublime Text is a shareware cross-platform source code editor supports free-software licenses plugins
- ◆ Download here: <https://www.sublimetext.com/>
- ◆ Click “Download” to find the OSX version

The image shows two screenshots of Sublime Text. The top screenshot is the official website for Sublime Text 3 (Build 3211). It features a dark header with the Sublime Text logo, a navigation bar with 'Download' (highlighted with a red box), 'Buy', 'Support', and links to 'News' and 'Forum'. The main content area has a light background with a central box containing the text 'A sophisticated text editor for code, markup and prose'. Below this is a large button labeled 'DOWNLOAD FOR WINDOWS' also highlighted with a red box. To the right of the main content, there's a sidebar with the heading 'Introducing our Git client' and a section titled 'Sublime Merge' showing a file merge interface. The bottom screenshot shows the Sublime Text application window. The title bar says 'Sublime Text'. The left sidebar shows a file tree with folders like 'tensorflow', 'third_party', 'tools', 'util', '.gitignore', 'ACKNOWLEDGMENTS', 'ADOPTERS.md', 'AUTHORS', 'BUILD', and 'CODEOWNERS'. The main editor pane displays a C++ code snippet for base64 encoding:

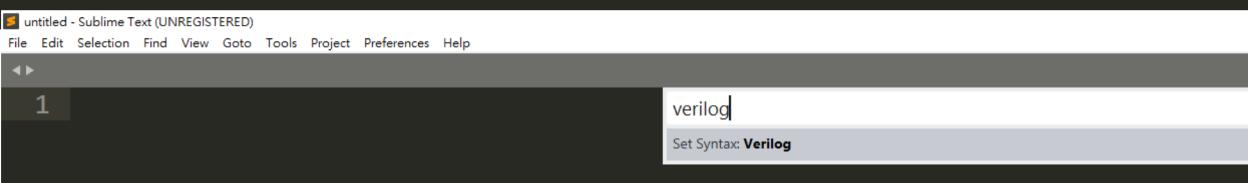
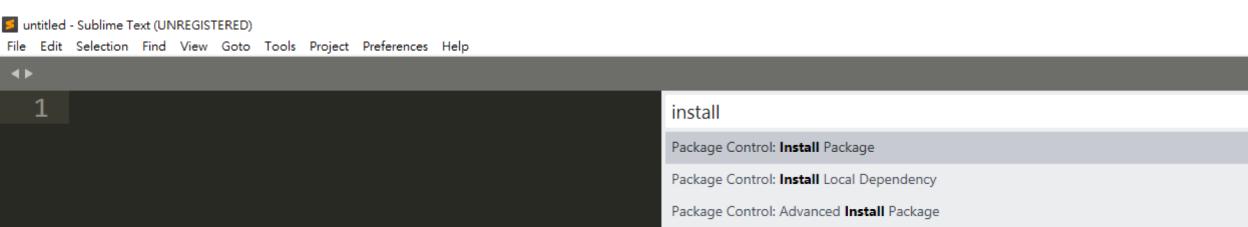
```
34
35 void base64_encode(const uint8_t * data, size_t leng, char * dst,
36     base64_charset variant)
37 {
38     const char * charset = (variant == base64_charset::URL_SAFE)
39         ? URL_SAFE_CHARSET
40         : STANDARD_CHARSET;
41
42     size_t src_idx = 0;
43     size_t dst_idx = 0;
44     for (; (src_idx + 2) < leng; src_idx += 3, dst_idx += 4)
45     {
46         uint8_t s0 = data[src_idx];
47         uint8_t s1 = data[src_idx + 1];
48         uint8_t s2 = data[src_idx + 2];
```

The application window also includes a status bar at the bottom right with the number '16'.



Introduction to Sublime Text (2/5)

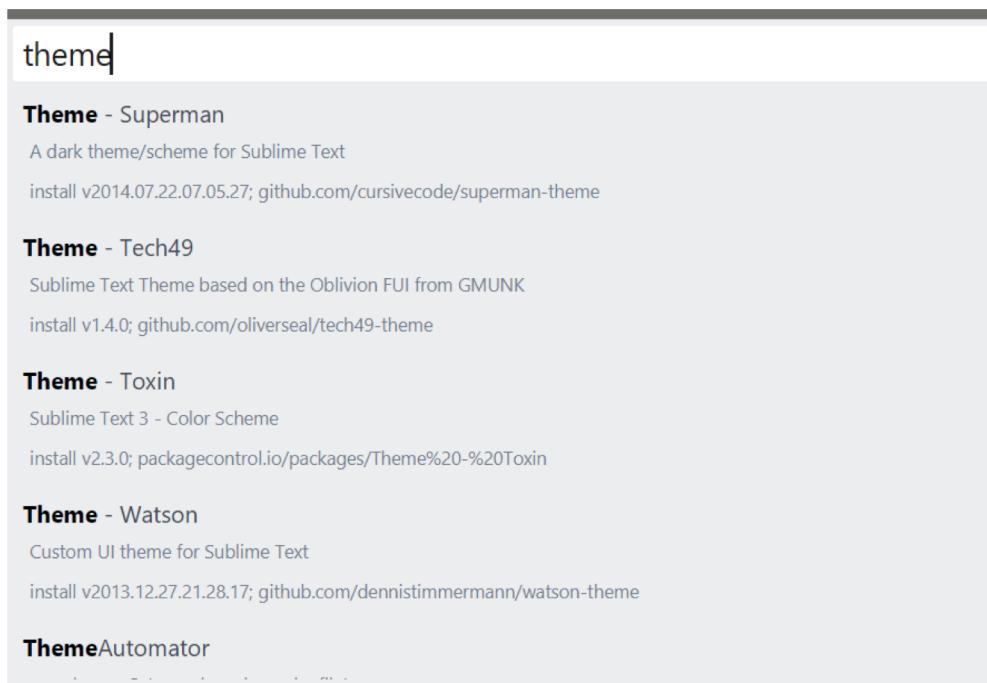
- ◆ Changes font size
 - ◆ Ctrl + scroll
- ◆ Install Verilog syntax
 - ◆ Ctrl+Shift+P opens command line
 - ◆ Type “Package Control: Install Package”
 - ◆ Type “Verilog” to install Verilog syntax
 - ◆ Type “Verilog” to set syntax as Verilog





Introduction to Sublime Text (3/5)

- ◆ Want a different theme?
 - ◆ Ctrl+Shift+P opens command line
 - ◆ Type “Package Control: Install Package”
 - ◆ Type “theme”
 - ◆ Try to pick one





Introduction to Sublime Text (4/5)

- ◆ Why sublime text?
 - ◆ Free
 - ◆ Cool
 - ◆ Convenient
- ◆ Example: need to change “reg” into “wire” and add semicolon for each line?
 - ◆ Select one “reg”
 - ◆ Ctrl+D to select all “reg”
 - ◆ Just modify them simultaneously
- ◆ Other tips?
 - ◆ Discover it or google it by yourself

```
reg a
1 reg a
2 reg b
3 reg c
```

```
reg a
1 reg a
2 reg b
3 reg c
```

```
reg a
1 reg a
2 reg b
3 reg c
```

```
wire a;
1 wire a;
2 wire b;
3 wire c;
```



Introduction to Sublime Text (5/5)

- ◆ Need to create some codes including a increasing number?
 - ◆ Install “insertNums” via package control
 - ◆ <https://packagecontrol.io/packages/Insert%20Nums>
- ◆ Other tips?
 - ◆ Discover it or google it by yourself

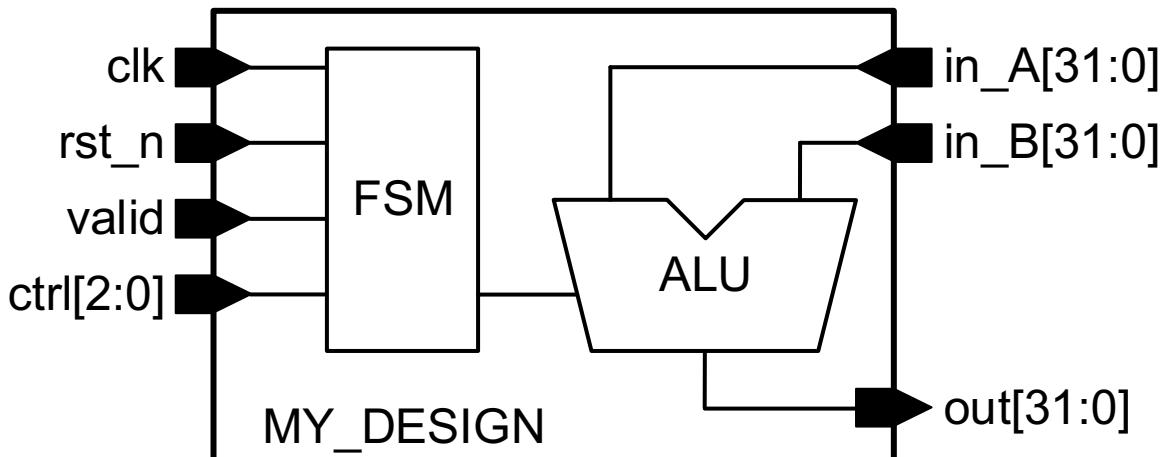
The screenshot shows three panels of Sublime Text. The left panel contains a sequence of 15 identical lines of code: "a01*b01,". The middle panel shows the result of using the "insertNums" package, where each line is preceded by a line number from 36 to 54. A red arrow points from the text "Ctrl+alt+n" to the first line of the generated code. The right panel shows the final output, where each line is followed by a plus sign (+), indicating the code is ready for concatenation.

Line Number	Generated Code
36	a01*b01, +
37	a02*b01, +
38	a03*b01, +
39	a04*b01, +
40	a05*b01, +
41	a06*b01, +
42	a07*b01, +
43	a08*b01, +
44	a09*b01, +
45	a10*b01, +
46	a11*b01, +
47	a12*b01, +
48	a13*b01, +
49	a14*b01, +
50	
51	
52	
53	
54	



Lab (1/6)

- ◆ FSM and ALU
- ◆ Specification



Name	I/O	Width	Description
clk	I	1	Positive edge-triggered clock
rst_n	I	1	Asynchronous negative edge reset
valid	I	1	Update state if valid = 1
ctrl[2:0]	I	3	Input to modify state
in_A[31:0]	I	32	Input A
in_B[31:0]	I	32	Input B
out[31:0]	O	32	Output



Lab (2/6)

- ◆ State description (all signed computation)

State	Function	Description
3'd0	ADD	$out = in_A + in_B$ (neglect overflow)
3'd1	SUB	$out = in_A - in_B$ (neglect overflow)
3'd2	SLL	$out = \text{Logical left shift } in_A \text{ by } in_B \text{ bits } (0 \leq in_B \leq 31)$
3'd3	XOR	$out = in_A \text{ XOR } in_B$ (bitwise)
3'd4	SRL	$out = \text{Logical right shift } in_A \text{ by } in_B \text{ bits } (0 \leq in_B \leq 31)$
3'd5	SRA	$out = \text{Arithmetic right shift } in_A \text{ by } in_B \text{ bits } (0 \leq in_B \leq 31)$
3'd6	OR	$out = in_A \text{ OR } in_B$ (bitwise)
3'd7	AND	$out = in_A \text{ AND } in_B$ (bitwise)



Lab (3/6)

◆ Top module

- ◆ Instantiate relating sub-modules
- ◆ Connect them

```
module MY_DESIGN(
    clk,
    rst_n,
    valid,
    ctrl,
    in_A,
    in_B,
    out
);
    // Port list
    input      clk, rst_n, valid;
    input [ 2:0] ctrl;
    input [31:0] in_A, in_B;
    output [31:0] out;
    // Wire and reg
    wire   [ 2:0] FSM_ctrl, ALU_ctrl;
```

```
// Instantiate sub-modules
FSM FSM_INST(
    .clk(clk),
    .rst_n(rst_n),
    .valid(valid),
    .FSM_ctrl(FSM_ctrl),
    .ALU_ctrl(ALU_ctrl)
);
ALU ALU_INST(
    .ctrl(ALU_ctrl),
    .in_A(in_A),
    .in_B(in_B),
    .out(out)
);
// Continuous assignment
assign FSM_ctrl = ctrl;
endmodule
```



Lab (4/6)

- ◆ There are some mistakes in my_design.v
 - ◆ Only in module ALU
- ◆ Revise it!

```
// Combinational block
always @(*) begin
    case(ctrl)
        // Todo: Complete ALU
        3'd0: out = in_A + in_B;
        3'd1: out = in_A + in_B;
        3'd2: out = in_A + in_B;
        3'd3: out = in_A + in_B;
        3'd4: out = in_A + in_B;
        3'd5: out = in_A + in_B; Wrong!
        3'd6: out = in_A + in_B;
        3'd7: out = in_A + in_B;
        // Full-case, neglect default case
    endcase
end
```



Lab (5/6)

- ◆ Simulation
- ◆ Create directory: CA_Lab
 - ◆ \$ mkdir CA_Lab
 - ◆ \$ cd CA_Lab/
- ◆ Upload related files to CA_Lab/
 - ◆ license.cshrc
 - ◆ run.f
 - ◆ testbench.v
 - ◆ my_design.v

The screenshot shows a MobaXterm window with two tabs. The left tab, titled '140.112.20.72 (r08003)', displays a file browser showing files: .., testbench.v, run.f, my_design.v, and license.cshrc. The right tab, titled '2. 140.112.20.72 (r08003)', shows the simulation results:

```
Test function: SUB
Test function: XOR
*****
* Fisnish Simulation *
*****
* Simulation Result: *
* You pass!           *
*****
Simulation complete via $finish(1) at time 985 NS + 0
./testbench.v:71      $finish;
ncsim> exit
[r08003@cad29 lab]$ cd ~
[r08003@cad29 ~]$ mkdir CA_Lab
[r08003@cad29 ~]$ cd CA_Lab/
[r08003@cad29 ~/CA_Lab]$
```

At the bottom of the window, there is a message: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".



Lab (6/6)

◆ Source

- ◆ \$ source license.cshrc

◆ Simulation

- ◆ \$ ncverilog -f run.f

◆ Debug

- ◆ \$ nWave &

```
*****
* Begin Simulation... *
*****
Test function: ADD
Test function: AND
Test function: OR
Test function: SLL
Test function: SRA
Test function: SRL
Test function: SUB
Test function: XOR
*****
* Fisnish Simulation *
*****
* Simulation Result: *
* You pass! *
*****
```