

EE 2310 Worksheet #6A – Load/Store and Shift/Rotate Instructions

MIPS Registers

| |
|----------|
| .data |
| n: .word |
| p: .word |
| q: .word |
| r: .word |
| s: .word |
| t: .word |
| u: .word |
| v: .word |
| w: .word |
| x: .word |
| y: .word |
| z: .word |
| str: |
| .asciiz |
| "hello |
| world\n" |

| | | | |
|------------------------|-------------------------|-------------------------|--------------------------|
| R0 (r0): 0x00000000 | R8 (t0): 0x00000000 | R16 (s0): 0xff00ff00 | R24 (t8): 0x00000000 |
| R1 (at): 0x10010000 | R9 (t1): 0x0000ffff | R17 (s1): 0x00000000 | R25 (t9): 0x00000000 |
| R2 (v0): 0x00000000 | R10 (t2): 0x00000000 | R18 (s2): 0x00000023 | R26 (k0): 0x00000000 |
| R3 (v1): 0x0000000c | R11 (t3): 0x10010020 | R19 (s3): 0x10010000 | R27 (k1): 0x00000000 |
| R4 (a0): 0x00000023 | R12 (t4): 0x10010030 | R20 (s4): 0x00400020 | R28 (gp): 0x10008000 |
| R5 (a1): 0x10010010 | R13 (t5): 0x10010020 | R21 (s5): 0x00000000 | R29 (sp): 0x7ffffeff0 |
| R6 (a2): 0x0000000a | R14 (t6): 0x80000010 | R22 (s6): 0x800c1001 | R30 (s8): 0x00000000 |
| R7 (a3): 0x00000050 | R15 (t7): 0xffff0000 | R23 (s7): 0x00000010 | R31 (ra): 0x00400060 |

Data

[0x10000000]...[0x1000fffc] 0x00000000

| | | | | |
|--------------|------------|------------|------------|------------|
| [0x10010000] | 0x5350494d | 0x20697320 | 0x61207573 | 0x6566756c |
| [0x10010010] | 0x2073696d | 0x756c6174 | 0x6f722066 | 0x6f72206c |
| [0x10010020] | 0x6561726e | 0x5350494d | 0x4d495053 | 0x20617373 |
| [0x10010030] | 0x68656c6c | 0x6f20776f | 0x726c640a | 0x00000000 |
| [0x10010040] | 0x726f6772 | 0x00000000 | 0x00000000 | 0x6e642063 |
| [0x10010050] | 0xf7f7f7f7 | 0xf7f7f7f7 | 0xf7f7f7f7 | 0xf7f7f7f7 |

[0x10010060]...[0x10020000] 0x00000000

Use the data declaration, SPIM memory dump, and MIPS register readouts above as directed in the problems starting on the following page. Note: If a register or memory location is changed in any problem, the change **DOES NOT CARRY OVER** to another problem!

EE 2310 Worksheet #6A – Load/Store and Shift/Rotate Instructions

1. After: lw \$t0, w, what are the contents of \$t0? _____

2. After: sw \$s0, w, (a) what are the contents of memory location w?
(b) at what real memory address is w? _____

3. After: lw \$t0, 12(\$a1), what are the contents of \$t0? _____

4. After: sw \$t7, 16(\$t4), what memory location contains the
contents of \$t7? _____

5. After: srl \$t0, \$ra, 12, what are the contents of \$t0? _____

6. After: sra \$t0, \$s6, 16, what are the contents of \$t0? _____

7. After: rol \$t0, \$t7, 20, what are the contents of \$t0? _____

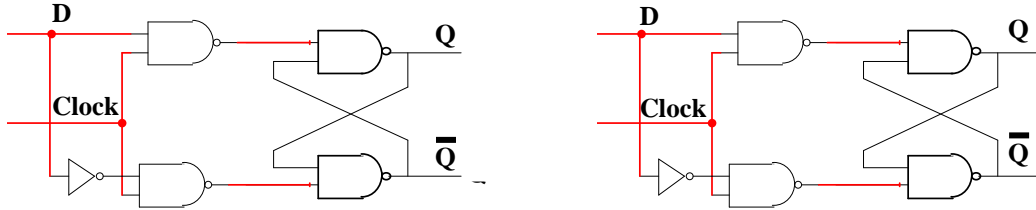
8. After: lw \$t6, 16(\$t4); ror \$t6, \$t6, 16, what are the contents of \$t6? _____

9. After completion of: jr \$s4, what are the contents of the PC? _____

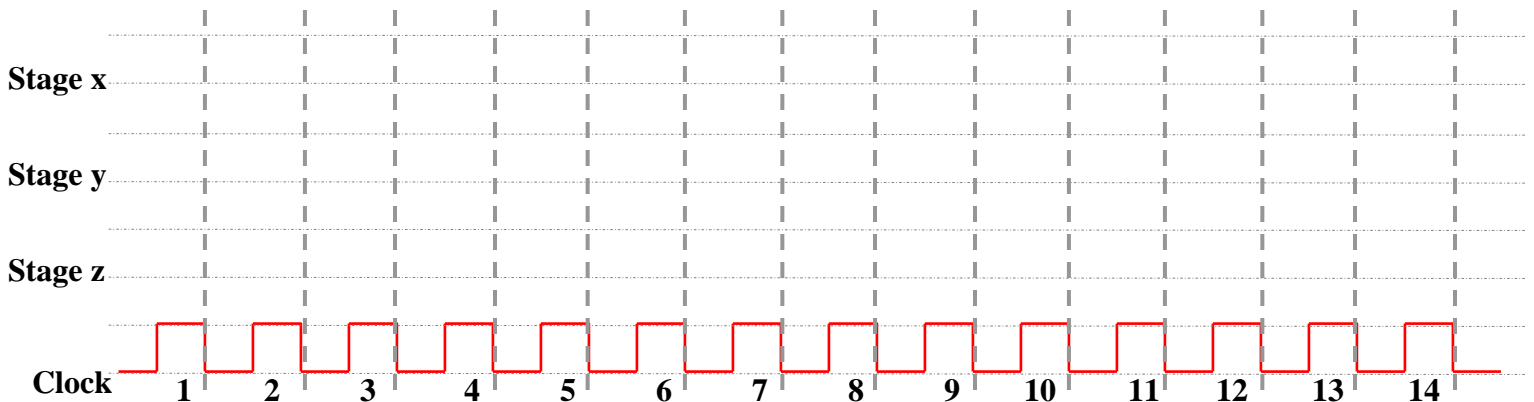
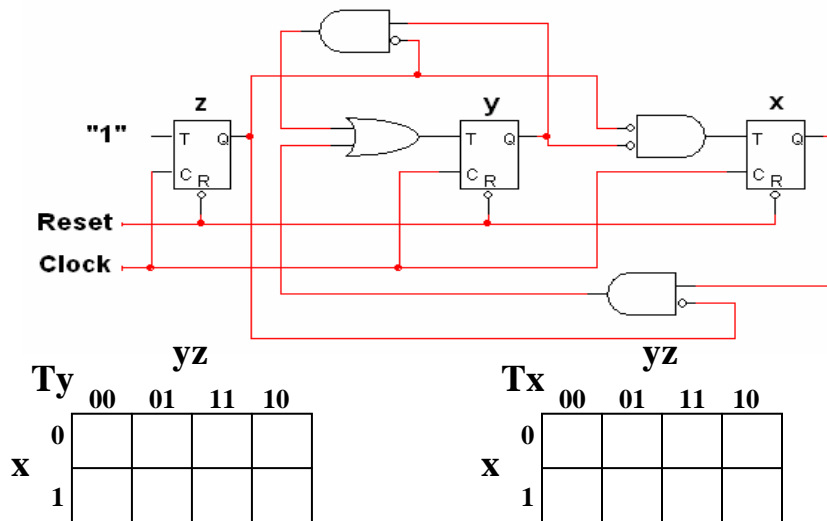
10. After: ror \$t0, \$sp, 14; sra \$t0, \$t0, 6 what are the contents of \$t0? _____

EE 2310 Worksheet #6A – Sequential Logic

- Using the simple D FFs shown below, make a simplified T master-slave FF with T and clock inputs. Remember: When T is high and the clock goes through one cycle, the FF toggles (that is, goes to the opposite state). With T low, the FF will not change states regardless of the clock. You will need a few extra gates. Note: This is a thought problem. That is, there is no “plug-in” solution. You must REASON it out.



- The counter below counts in an unusual cycle, so simply finding “m-1” won’t help you in this case. Using the K-maps below, find the Boolean expressions for T_y and T_x (clearly, $T_z = 1$). With those, you can determine the T’s at count 0, then let the clock tick. You can then determine the T’s at count 1 and repeat, etc. using this method, you can find the count cycle. You will eventually discover some “don’t cares.” Make a timing diagram to help you with the count.



3. Develop a timing diagram for the MUX on the chart shown below. Plot the timing of FF's x, y, and z, and also the MUX Out signal.

