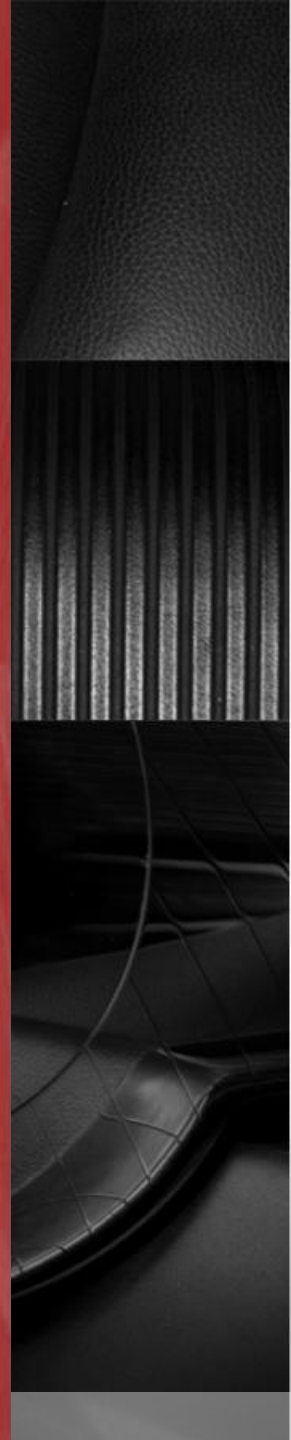
EECT 112 & Projects from MultiSim

Jeff Noggle & Stephen Smith

EECT 112

Summer 2013



Contents

- Conversion Table
- Multisim Circuits
- Boolean Diagrams using Multisim
- LABS
- Final Project

- Additional Data
  - Logic Gates, Boolean Diagrams, Chip Diagrams,

BCD, HEX, Base 10, Octal, Gray Conversions

Base 10			
100	10	1	Dec
0	0	0	0
0	0	1	1
0	0	2	2
0	0	3	3
0	0	4	4
0	0	5	5
0	0	6	6
0	0	7	7
0	0	8	8
0	0	9	9
0	1	0	10
0	1	1	11
0	1	2	12
0	1	3	13
0	1	4	14
0	1	5	15
0	1	6	16
0	1	7	17
0	1	8	18
0	1	9	19
0	2	0	20

Binary					Dec2Bin
8	4	2	1	Dec	
0	0	0	0	0	0000
0	0	0	1	1	0001
0	0	1	0	2	0010
0	0	1	1	3	0011
0	1	0	0	4	0100
0	1	0	1	5	0101
0	1	1	0	6	0110
0	1	1	1	7	0111
1	0	0	0	8	1000
1	0	0	1	9	1001
1	0	1	0	10	1010
1	0	1	1	11	1011
1	1	0	0	12	1100
1	1	0	1	13	1101
1	1	1	0	14	1110
1	1	1	1	15	1111

Hexadecimal				Excel
256	16	1	Dec	Dec2Hex
0	0	0	0	0000
0	0	1	1	0001
0	0	2	2	0002
0	0	3	3	0003
0	0	4	4	0004
0	0	5	5	0005
0	0	6	6	0006
0	0	7	7	0007
0	0	8	8	0008
0	0	9	9	0009
0	0	A	10	000A
0	0	B	11	000B
0	0	C	12	000C
0	0	D	13	000D
0	0	E	14	000E
0	0	F	15	000F

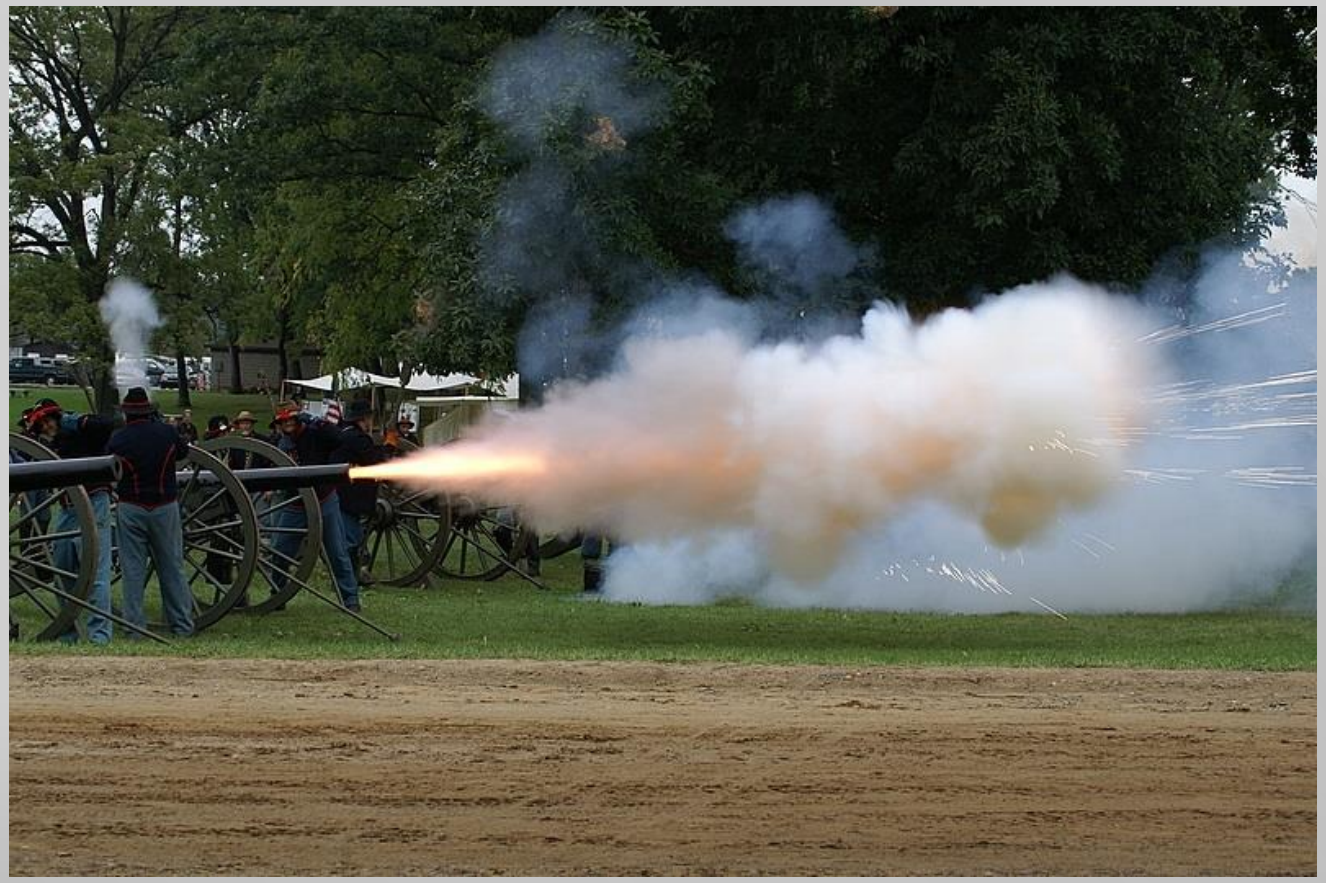
Octal				Excel
64	8	1	Dec	Dec2Oct
0	0	0	0	0000
0	0	1	1	0001
0	0	2	2	0002
0	0	3	3	0003
0	0	4	4	0004
0	0	5	5	0005
0	0	6	6	0006
0	0	7	7	0007
0	1	0	8	0000
0	1	1	9	0001
0	1	2	10	0002
0	1	3	11	0003
0	1	4	12	0004
0	1	5	13	0005
0	1	6	14	0006
0	1	7	15	0007

Excel Conversion		
Oct2Bin	Oct2Dec	Oct2Hex
0000	0	0000
0001	1	0001
0010	2	0002
0011	3	0003
0100	4	0004
0101	5	0005
0110	6	0006
0111	7	0007
0000	0	0000
0001	1	0001
0010	2	0002
0011	3	0003
0100	4	0004
0101	5	0005
0110	6	0006
0111	7	0007

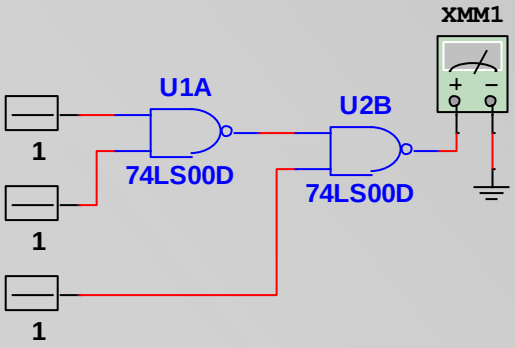
Gray			Dec
0	0	0	0
0	0	1	1
0	1	1	3
0	1	0	2
1	1	0	6
1	1	1	7
1	0	1	5
1	0	0	4

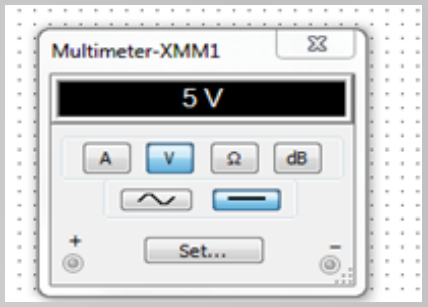
BCD		Dec
	0000	0
	0001	1
	0010	2
	0011	3
	0100	4
	0101	5
	0110	6
	0111	7
	1000	8
	1001	9
0001	0000	10
0001	0001	11
0001	0010	12
0001	0011	13
0001	0100	14
0001	0101	15
0001	0110	16

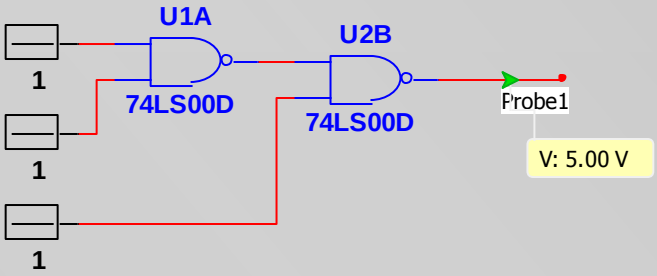
Using Multisim. . . It’s a BLAST!!!!!

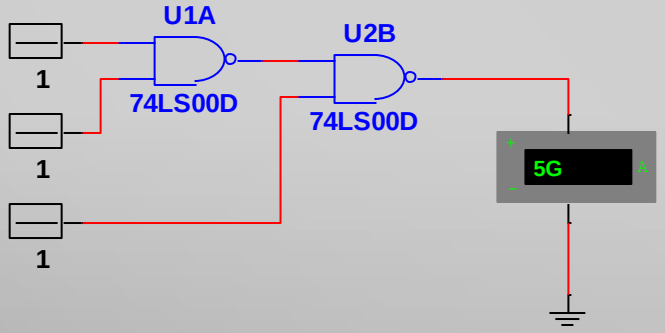


Dual NAND Gates and Methods to set them up:
Slides 2 thru 5

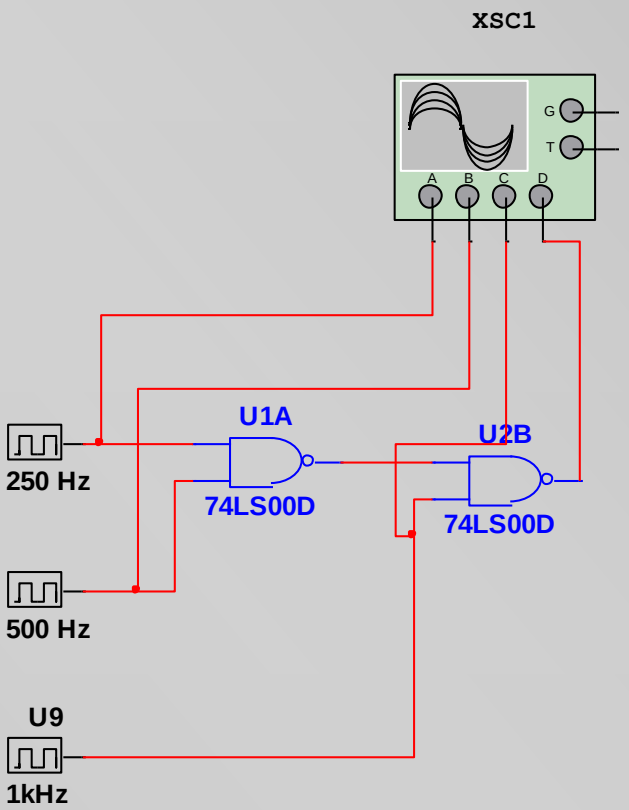


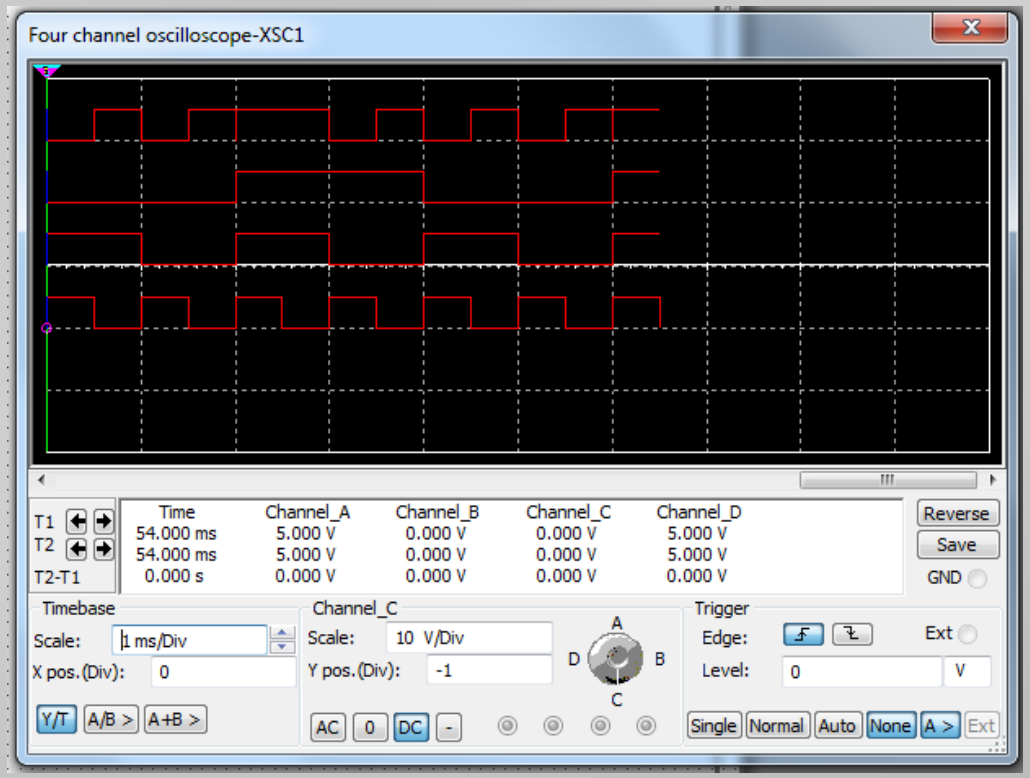




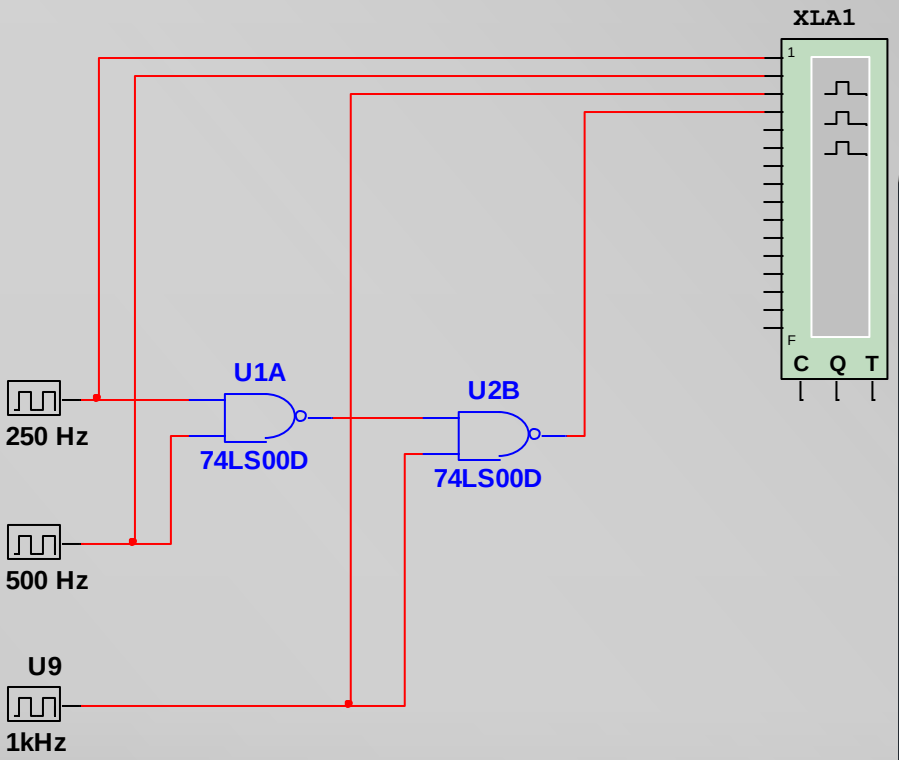


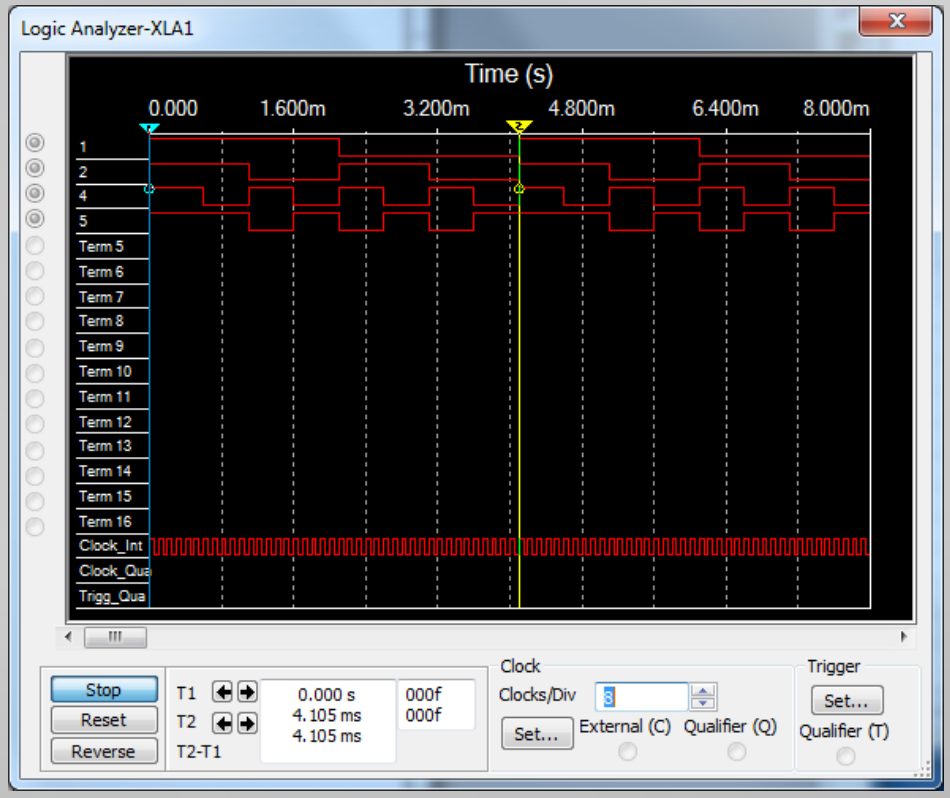
NAND GATES – Setup one

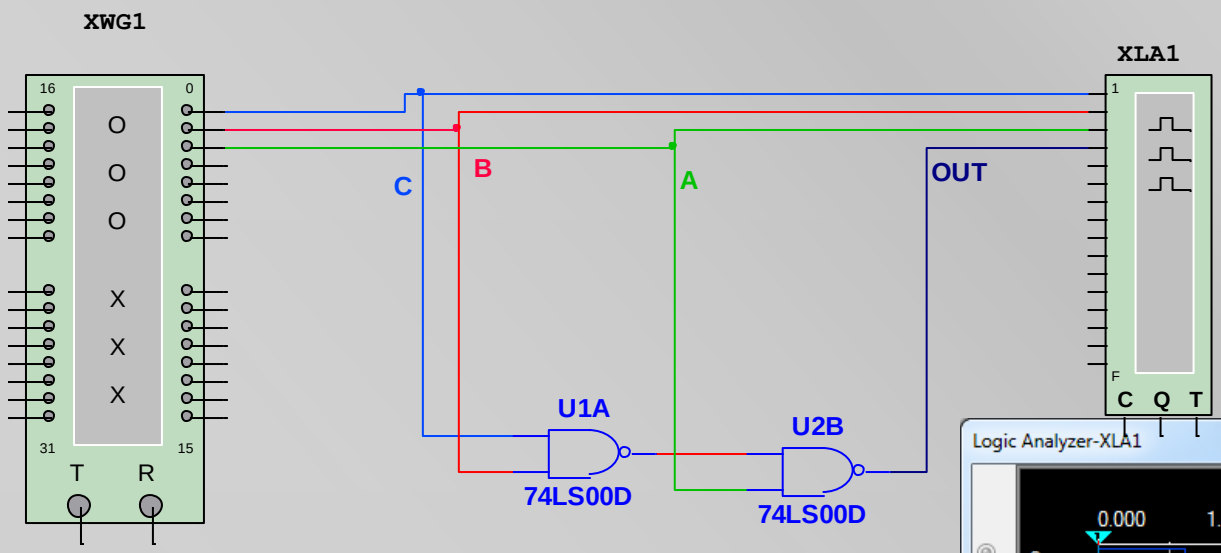


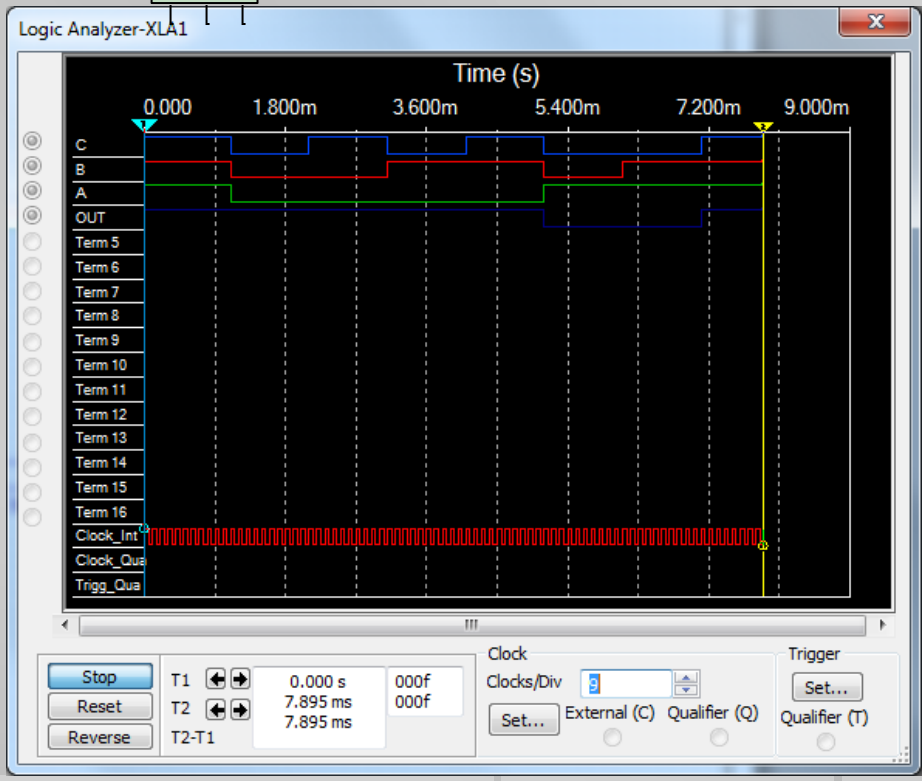


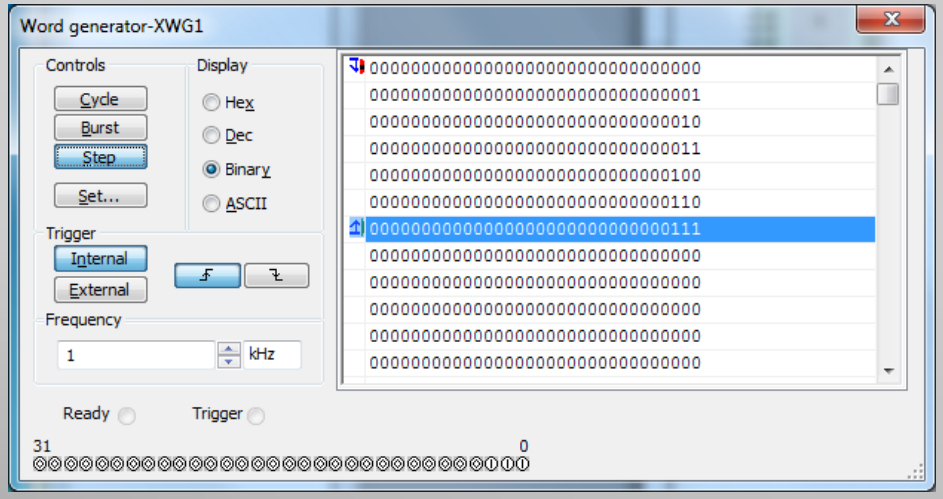
NAND GATES – Setup Two

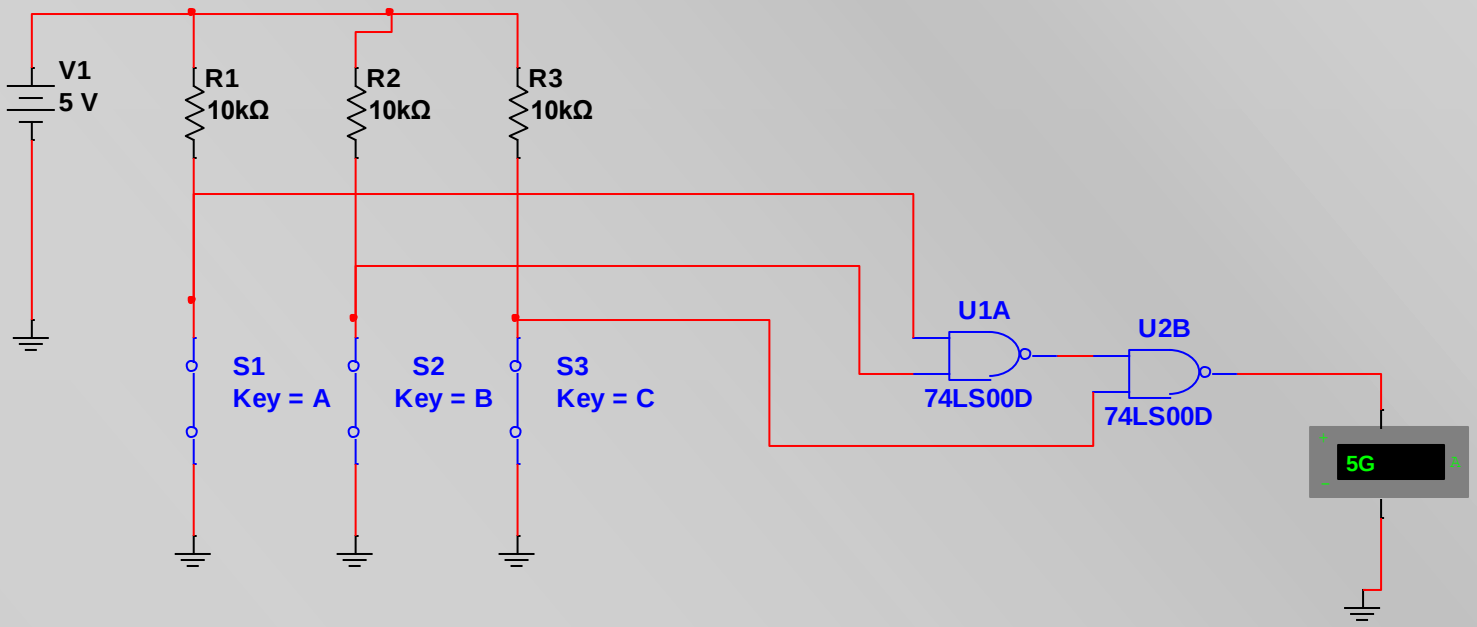




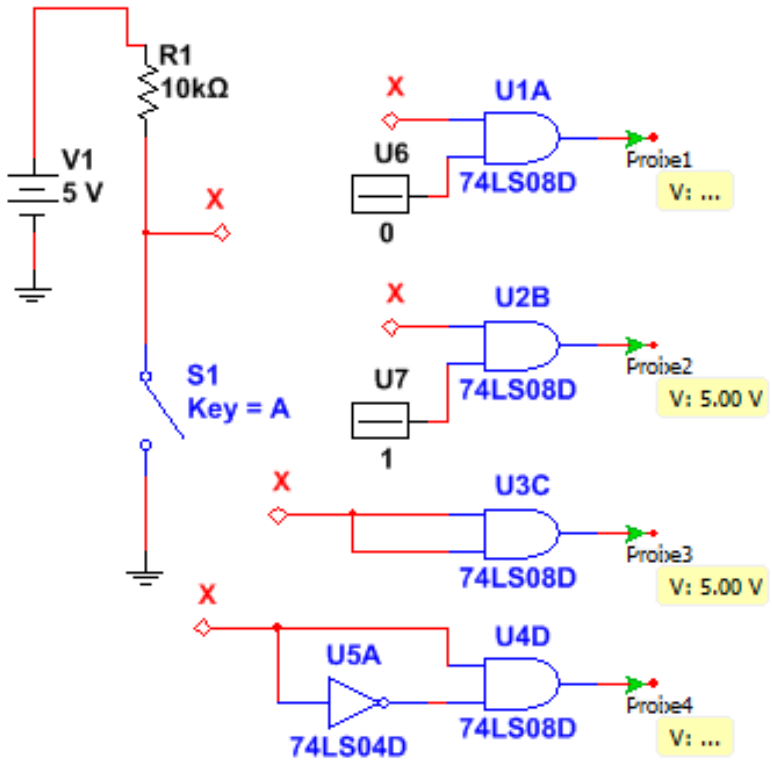

XWG1
XLA1
16
0
31
15
T
R
O
O
O
X
X
X
1
F
C
Q
T
C
B
A
OUT
U1A
U2B
74LS00D
74LS00D


Logic Analyzer-XLA1
Time (s)
0.000
1.800m
3.600m
5.400m
7.200m
9.000m
C
B
A
OUT
Term 5
Term 6
Term 7
Term 8
Term 9
Term 10
Term 11
Term 12
Term 13
Term 14
Term 15
Term 16
Clock_Int
Clock_Qua
Trigg_Qua
Stop
Reset
Reverse
T1
T2
T2-T1
0.000 s
7.895 ms
7.895 ms
000f
000f
Clock
Clocks/Div
Set...
External (C)
Qualifier (Q)
Trigger
Set...
Qualifier (T)


Word generator-XWG1
Controls
Cycle
Burst
Step
Set...
Display
Hex
Dec
Binary
ASCII
Trigger
Internal
External
Frequency
1
kHz
Ready
Trigger
31
0


V1
5 V
R1
10kΩ
R2
10kΩ
R3
10kΩ
S1
Key = A
S2
Key = B
S3
Key = C
U1A
74LS00D
U2B
74LS00D
5G

Boolean Theorems for AND Gates



Boolean Theorems for AND Gates

T1: X x 0 = 0

InX	In0	Out
0	0	0
1	0	0

T2: X x 1 = X

InX	In1	Out
0	1	0
1	1	1

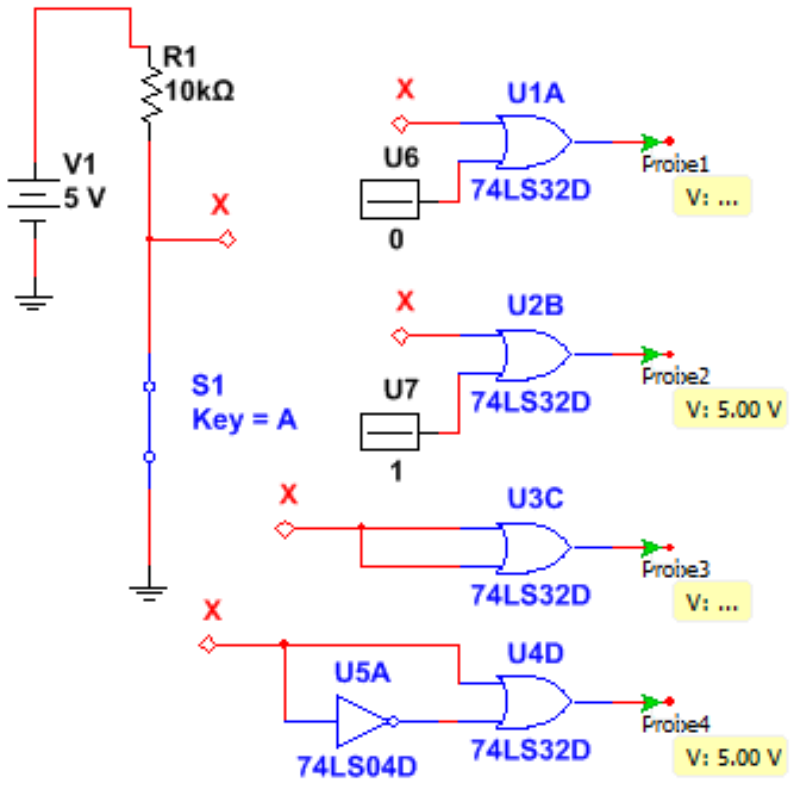
T3: X x X = X

InX	InX	Out
1	1	1
0	0	0

T4: X x X(inverted) = 0

InX	InX(Bar)	Out
1	1(Bar)	0
0	0(Bar)	0

Boolean Theorems for OR Gates



Boolean Theorems for OR Gates

T1: X x 0 = X

InX	In0	Out
0	0	0
1	0	1

T2: X x 1 = 1

InX	In1	Out
0	1	1
1	1	1

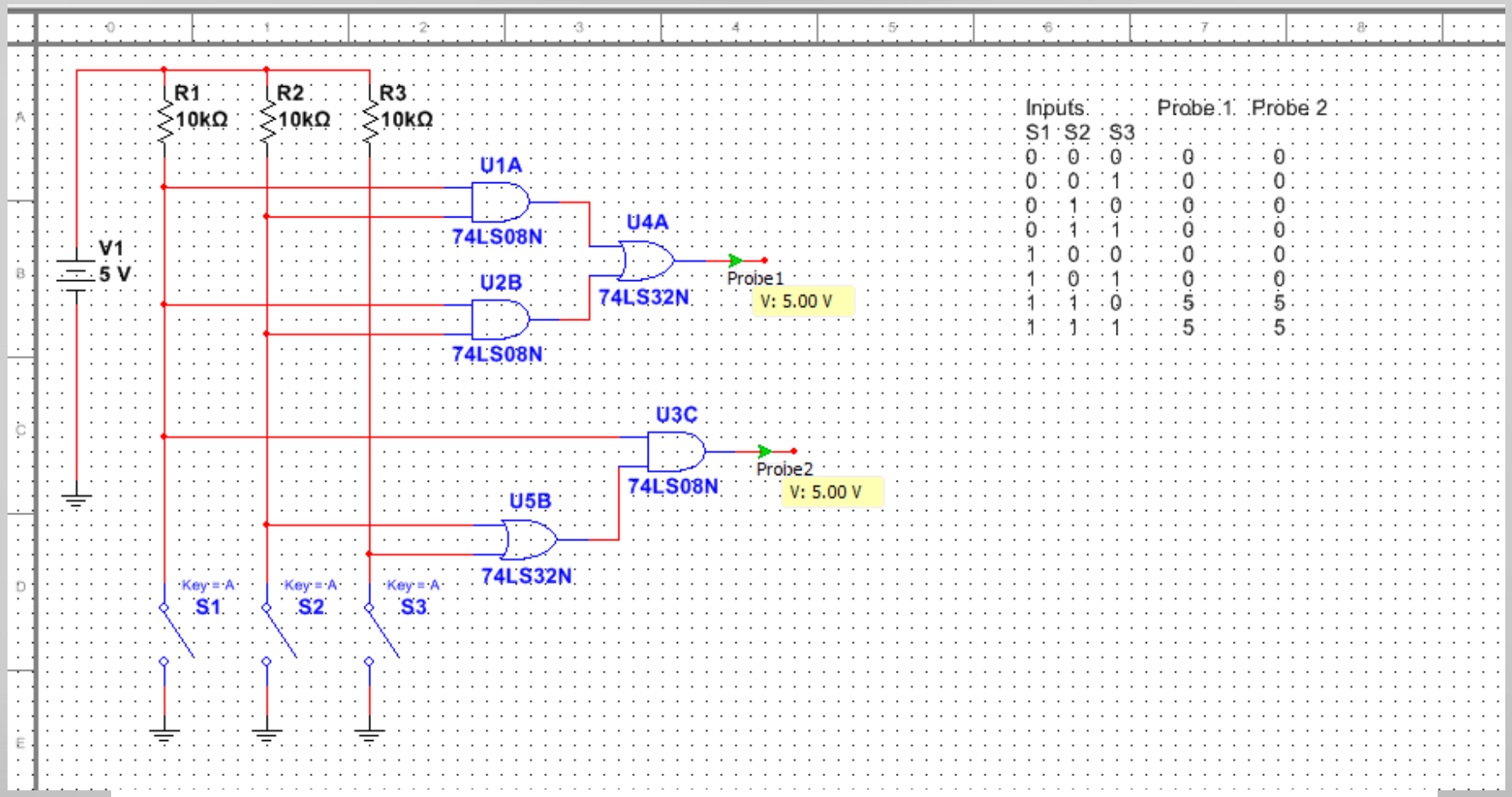
T3: X x X = X

InX	InX	Out
1	1	1
0	0	0

T4: X x X(inverted) = 1

InX	InX(Bar)	Out
1	1(Bar)	1
0	0(Bar)	1

Multivariable Distributive Laws 13a

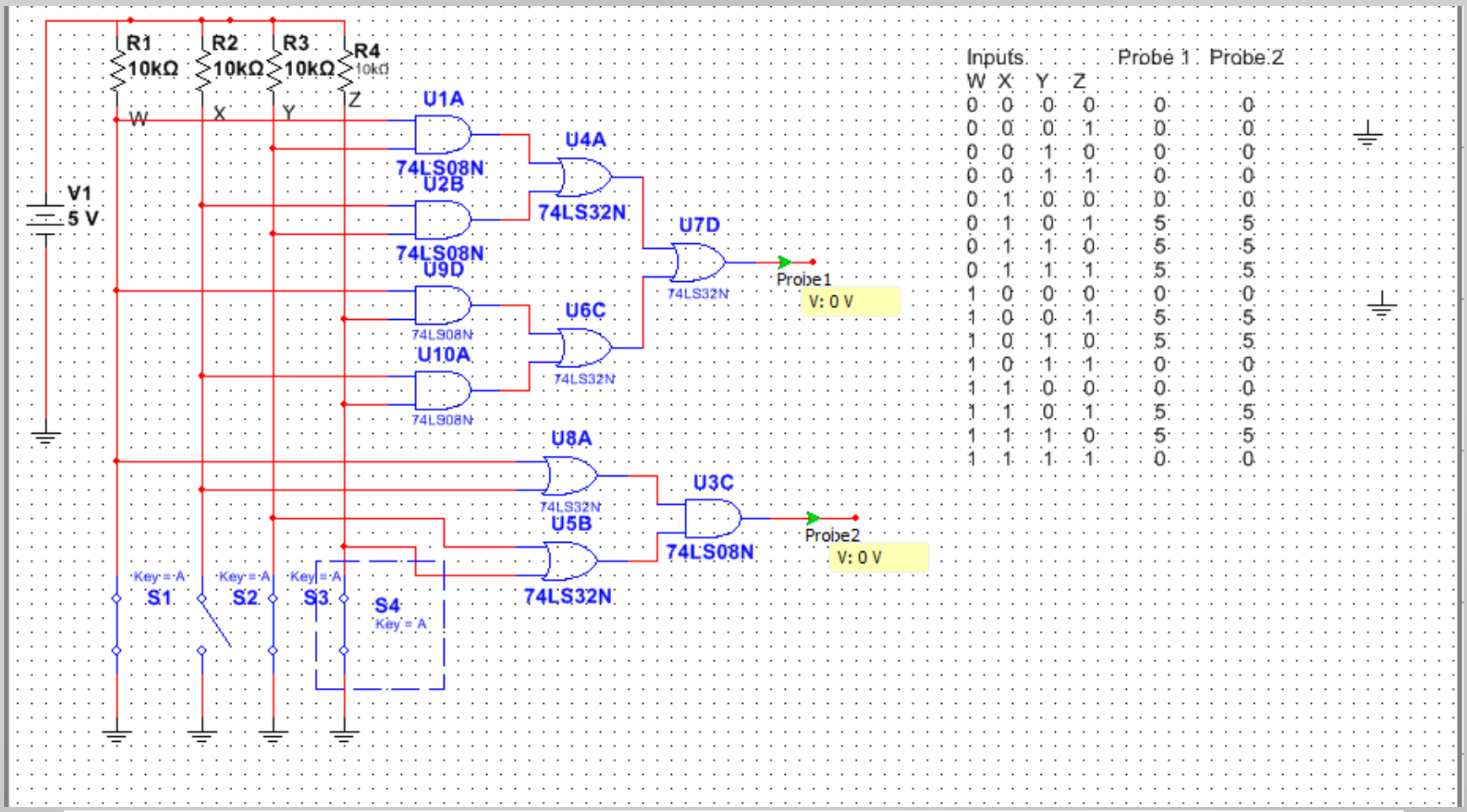


Inputs			Probe 1	Probe 2
S1	S2	S3		
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	5	5
1	1	1	5	5

$$(13a) \quad x(y+z) = xy + xz$$

$$(13b) \quad (w+x)(y+z) = wy + xy + wz + xz$$

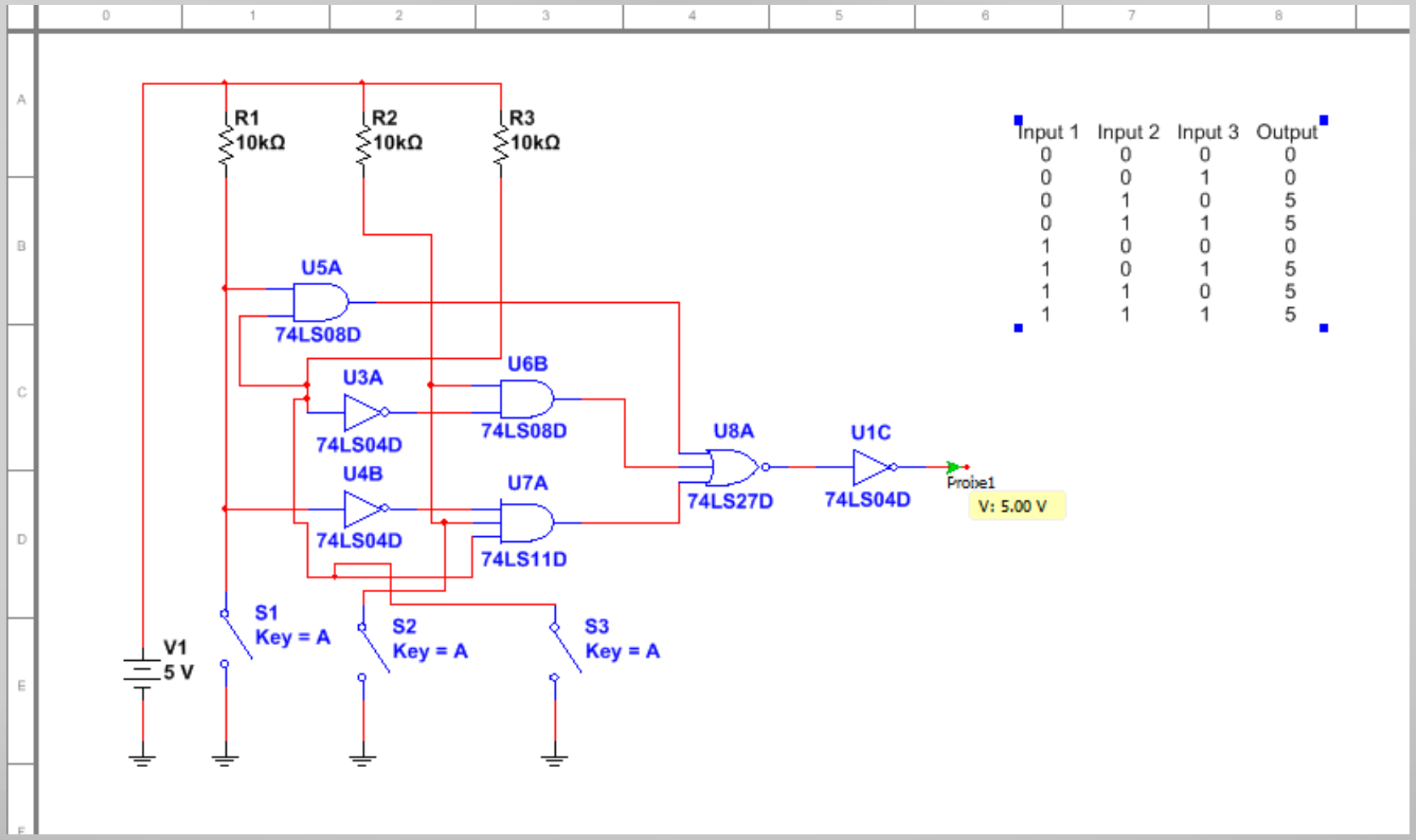
Multivariable Distributive Law 13b



$$(13a) \quad x(y+z) = xy + xz$$

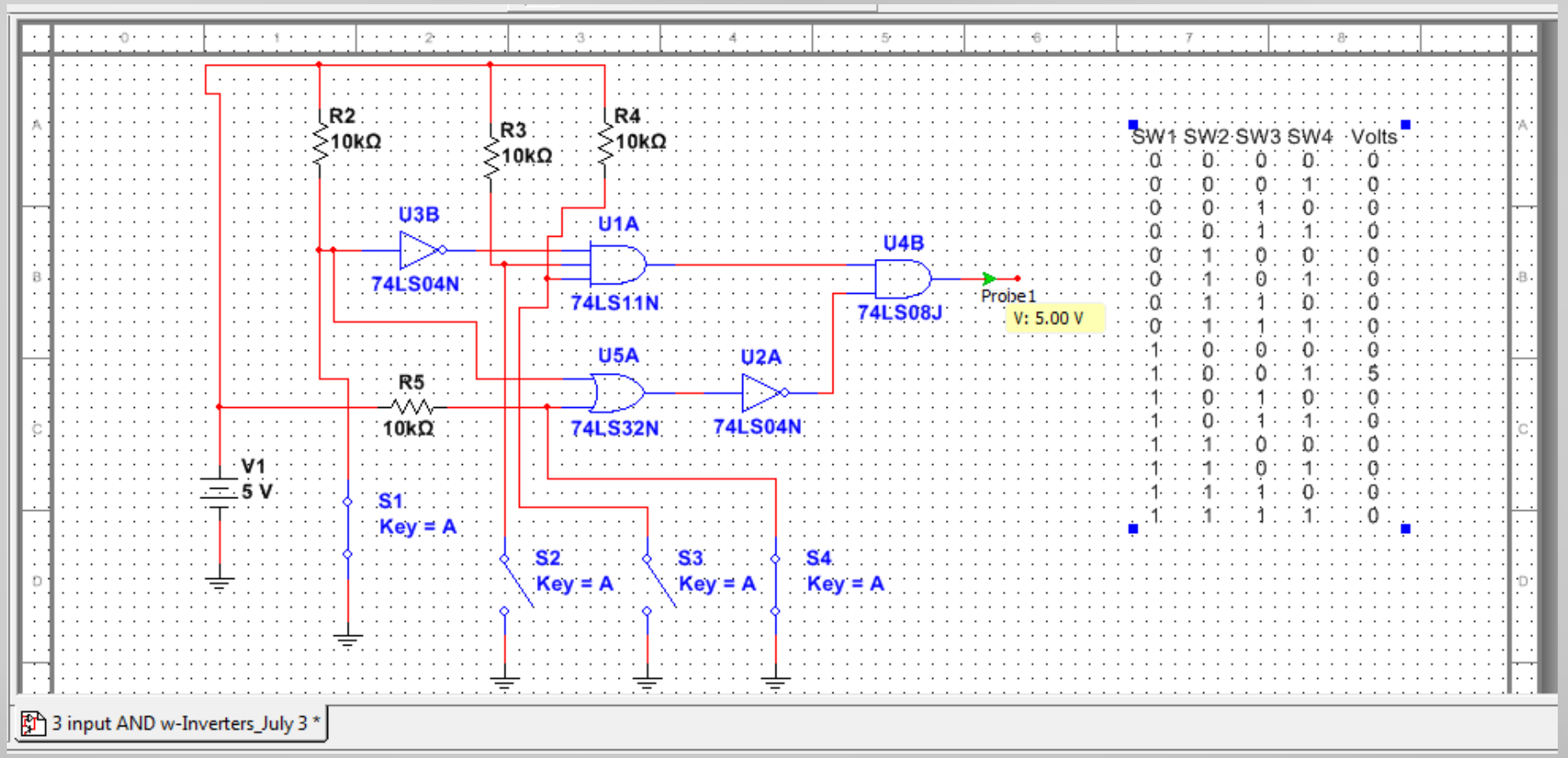
$$(13b) \quad (w+x)(y+z) = wy + xy + wz + xz$$

3 input OR Gate

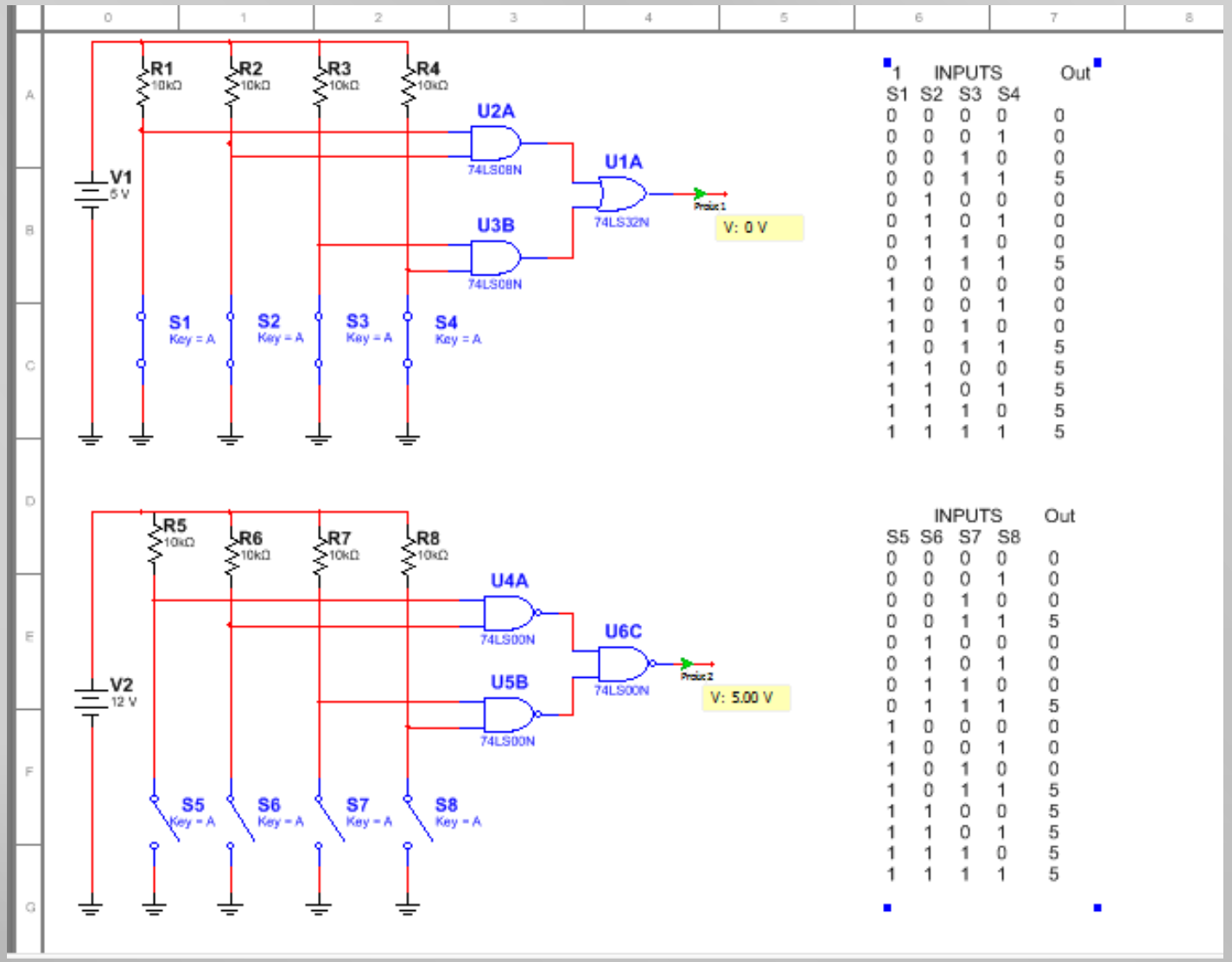


Lecture 3, Slide 20, Simulated on MultiSim

3 Input AND with Inverter Circuit



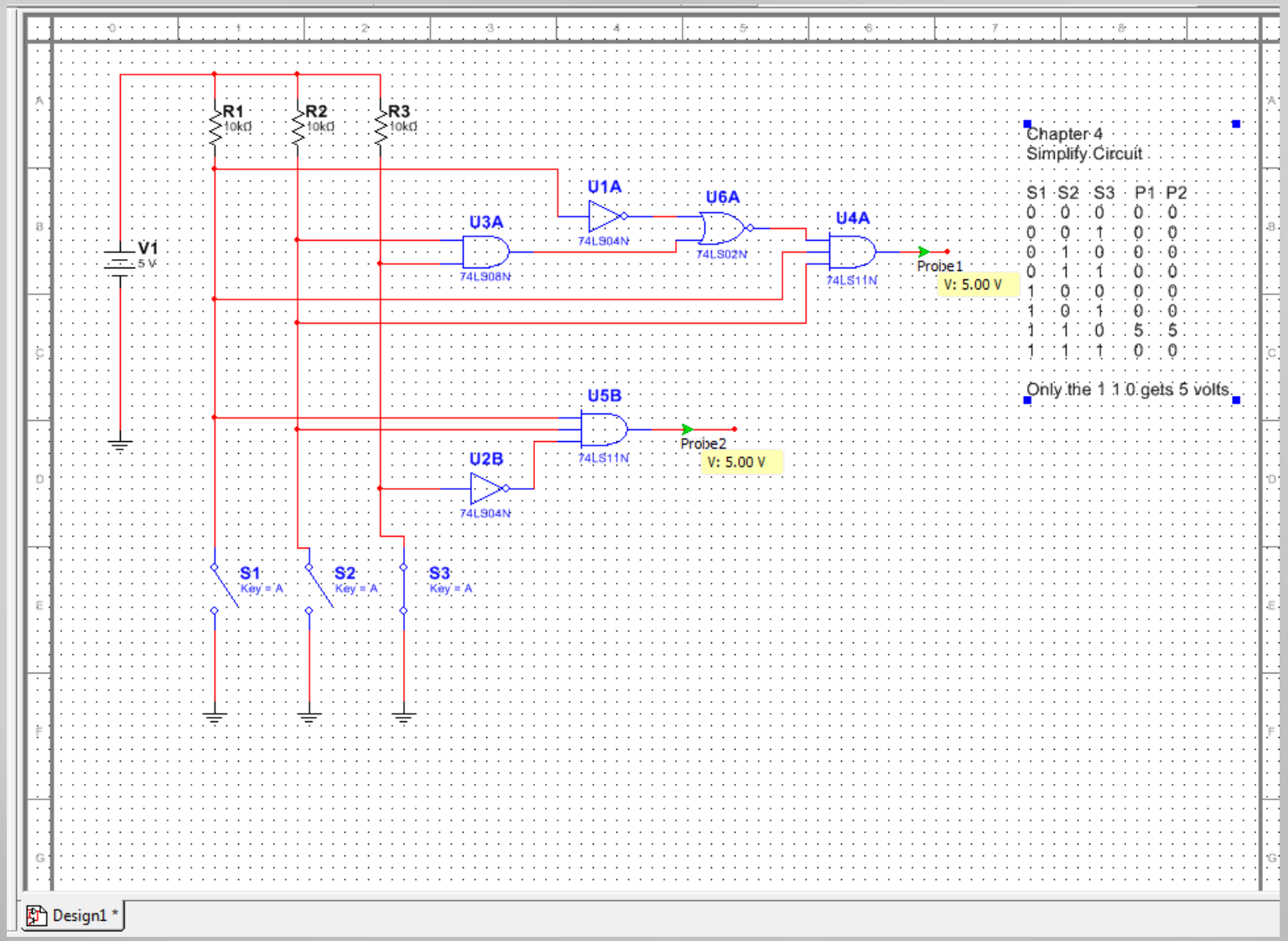
Universality of NAND and NOR Gates



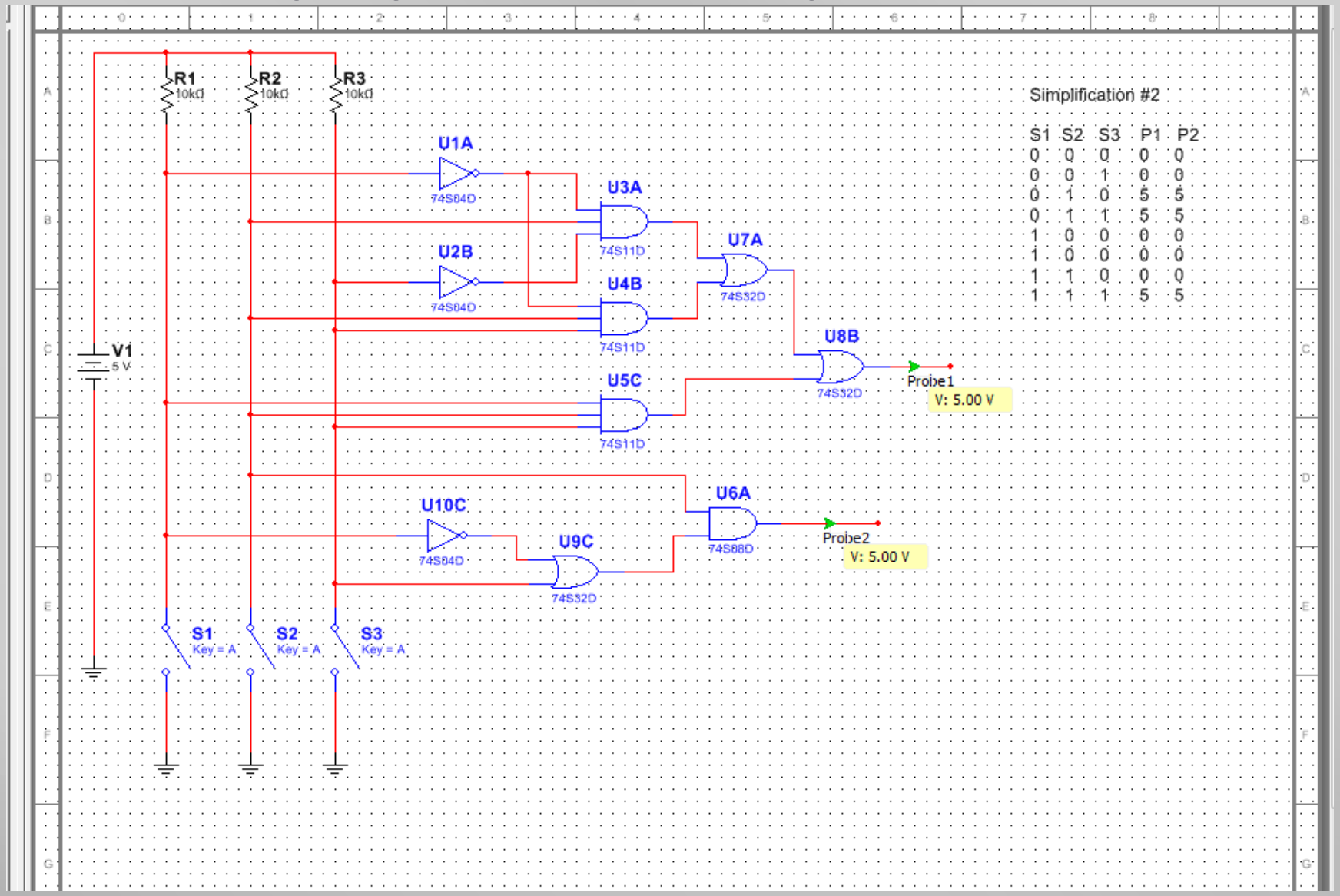
1	INPUTS			Out
S1	S2	S3	S4	
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	5
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	5
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	5
1	1	0	0	5
1	1	0	1	5
1	1	1	0	5
1	1	1	1	5

	INPUTS			Out
S5	S6	S7	S8	
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	5
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	5
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	5
1	1	0	0	5
1	1	0	1	5
1	1	1	0	5
1	1	1	1	5

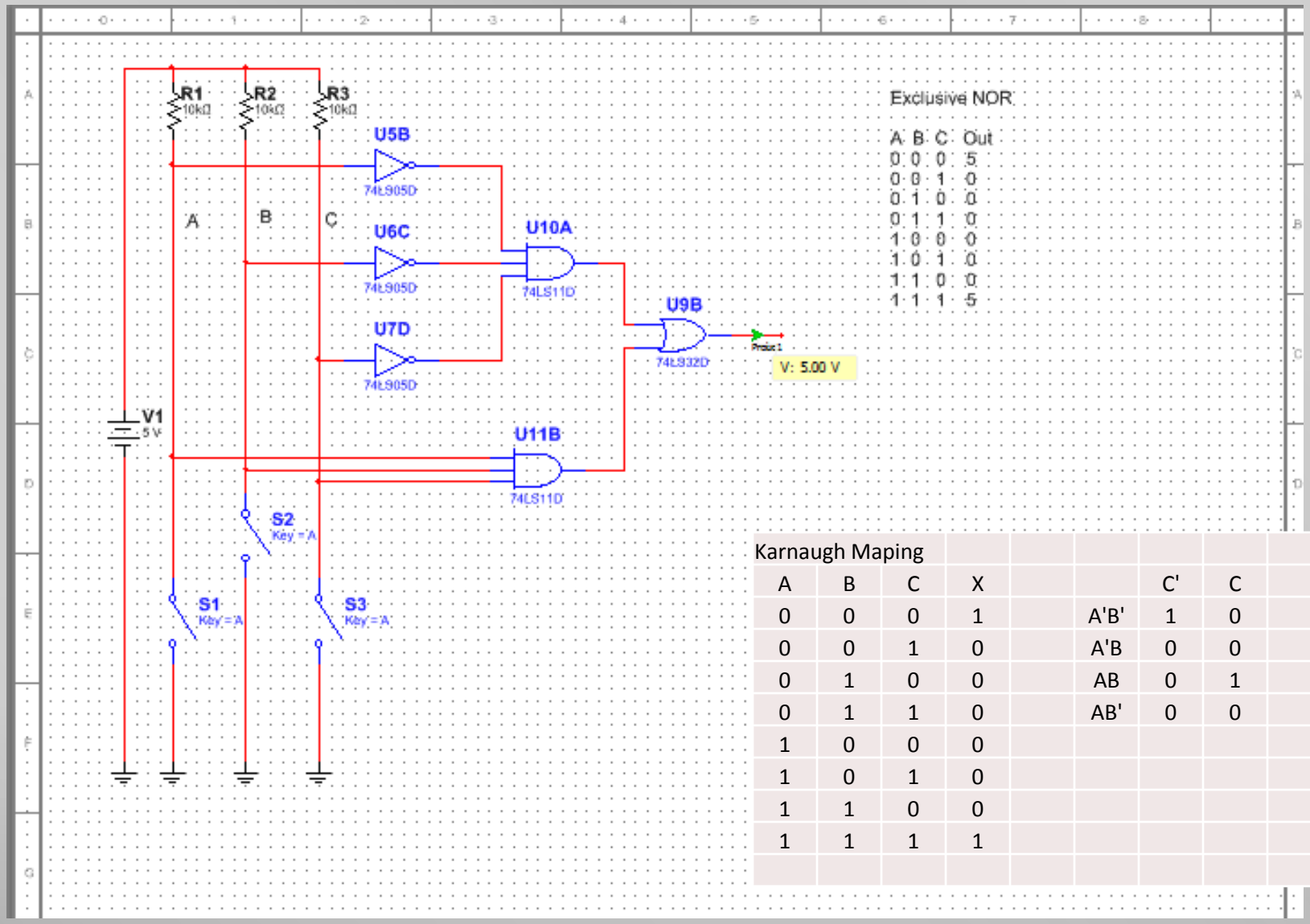
Simplify Circuit Example #1



Simplify Circuit Example #2



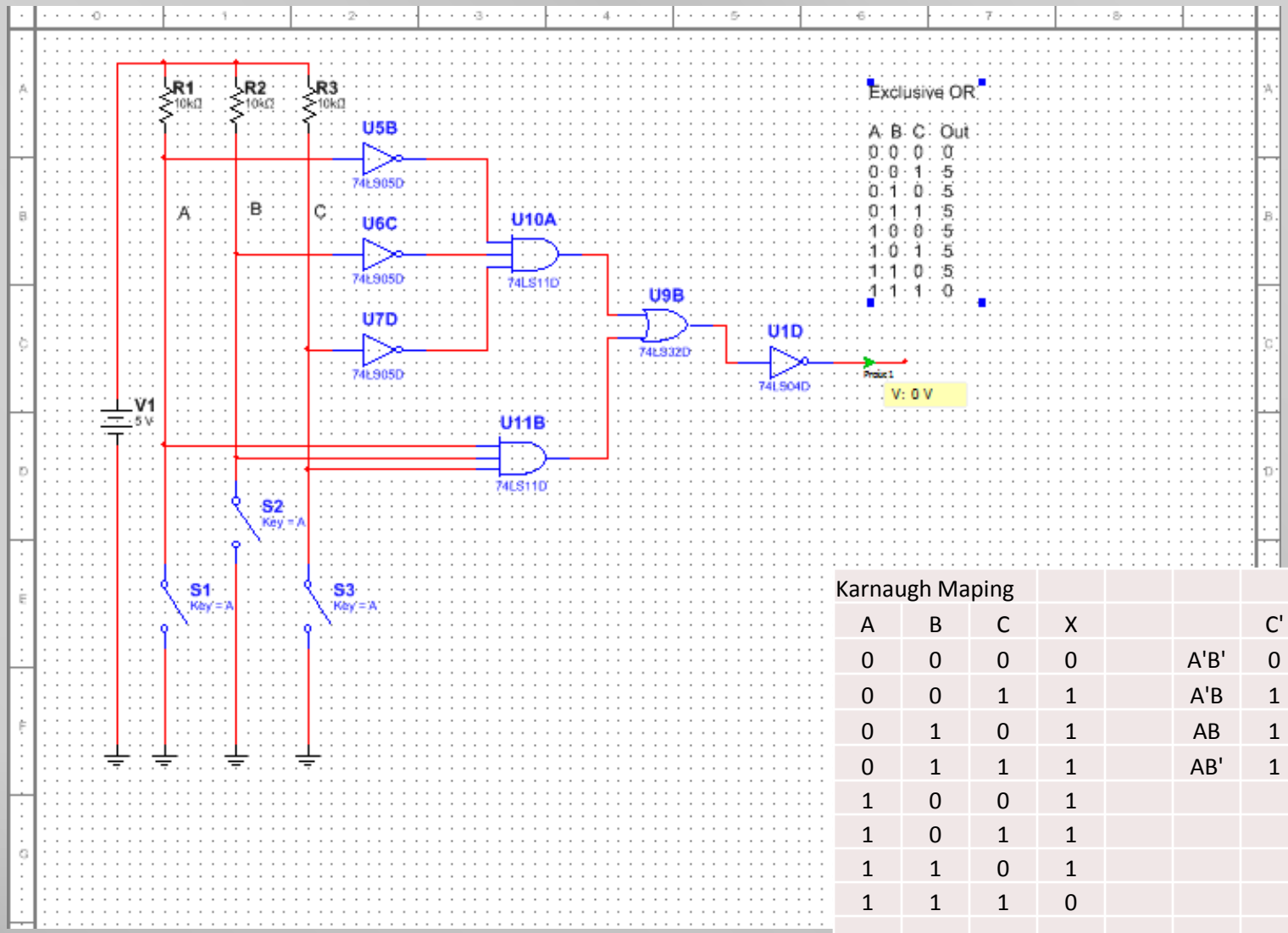
Exclusive NOR



Karnaugh Maping

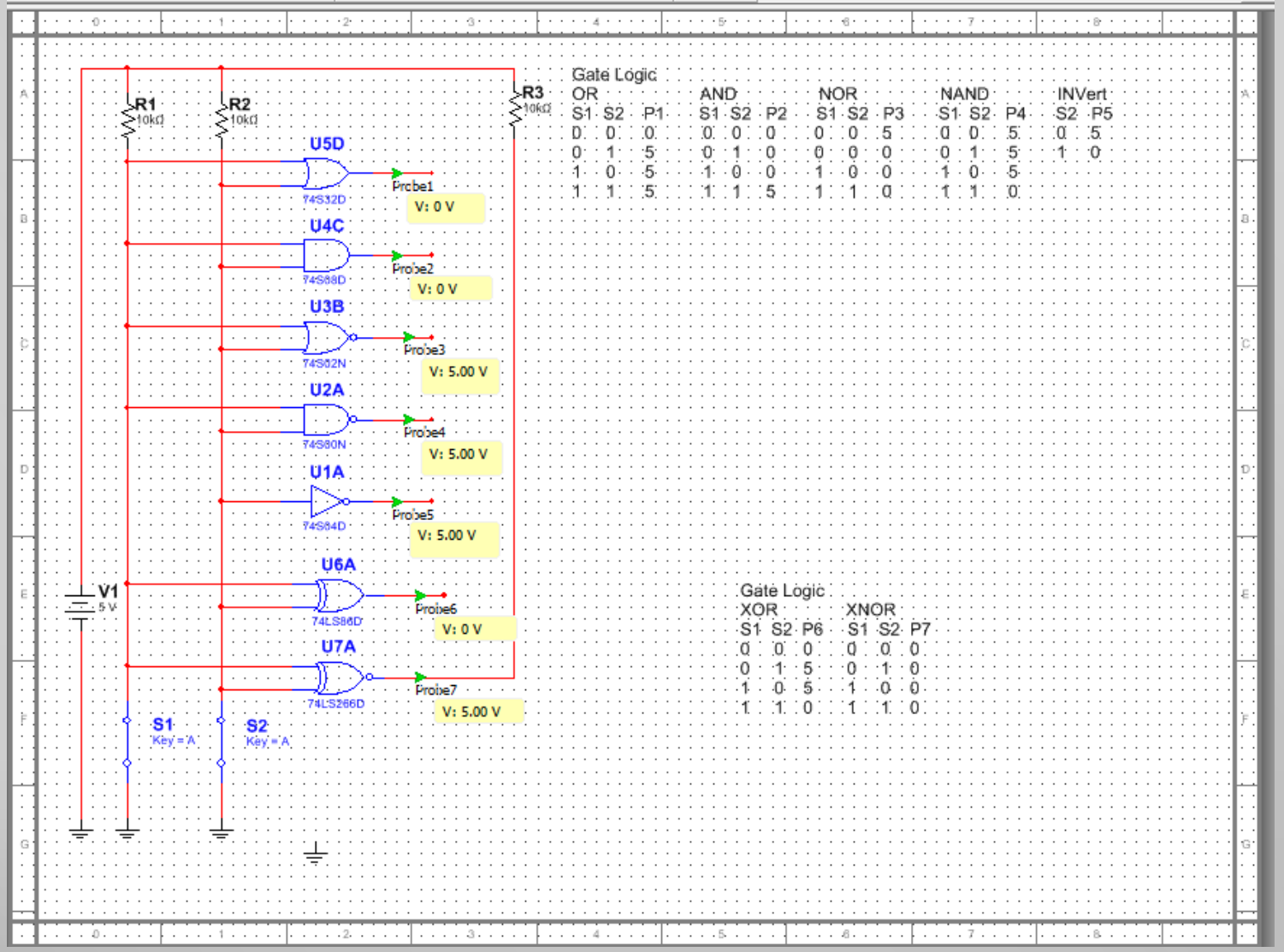
A	B	C	X			C'	C	
0	0	0	1		A'B'	1	0	
0	0	1	0		A'B	0	0	
0	1	0	0		AB	0	1	
0	1	1	0		AB'	0	0	
1	0	0	0					
1	0	1	0					
1	1	0	0					
1	1	1	1					

Exclusive OR

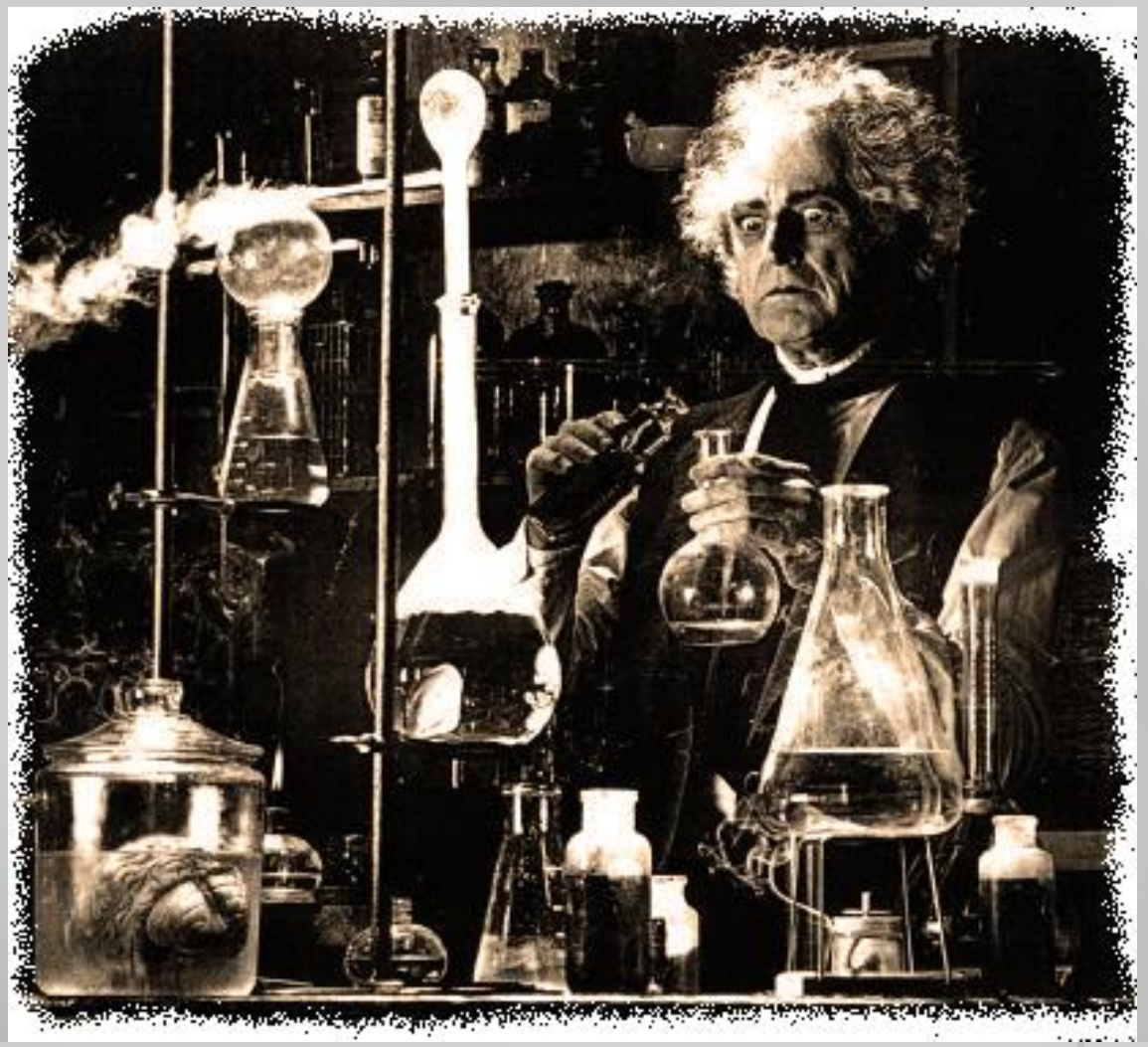


Karnaugh Maping							
A	B	C	X			C'	C
0	0	0	0		A'B'	0	1
0	0	1	1		A'B	1	1
0	1	0	1		AB	1	0
0	1	1	1		AB'	1	1
1	0	0	1				
1	0	1	1				
1	1	0	1				
1	1	1	0				

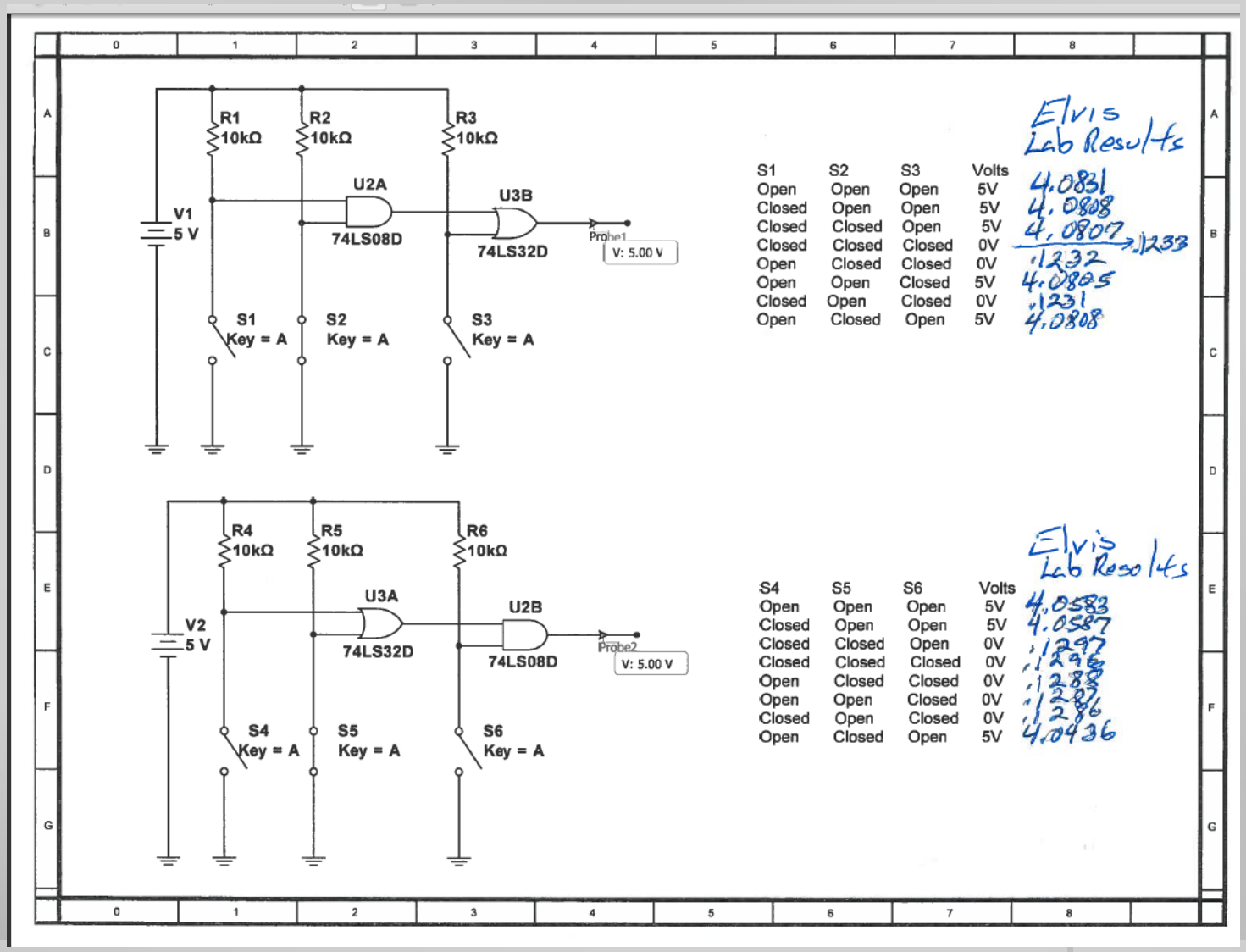
Gates with XOR & XNOR



LABS



ELVIS Lab with Multisim Verification



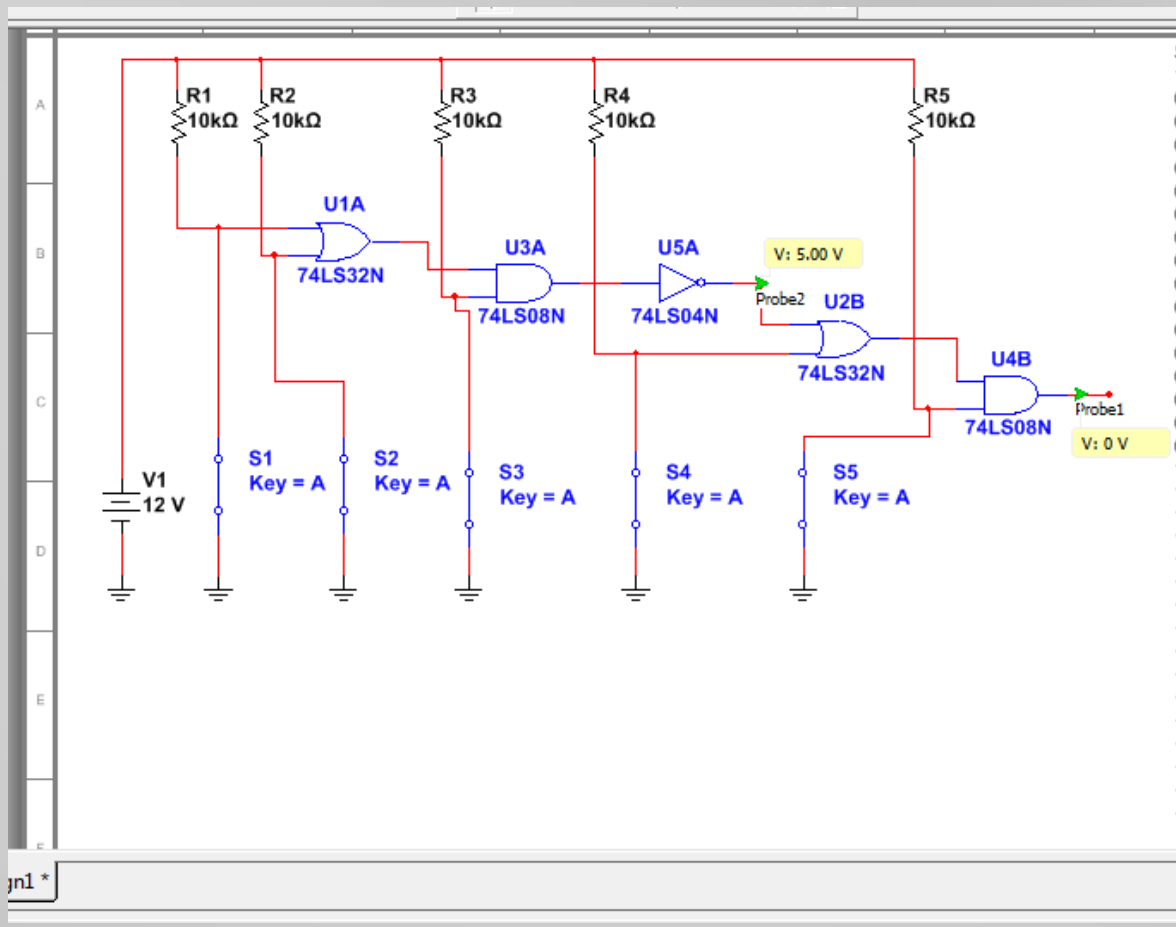
Elvis Lab Results

S1	S2	S3	Volts	
Open	Open	Open	5V	4.0831
Closed	Open	Open	5V	4.0808
Closed	Closed	Open	5V	4.0807 → .1233
Closed	Closed	Closed	0V	.1232
Open	Closed	Closed	0V	4.0805
Open	Open	Closed	5V	.1231
Closed	Open	Closed	0V	4.0808
Open	Closed	Open	5V	

Elvis Lab Results

S4	S5	S6	Volts	
Open	Open	Open	5V	4.0583
Closed	Open	Open	5V	4.0587
Closed	Closed	Open	0V	.1297
Closed	Closed	Closed	0V	.1296
Open	Closed	Closed	0V	.1288
Open	Open	Closed	0V	.1287
Closed	Open	Closed	0V	.1286
Open	Closed	Open	5V	4.0436

Lecture 3, Simulated on MultiSim and in LAB



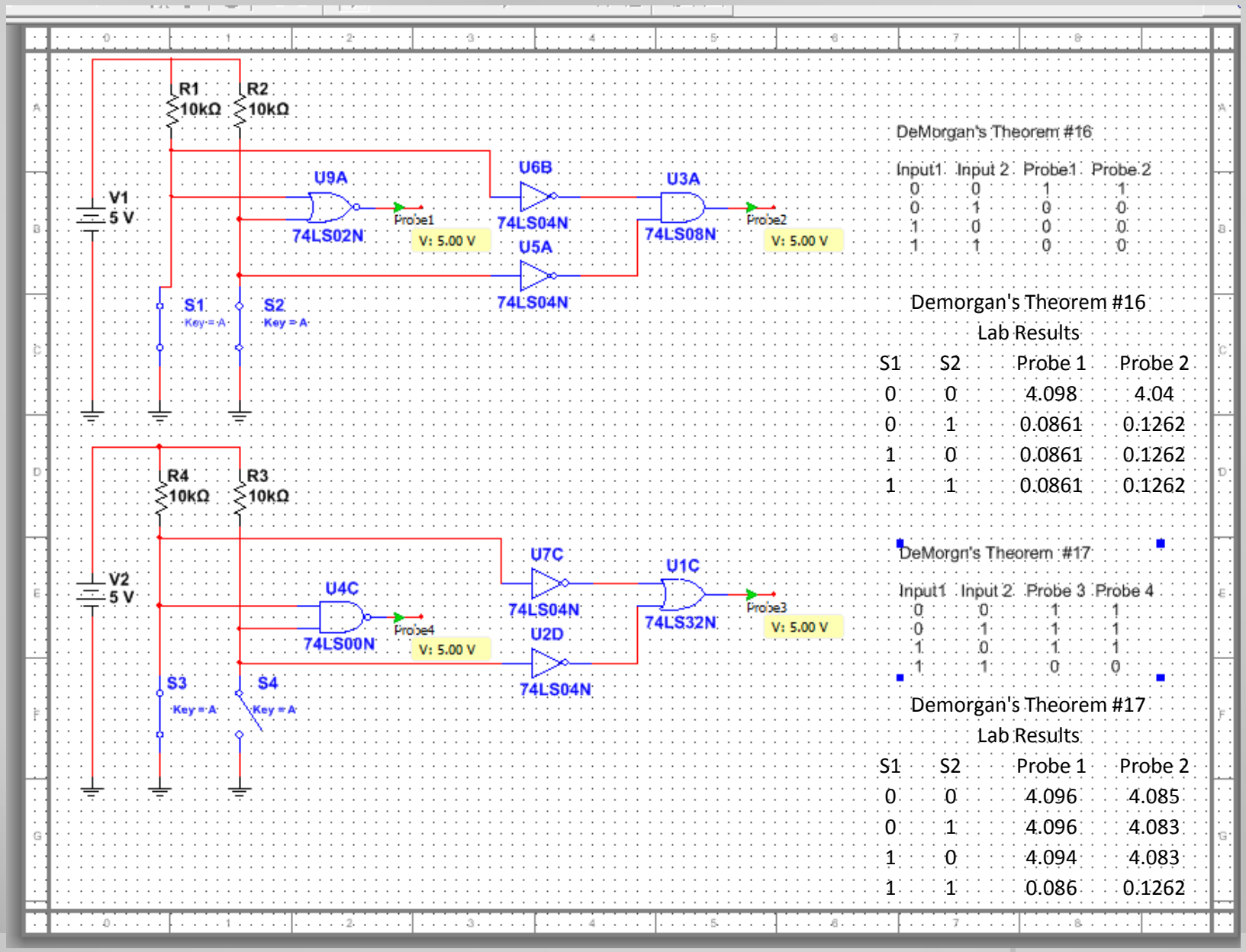
Switches					Probes	
1	2	3	4	5	2	1
0	0	0	0	0	0	5
0	0	0	0	1	0	0
0	0	0	1	0	0	0
0	0	0	1	1	0	0
0	0	1	0	0	5	5
0	0	1	0	1	5	0
0	0	1	1	0	5	5
0	0	1	1	1	5	0
0	1	0	0	0	0	5
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	0	1	1	0	0
0	1	1	0	0	5	5
0	1	1	0	1	5	0
0	1	1	1	0	5	5
0	1	1	1	1	5	0
1	0	0	0	0	0	5
1	0	0	0	1	0	0
1	0	0	1	0	0	0
1	0	0	1	1	0	0
1	0	1	0	0	5	5
1	0	1	0	1	5	0
1	0	1	1	0	5	5
1	0	1	1	1	5	0
1	1	0	0	0	5	5
1	1	0	0	1	5	0
1	1	0	1	0	5	5
1	1	0	1	1	5	0
1	1	1	0	0	5	5
1	1	1	0	1	5	0
1	1	1	1	0	5	5
1	1	1	1	1	5	0

LAB Lecture 3

5 Gate Input Results

S1	S2	S3	S4	S5	Probe
0	0	0	0	0	5.156
0	0	0	0	1	0.1342
0	0	0	1	0	0.1362
0	0	0	1	1	0.1343
0	0	1	0	0	5.161
0	0	1	0	1	0.1336
0	0	1	1	0	5.111
0	0	1	1	1	0.1337
0	1	0	0	0	5.174
0	1	0	0	1	0.1337
0	1	0	1	0	0.1339
0	1	0	1	1	0.1339
0	1	1	0	0	5.177
0	1	1	0	1	0.1335
0	1	1	1	0	5.178
0	1	1	1	1	0.1334
1	0	0	0	0	5.176
1	0	0	0	1	0.1336
1	0	0	1	0	0.1336
1	0	0	1	1	0.1338
1	0	1	0	0	5.178
1	0	1	0	1	0.1334
1	0	1	1	0	5.178
1	0	1	1	1	0.1334
1	1	0	0	0	5.174
1	1	0	0	1	0.1334
1	1	0	1	0	5.174
1	1	0	1	1	0.1335
1	1	1	0	0	5.188
1	1	1	0	1	0.1333
1	1	1	1	0	5.182
1	1	1	1	1	0.1334

DeMorgan's Theorem, Simulated on MultiSim & Lab Results



DeMorgan's Theorem #16

Input1	Input 2	Probe1	Probe 2
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Demorgan's Theorem #16
Lab Results

S1	S2	Probe 1	Probe 2
0	0	4.098	4.04
0	1	0.0861	0.1262
1	0	0.0861	0.1262
1	1	0.0861	0.1262

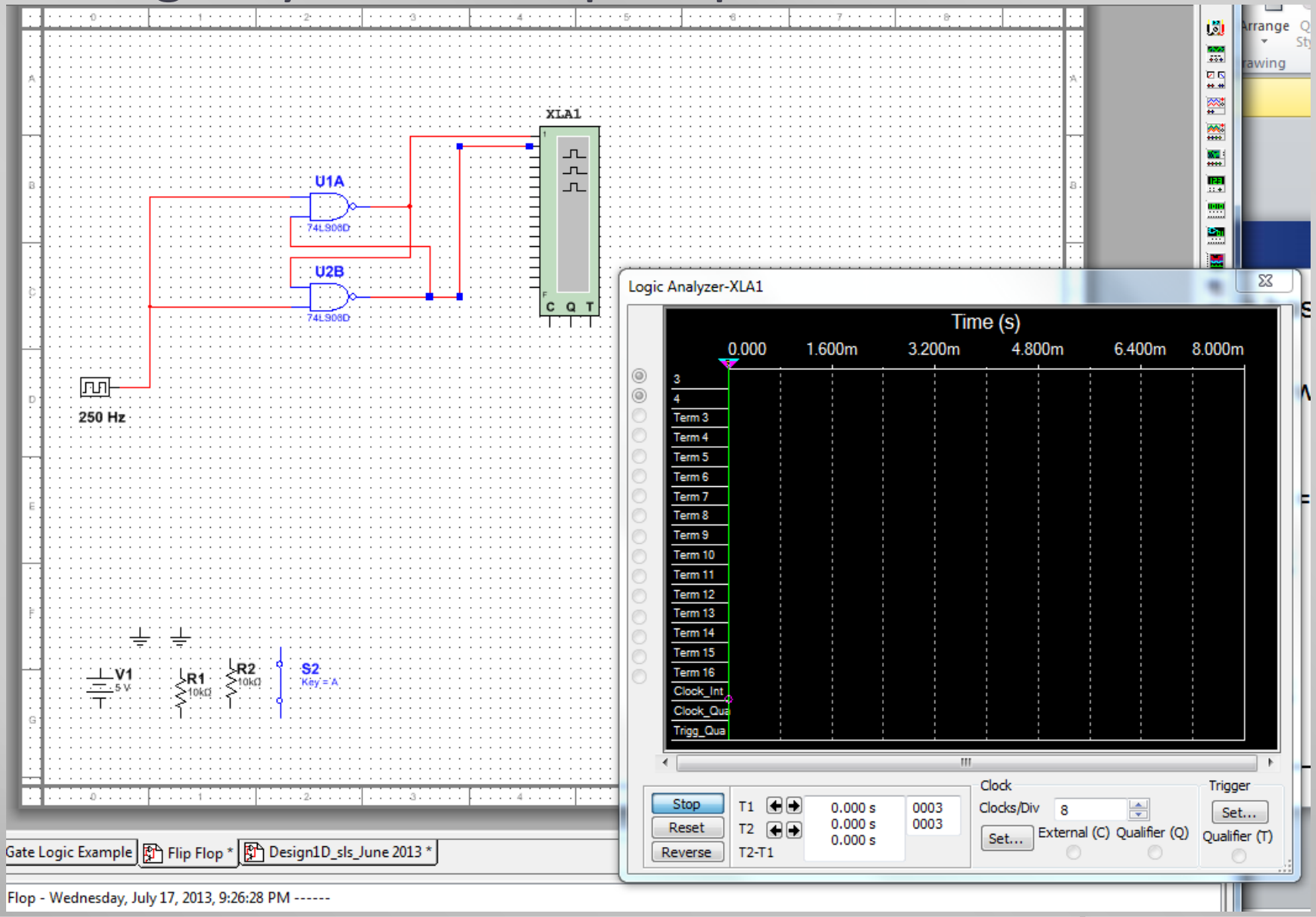
DeMorgn's Theorem #17

Input1	Input 2	Probe 3	Probe 4
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

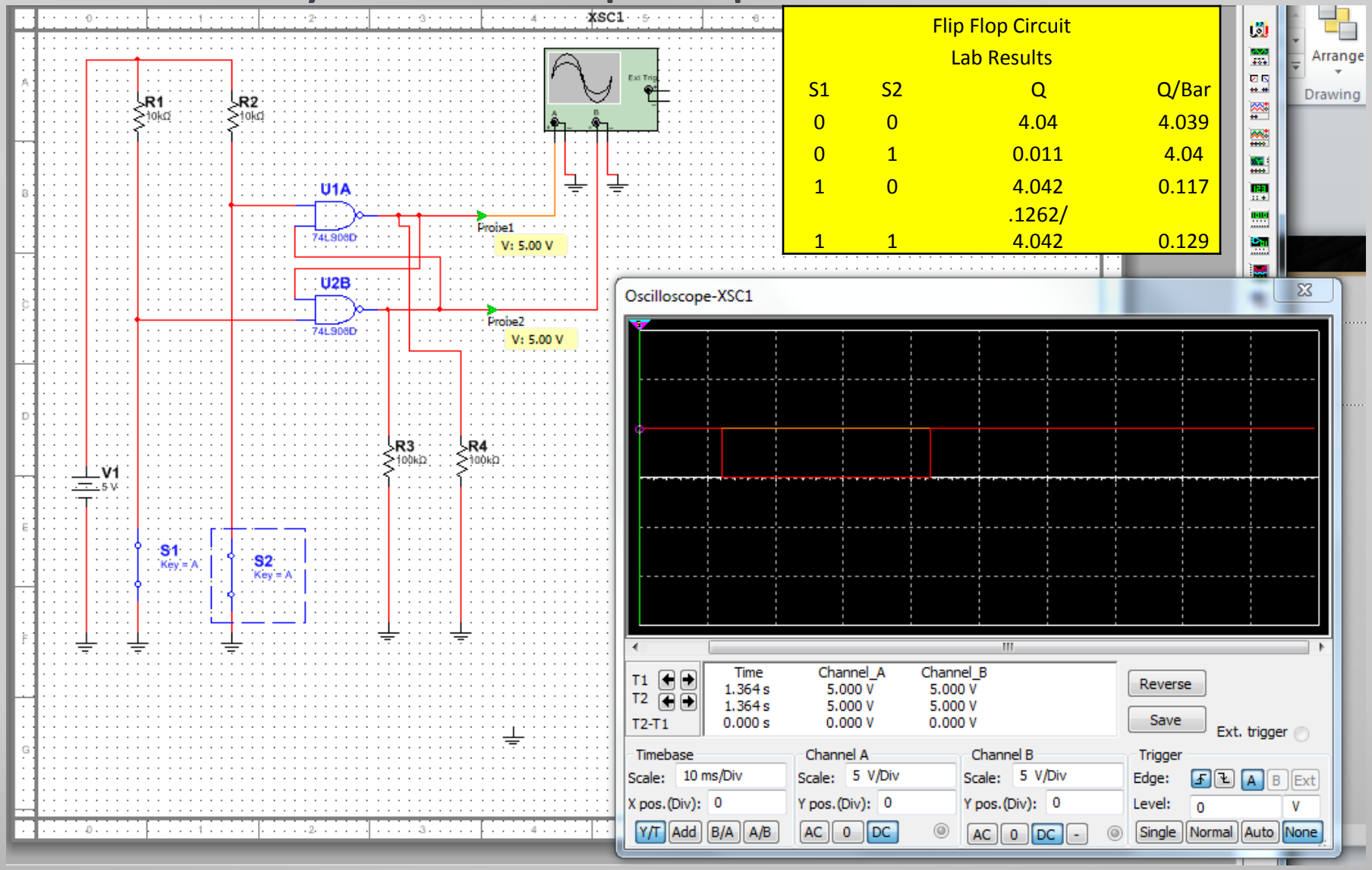
Demorgan's Theorem #17
Lab Results

S1	S2	Probe 1	Probe 2
0	0	4.096	4.085
0	1	4.096	4.083
1	0	4.094	4.083
1	1	0.086	0.1262

Wrong way to do a Flip Flop

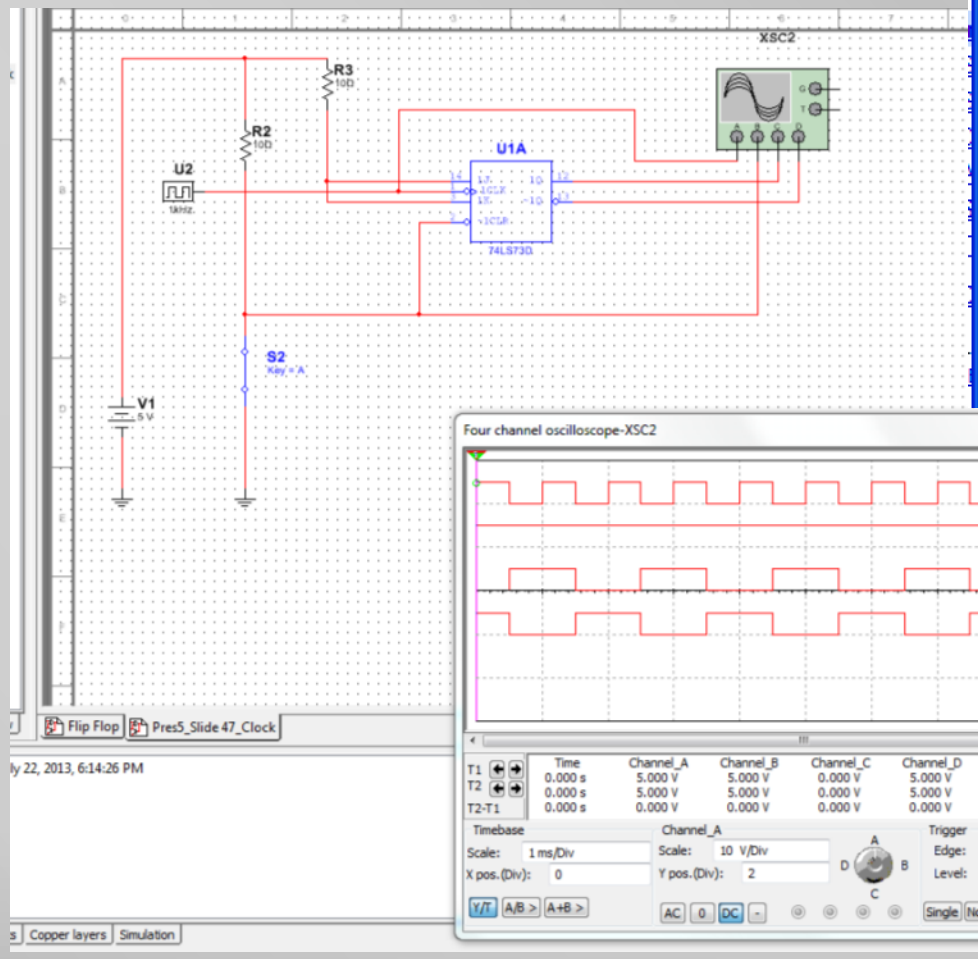


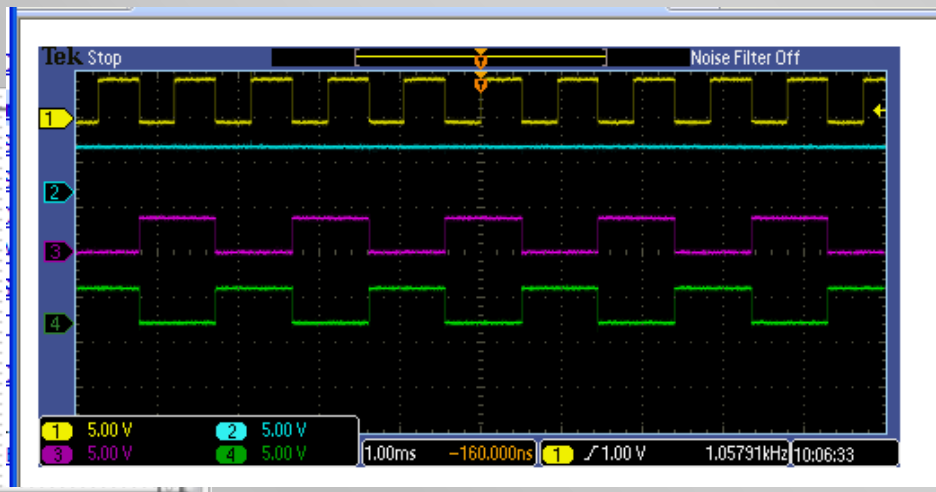
Correct Way to do a Flip Flop & Lab Results

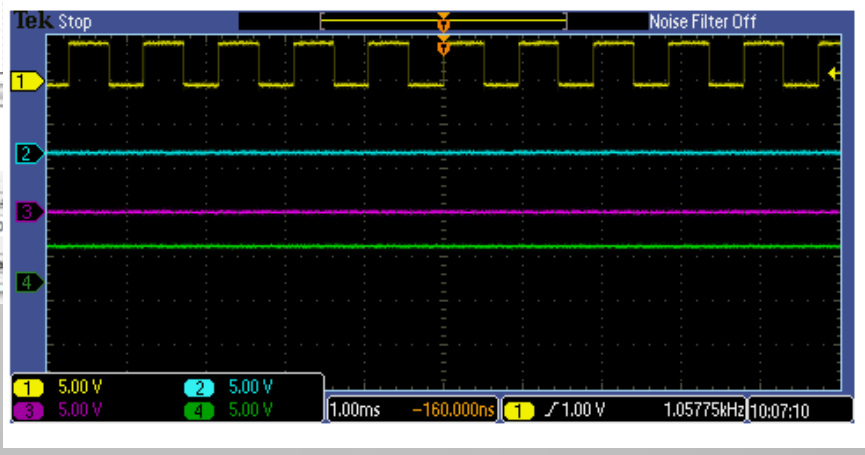


Flip Flop Circuit Lab Results			
S1	S2	Q	Q/Bar
0	0	4.04	4.039
0	1	0.011	4.04
1	0	4.042	0.117
1	1	.1262/ 4.042	0.129

Presentation 5: Slide 47 Lab Results







Test Results for DeMorgan from Elvis Simulation:

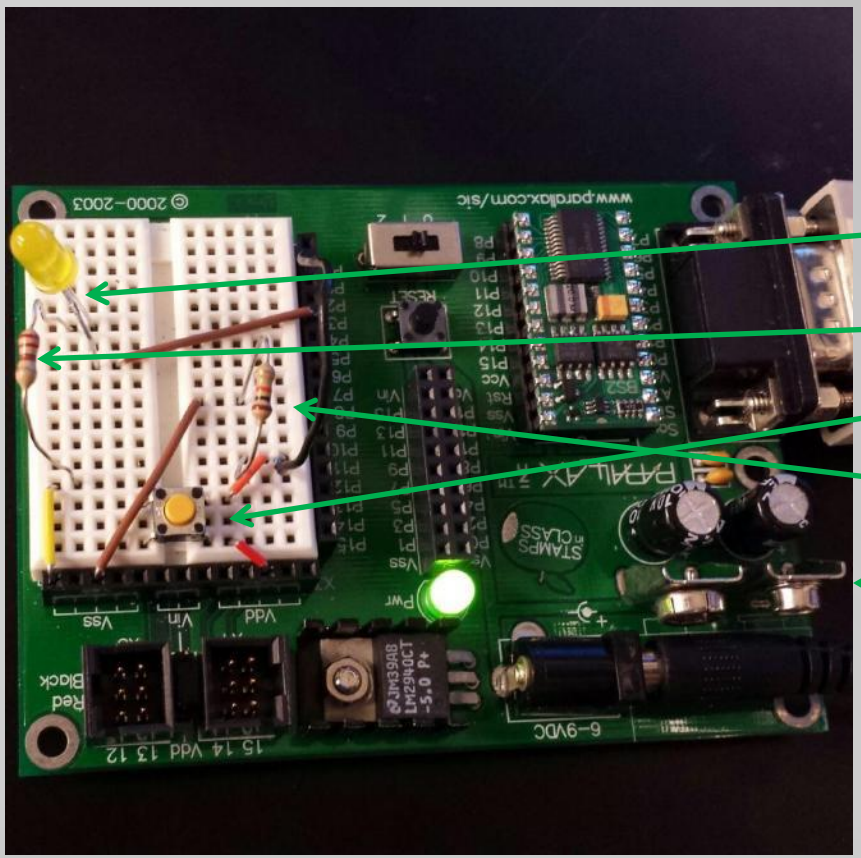
DeMorgan's Theorem #16						
Probe After NOR Gate				Probe After AND Gate		
Input 1	Input 2	Volts		Input 1	Input 2	Volts
0	0	4.018		0	0	4.04
0	1	0.0861		0	1	0.1262
1	0	0.0861		1	0	0.1262
1	1	0.0861		1	1	0.1263
DeMorgan's Theorem #17						
Probe After NAND Gate				Probe After OR Gate		
Input 1	Input 2	Volts		Input 1	Input 2	Volts
0	0	4.096		0	0	4.085
0	1	4.096		0	1	4.083
1	0	4.094		1	0	4.083
1	1	0.086		1	1	0.1262

LAB: Blinking LED on Basic Stamp 2; Programming

```
'{$STAMP BS2}
'PBASIC 2.0}
'{$PORT {$COM1}
'Program 1.1/;  Blinking LED Example
Cnt      VAR     Byte               'Variable for counting
PB1      VAR     IN1                'Variable for PB1 input
LED1     CON     4                  'Variable for LED1  output
INPUT 1                             'Set PB1 as input
OUTPUT 4                            'Set LED1 as output
LOW LED1                            'Turn off LED
Start:
  IF PB1 = 0 THEN Start              'Not Pressed? Go back to loop
  GOSUB Blink_LED1                   'If it was pressed then perform subroutine
GOTO Start                          'Afer return, go back to start
Blink_LED1:                         'Subroutine to blink LED 5 repetitions
  FOR Cnt = 1 TO 5                  'Setup loop for 5 counts
    HIGH LED1                       'Turn on LED
    PAUSE 1000                      'Wait 1 second
    LOW LED1                        'Turn off LED
    PAUSE 1000                      'wait 1 second
  NEXT                              'Repeat loop until done
RETURN                              'return back to after gosub call
```

See Next Slide . . .

Blinking LED LAB using Basic Stamp 2 & BoE



Components:

1. LED Yellow
2. 220 ohm resistor
3. Push button
4. 10 k-ohm resistor
5. Board of Education

To watch a video of our blinking LED go to:

http://www.youtube.com/watch?v=zJ55SocmE8Y

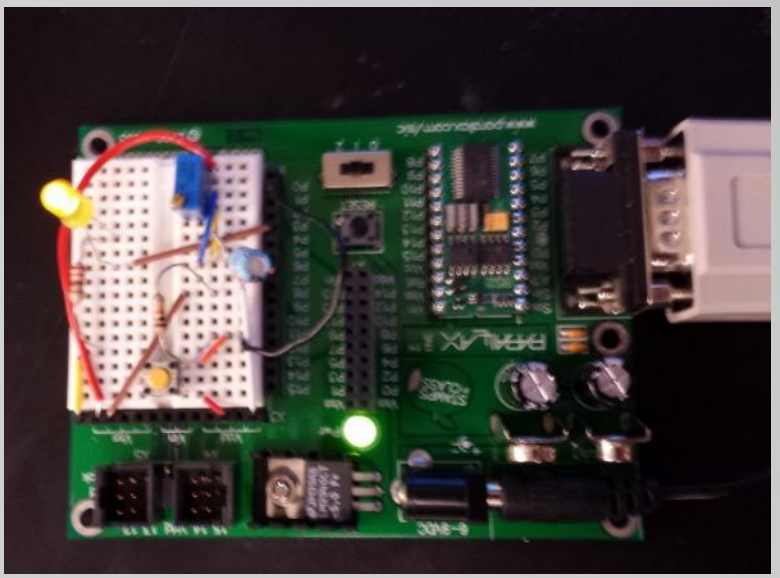
Basic Stamp 2, Temperature Control check using a potentiometer.

Program ->

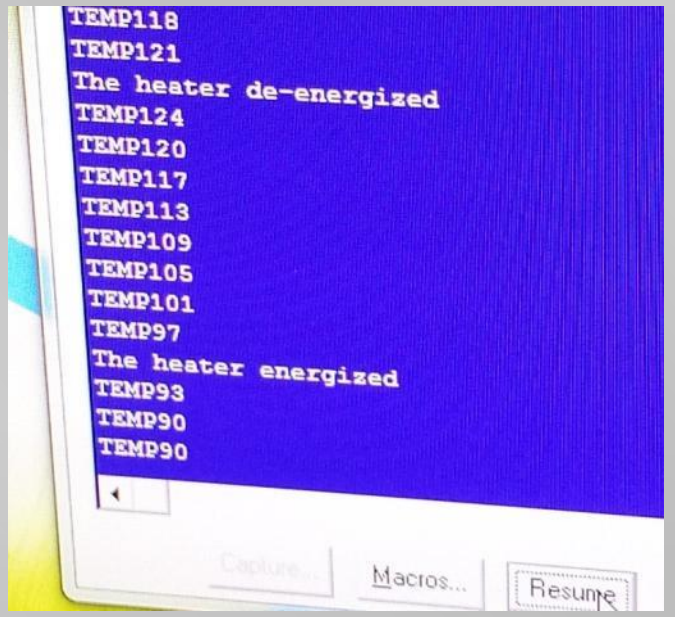
See Next Slide

```
Basic Stamp 2, Temp change with potentiometer
'{$STAMP BS2}
'Program 1.2, Simple Heater
LED1     VAR    OUT4
RC       CON    7
Temp     VAR    Word

OUTPUT 4
LED1 = 1

Main:
  GOSUB ReadTemp
  GOSUB Checktemp
  PAUSE 250
GOTO Main

ReadTemp
  HIGH RC
  PAUSE 10
  RCTIME RC, 1, Temp
  Temp = Temp/30

  DEBUG "Temp = ",dec Temp, CR
RETURN

CheckTemp:

  IF (Temp > 100) OR (LED1 = 1) THEN Checkoff
  LED1 = 1
  DEBUG "The heater energized", CR

Checkoff:
  IF (Temp < 120) OR (LED1 = 0) THEN Checkdone
  LED1 = 0
  DEBUG "The heater de-energized", CR
Checkdone:
RETURN
```

Heater Control



Adding a potentiometer (10K) and a capacitor (1 Uf) to the original Blinking light circuit, we were able to turn the LED on and off using the potentiometer. Of course a new program was written to control this as well. See Below:



See the video of how this works at:

http://youtu.be/hX5S-4GpeyI

Heater Control Programming with Stamp Plot Lite

Screen Shot of changing the potentiometer

Heater Display with StampPlot Lite

```
'{$STAMP BS2}
'Program 1.2, Simple Heater
PAUSE 500
DEBUG "!SPAN 50,150",CR
DEBUG "!TMAX 60",CR
DEBUG "!PNTS 500",CR
DEBUG "!TITL Simple Heater Control",CR
DEBUG "!SHFT ON",CR
DEBUG "!TSMP ON",CR
DEBUG "!PLOT ON",CR
DEBUG "!RSET",CR

LED1    VAR   OUT4
RC      CON   7
Temp    VAR   Word

OUTPUT 4
LED1 = 1

Main:
  GOSUB ReadTemp
  GOSUB Checktemp
  PAUSE 250
GOTO Main

ReadTemp
  HIGH RC
  PAUSE 10
  RCTIME RC, 1, Temp
  Temp = Temp/30

  DEBUG DEC Temp, CR
  DEBUG IBIN LED1,CR

  DEBUG "Temp = ",DEC Temp, CR
RETURN

CheckTemp:

  IF (Temp > 100) OR (LED1 = 1) THEN Checkoff
  LED1 = 1
  DEBUG "The heater energized", CR
  DEBUG "!USRC The heater is energized!",CR

Checkoff:
  IF (Temp < 120) OR (LED1 = 0) THEN Checkdone
  LED1 = 0
  DEBUG "The heater de-energized", CR
  DEBUG "!USRS The heater is de-energized!",CR
Checkdone:
RETURN
```

Blinking Binary Code: Programming

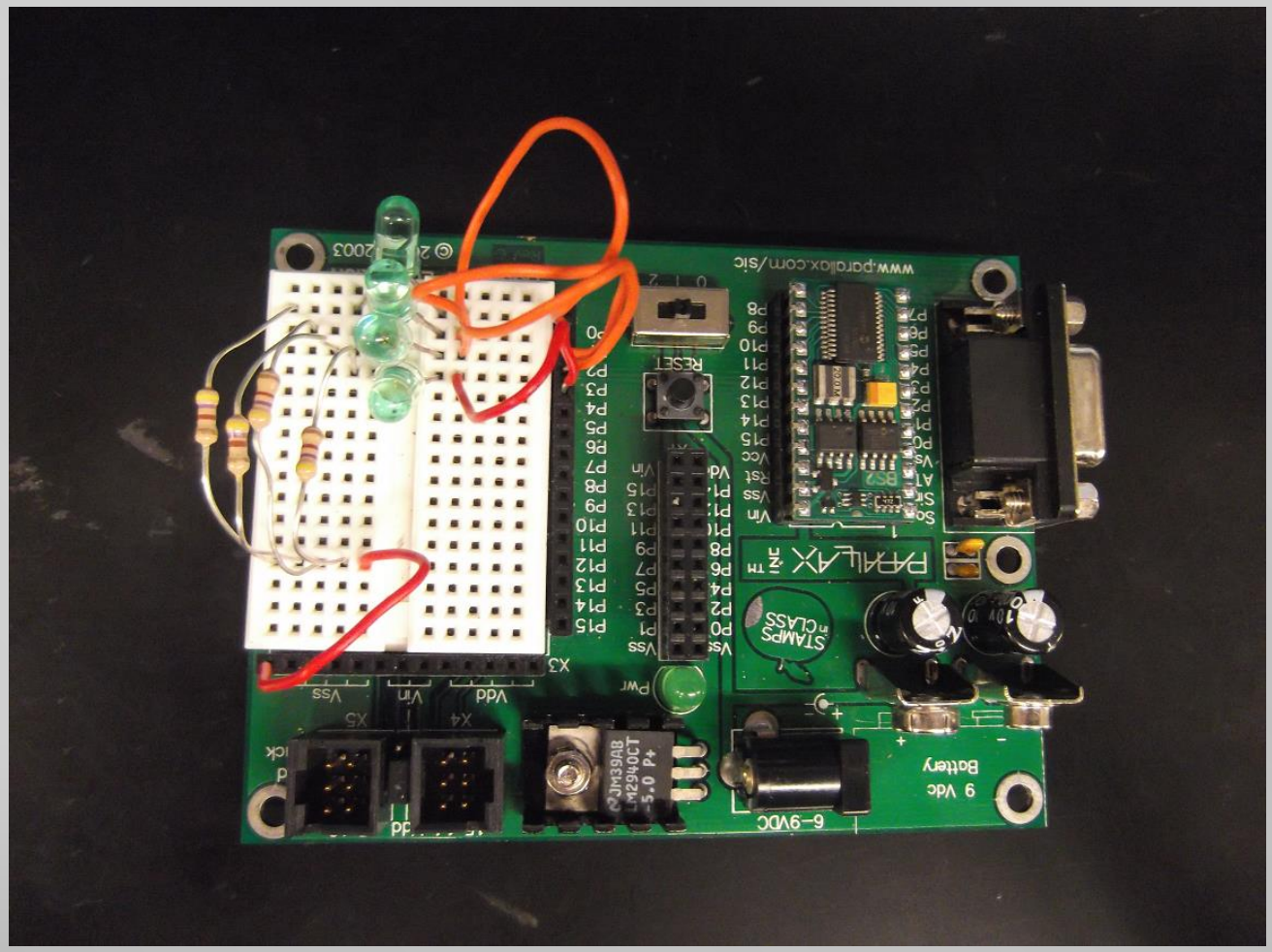
This is a little hard to read but to follow the columns.

See the link below to the Youtube video.

http://youtu.be/Cm8NYPWkp_s

```
PROGRAM FOR BINARY COUNTER FOR BASIC STAMP 2  JULY 25, 2013 JEFF NOGGLE & STEVE SMITH

'{$STAMP BS2}
Cnt  VAR  Byte
LED1  CON  1
LED2  CON  2
LED3  CON  3
LED4  CON  4

OUTPUT 1
OUTPUT 2
OUTPUT 3
OUTPUT 4

LOW  LED1
LOW  LED2
LOW  LED3
LOW  LED4

START:
 LOW LED1
 PAUSE 1500

 HIGH LED1
 PAUSE 1500
 LOW LED1
 PAUSE 1500

 HIGH LED2
 PAUSE 1500
 LOW LED2
 PAUSE 1500

 HIGH LED1
 HIGH LED2
 PAUSE 1500
 LOW LED1
 LOW LED2
 PAUSE 1500

 HIGH LED3
 PAUSE 1500
 LOW LED3
 PAUSE 1500

 HIGH LED3
 HIGH LED1
 PAUSE 1500
 LOW LED3
 LOW LED1
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 PAUSE 1500
 LOW LED3
 LOW LED2
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED1
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED1
 PAUSE 1500

 HIGH LED4
 PAUSE 1500
 LOW LED4
 PAUSE 1500

 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED1
 LOW LED4
 PAUSE 1500

 HIGH LED2
 HIGH LED4
 PAUSE 1500
 LOW LED2
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED1
 LOW LED4
 PAUSE 1500

Cnt  VAR  Byte
LED1  CON  1
LED2  CON  2
LED3  CON  3
LED4  CON  4

OUTPUT 1
OUTPUT 2
OUTPUT 3
OUTPUT 4

LOW  LED1
LOW  LED2
LOW  LED3
LOW  LED4

START:
 HIGH LED1
 PAUSE 1500
 LOW LED1
 PAUSE 1500

 HIGH LED2
 PAUSE 1500
 LOW LED2
 PAUSE 1500

 HIGH LED1
 HIGH LED2
 PAUSE 1500
 LOW LED1
 LOW LED2
 PAUSE 1500

 HIGH LED3
 PAUSE 1500
 LOW LED3
 PAUSE 1500

 HIGH LED3
 HIGH LED1
 PAUSE 1500
 LOW LED3
 LOW LED1
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 PAUSE 1500
 LOW LED3
 LOW LED2
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED1
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED1
 PAUSE 1500

 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED1
 LOW LED4
 PAUSE 1500

 HIGH LED2
 HIGH LED4
 PAUSE 1500
 LOW LED2
 LOW LED4
 PAUSE 1500

 HIGH LED1
 HIGH LED2
 HIGH LED4
 PAUSE 1500
 LOW LED1
 LOW LED2
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED1
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED2
 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED1
 LOW LED4
 PAUSE 1500

 HIGH LED3
 HIGH LED1
 HIGH LED4
 PAUSE 1500
 LOW LED3
 LOW LED2
 LOW LED1
 LOW LED4
 PAUSE 1500

RETURN
```

Picture of our Binary Code LED Flasher



FINAL PROJECT

7 Segment LED displaying a sequential count of the hexadecimal numbers 1 thru 9 then A thru F.
This is followed by tones from a speaker playing . . .
Frere Jacques!

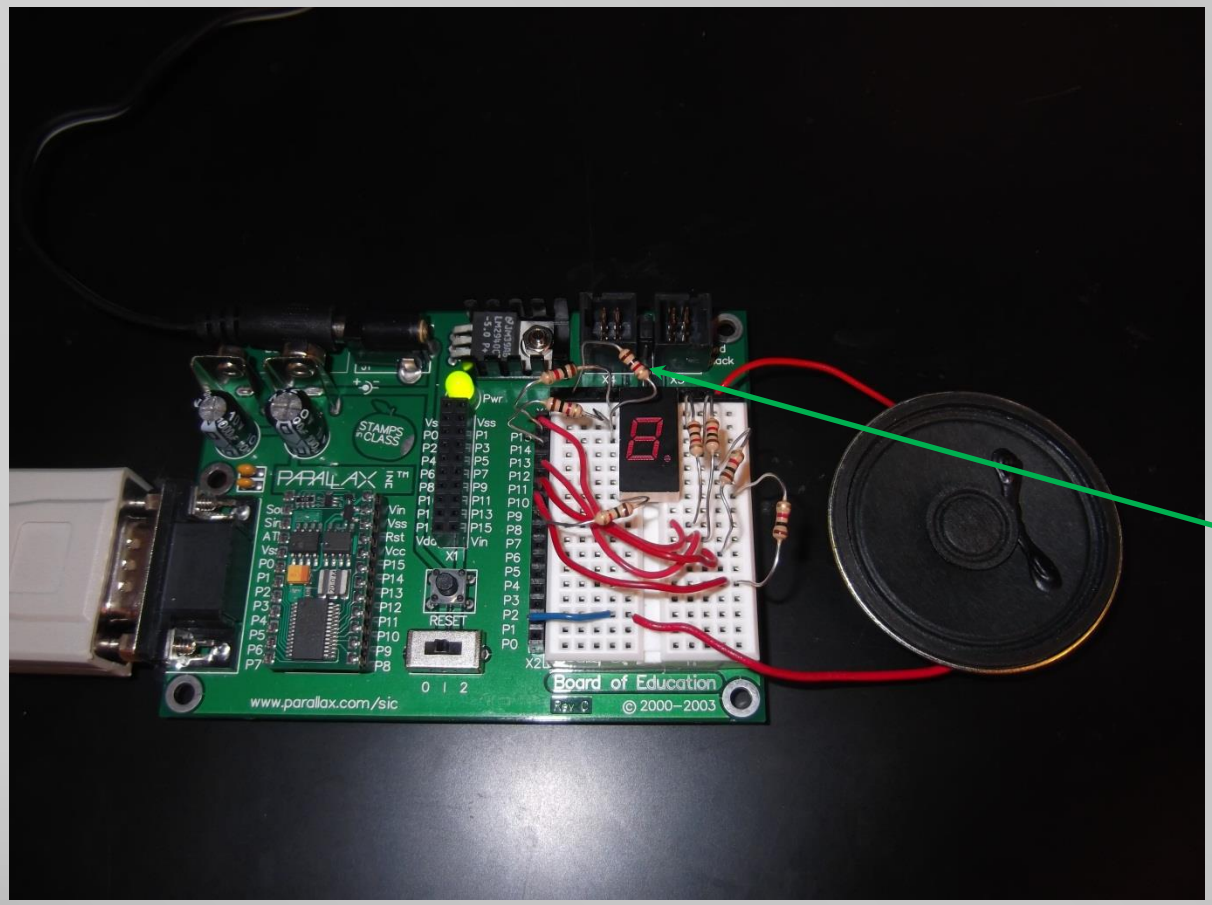
Final Project Program Code for 7 segment LED with Speaker Output

```
EECT 112 Final Project:  Jeff Noggle & Steve Smith

 Display the digits 0 through 9, and A thru F on a 7-segment LED display, then play Play the first few notes from Frere Jacques

'{$STAMP BS2}
'{$PBASIC 2.5}

DEBUG "Program Running!"

OUTH = %00000000              ' OUTH initialized to low.
DIRH = %11111111              ' Set P8-P15 to all output-low.
                              ' Digit:
' BAFG.CDE
OUTH = %00011000              ' 0
PAUSE 1500
OUTH = %01111011              ' 1
PAUSE 1500
OUTH = %00101100              ' 2
PAUSE 1500
OUTH = %00101001              ' 3
PAUSE 1500
OUTH = %01001011              ' 4
PAUSE 1500
OUTH = %10001001              ' 5
PAUSE 1500
OUTH = %10001000              ' 6
PAUSE 1500
OUTH = %00111011              ' 7
PAUSE 1500
OUTH = %00001000              ' 8
PAUSE 1500
OUTH = %00001001              ' 9
PAUSE 1500
OUTH = %00001010              ' A
PAUSE 1500
OUTH = %11001000              ' b
PAUSE 1500
OUTH = %10011100              ' C
PAUSE 1500
OUTH = %01101000              ' d
PAUSE 1500
OUTH = %10001100              ' E
PAUSE 1500
OUTH = %10001110              ' F
PAUSE 1500
```

(Program continued)

```
DIRH = %00000000              ' I/O pins to input,
                              ' segments off.

 Play the first few notes from Frere Jacques.

Notes     DATA    "C","D","E","C","C","D","E","C","E","F",
"G","E","F","G","Q"
Durations    DATA    4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
2, 4, 4, 2
WholeNote    CON    2000

index     VAR    Byte
offset     VAR    Nib

noteLetter    VAR    Byte
noteFreq    VAR    Word
noteDuration  VAR    Word

DO UNTIL noteLetter = "Q"

  READ Notes + index, noteLetter

  LOOKDOWN  noteLetter,  [ "A", "b", "B", "C", "d",
  "D", "e", "E", "F", "g",
  "G", "a", "P", "Q" ], offset

  LOOKUP offset,     [ 1760, 1865, 1976, 2093, 2217,
2349, 2489, 2637, 2794, 2960,
3136, 3322, 0, 0 ], noteFreq

  READ Durations + index, noteDuration

  noteDuration = WholeNote / noteDuration

  FREQOUT 1, noteDuration, noteFreq

  index = index + 1

LOOP
END
```

Picture: Board of Education Final Project



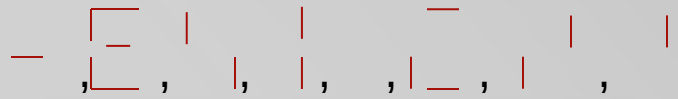
Materials:
7 – 1K resitors
1 – 7 segment LED
LN513RA
1 – 8 Ohm Speaker
Many Wires

We removed the one resistor and changed to a direct wire to get a brighter display.

Please enjoy Frere Jacques !!!!

Trials and Tribulations of the Final Project

We found that we were still having an issue with the display and it would not show the 0 thru F as we wanted to see it.
Finally we wrote down what we were seeing and noticed a pattern:



which was giving us the missing parts of the numbers.

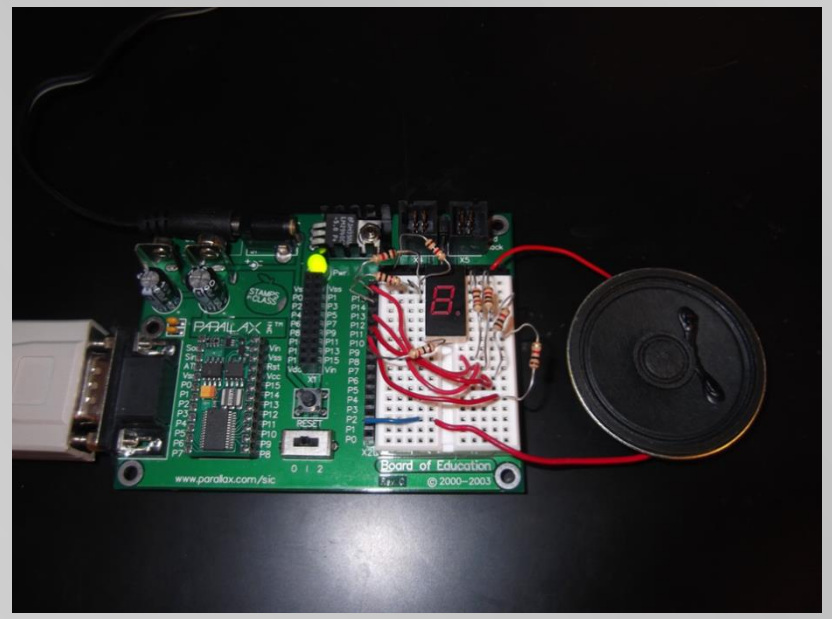
Therefore we established we needed to reverse what our code was being sent to the controller. Once we switched the program for our data strings we got the correct numbers and letters showing on the 7 segment LED. We added the finale of Frere Jacques!

At one point, we also had a wire that was not fully inserted into the board for the ‘c’ segment of the LED but that was quickly rectified.

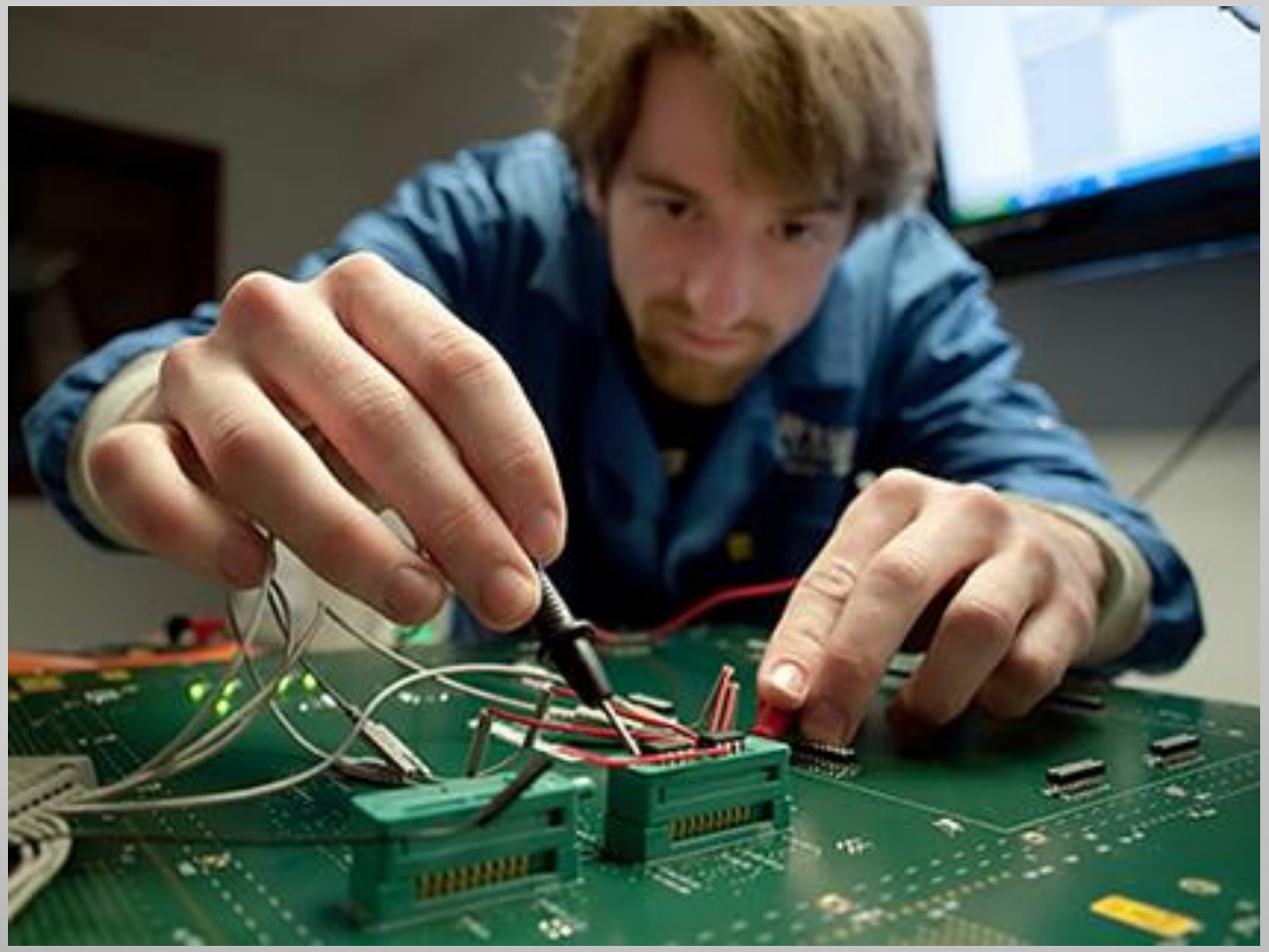
Go to next Slide for the link to the Video of our final project!

WOO HOO!!!

http://youtu.be/-DDEgVoWwuI



ADDITIONAL DATA



Alternate Logic-Gate Representations (From Class)

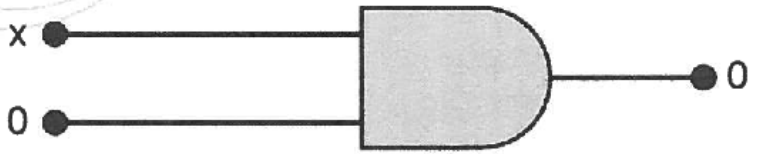
- To convert a standard symbol to an alternate:
  - Invert each input and output in standard symbols.

    Add an inversion bubble where there are none.

    Remove bubbles where they exist.

Boolean Theorems Continued…

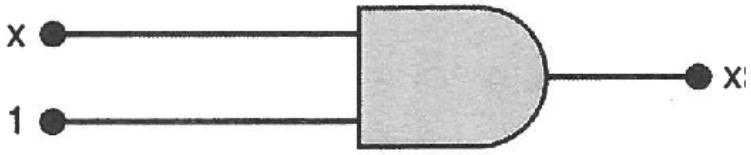
3-10 Boolean Theorems



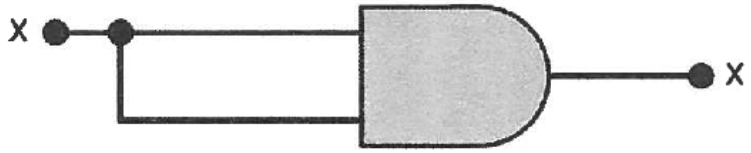
(1) $x \cdot 0 = 0$

Theorem (1) states that if any variable is ANDed with 0, the result must be 0.

Theorem (2) is also obvious by comparison with ordinary multiplication.



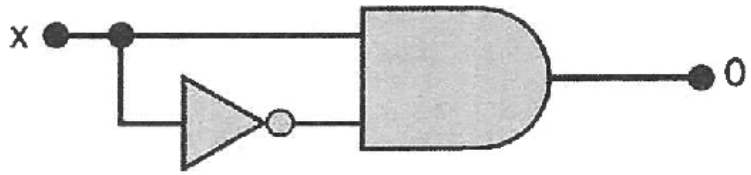
(2) $x \cdot 1 = x$



(3) $x \cdot x = x$

Prove Theorem (3) by trying each case.
If $x = 0$, then $0 \cdot 0 = 0$
If $x = 1$, then $1 \cdot 1 = 1$
Thus, $x \cdot x = x$

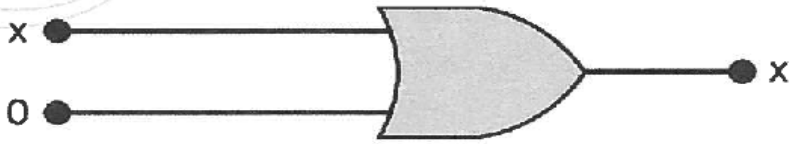
Theorem (4) can be proved in the same manner.



(4) $x \cdot \overline{x} = 0$

Boolean Theorems

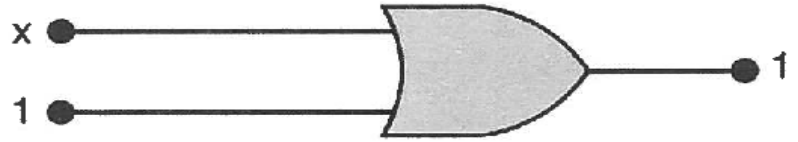
3-10 Boolean Theorems



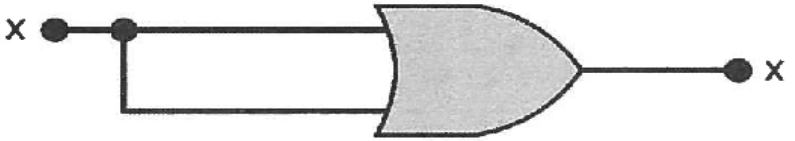
(5) $x + 0 = x$

Theorem (5) is straightforward, as 0 *added* to anything does not affect value, either in regular addition or in OR addition.

Theorem (6) states that if any variable is ORed with 1, the is always 1. Check values: 0 + 1 = 1 and 1 + 1 = 1.

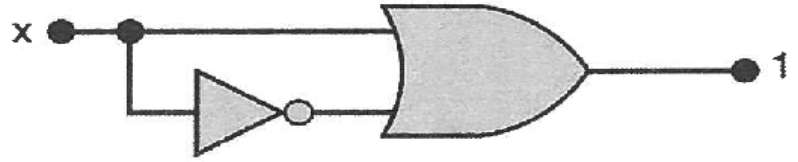


(6) $x + 1 = 1$



(7) $x + x = x$

Theorem (7) can be proved by checking for both values of x: 0 + 0 = 0 and 1 + 1 = 1.



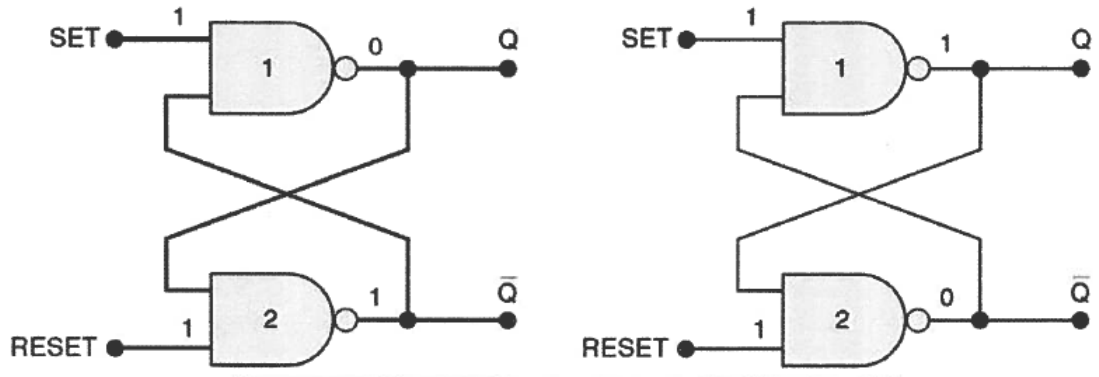
(8) $x + \overline{x} = 1$

Theorem (8) can be proved similarly.

FLIP-FLOP using NAND Gates

5-1 NAND Gate Latch

- The **NAND** gate latch or simply latch is a basic FF.
  - Inputs are *SET* and *CLEAR* (*RESET*).
- Inputs are active-LOW—output will change when the input is pulsed LOW.
  - When the latch is set: **Q** = 1 and $\overline{\mathbf{Q}}$ = 0
  - When the latch is clear or reset: **Q** = 0 and $\overline{\mathbf{Q}}$ = 1



The following pages contain Chip Diagrams 2 input NOR Gate

SDLS027

SN5402, SN54LS02, SN54S02, SN7402, SM74LS02, SN74S02
QUADRUPLE 2-INPUT POSITIVE-NOR GATES

DECEMBER 1983—REVISED MARCH 1988

- **Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

description

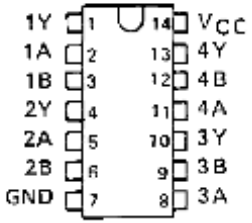
These devices contain four independent 2-input NOR gates.

The SN5402, SN54LS02, and SN54S02 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN7402, SN74LS02, and SN74S02 are characterized for operation from 0°C to 70°C.

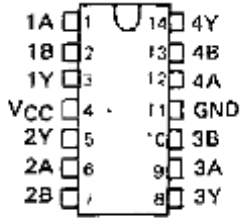
FUNCTION TABLE (each gate)

INPUTS		OUTPUT
A	B	Y
H	X	L
X	H	L
L	L	H

SN5402 . . . J PACKAGE
SN54LS02, SN54S02 . . . J OR W PACKAGE
SN7402 . . . N PACKAGE
SN74LS02, SN74S02 . . . D OR N PACKAGE
(TOP VIEW)



SN5402 . . . W PACKAGE
(TOP VIEW)



Inverter Diagram

SN5404, SN54LS04, SN54S04,
SN7404, SN74LS04, SN74S04
HEX INVERTERS

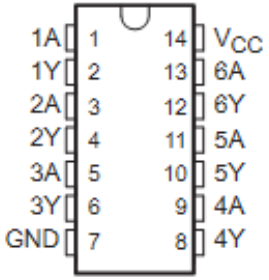
SDLS029C – DECEMBER 1983 – REVISED JANUARY 2004

- **Dependable Texas Instruments Quality and Reliability**

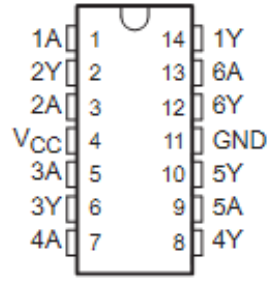
description/ordering information

These devices contain six independent inverters.

SN5404 . . . J PACKAGE
SN54LS04, SN54S04 . . . J OR W PACKAGE
SN7404, SN74S04 . . . D, N, OR NS PACKAGE
SN74LS04 . . . D, DB, N, OR NS PACKAGE
(TOP VIEW)



SN5404 . . . W PACKAGE
(TOP VIEW)



SN54LS04, SN54S04 . . . FK PACKAGE
(TOP VIEW)

AND Gate

SN5408, SN54LS08, SN54S08
SN7408, SN74LS08, SN74S08
QUADRUPLE 2-INPUT POSITIVE-AND GATES

SDLS033 – DECEMBER 1983 – REVISED MARCH 1988

- **Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

description

These devices contain four independent 2-input AND gates.

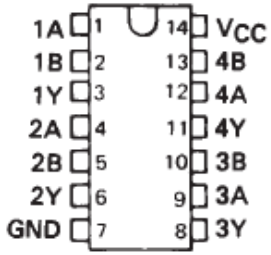
The SN5408, SN54LS08, and SN54S08 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN7408, SN74LS08 and SN74S08 are characterized for operation from 0° to 70°C.

FUNCTION TABLE (each gate)

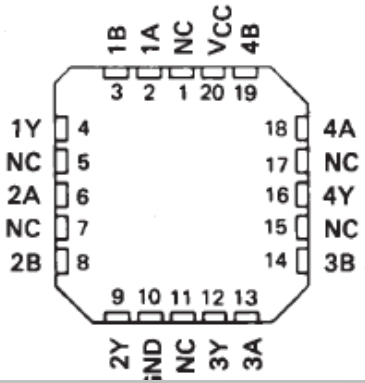
INPUTS		OUTPUT
A	B	Y
H	H	H
L	X	L
X	L	L

logic symbol†

SN5408, SN54LS08, SN54S08 . . . J OR W PACKAGE
SN7408 . . . J OR N PACKAGE
SN74LS08, SN74S08 . . . D, J OR N PACKAGE
(TOP VIEW)



SN54LS08, SN54S08 . . . FK PACKAGE
(TOP VIEW)



OR Gate

SDLS100

SN5432, SN54LS32, SN54S32, SN7432, SN74LS32, SN74S32
QUADRUPLE 2-INPUT POSITIVE-OR GATES

DECEMBER 1983 – REVISED MARCH 1988

- **Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

description

These devices contain four independent 2-input OR gates.

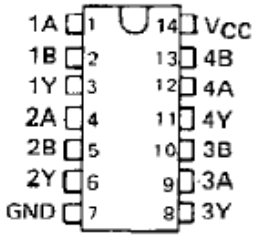
The SN5432, SN54LS32 and SN54S32 are characterized for operation over the full military range of −55°C to 125°C. The SN7432, SN74LS32 and SN74S32 are characterized for operation from 0°C to 70°C.

FUNCTION TABLE (each gate)

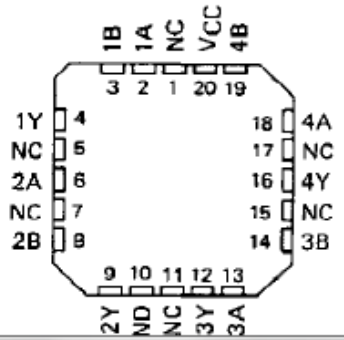
INPUTS		OUTPUT
A	B	Y
H	X	H
X	H	H
L	L	L

logic symbol†

SN5432, SN54LS32, SN54S32 . . . J OR W PACKAGE
SN7432 . . . N PACKAGE
SN74LS32, SN74S32 . . . D OR N PACKAGE
(TOP VIEW)



SN54LS32, SN54S32 . . . FK PACKAGE
(TOP VIEW)



Triple 3-Input AND Gate

SN54LS11, SN54S11,
SN74LS11, SN74S11
TRIPLE 3-INPUT POSITIVE-AND GATES

SDLS131 – APRIL 1985 – REVISED MARCH 1988

- **Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

description

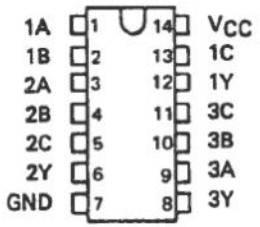
These devices contain three independent 3-input AND gates.

The SN54LS11 and SN54S11 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN74LS11 and SN74S11 are characterized for operation from 0°C to 70°C.

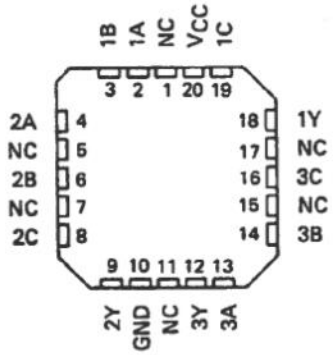
FUNCTION TABLE (each gate)

INPUTS			OUTPUT
A	B	C	Y
H	H	H	H
L	X	X	L
X	L	X	L
X	X	L	L

SN54LS11, SN74S11 . . . J OR W PACKAGE
SN74LS11, SN74S11 . . . D OR N PACKAGE
(TOP VIEW)

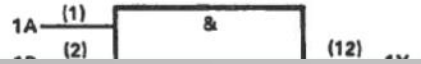


SN54LS11, SN54S11 . . . FK PACKAGE
(TOP VIEW)



NC—No internal connection

logic symbol†



logic diagram (positive logic)

Dual J-K Flip-Flops w/Clear

SN5473, SN54LS73A, SN7473, SN74LS73A
DUAL J-K FLIP-FLOPS WITH CLEAR

SDLS118 – DECEMBER 1983 – REVISED MARCH 1988

- **Package Options Include Plastic "Small Outline" Packages, Flat Packages, and Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

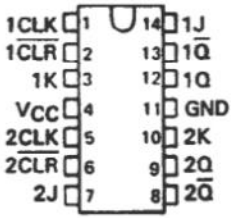
description

The '73, and 'H73, contain two independent J-K flip-flops with individual J-K, clock, and direct clear inputs. The '73, and 'H73, are positive pulse-triggered flip-flops. J-K input is loaded into the master while the clock is high and transferred to the slave on the high-to-low transition. For these devices the J and K inputs must be stable while the clock is high.

The 'LS73A contains two independent negative-edge-triggered flip-flops. The J and K inputs must be stable one setup time prior to the high-to-low clock transition for predictable operation. When the clear is low, it overrides the clock and data inputs forcing the Q output low and the $\overline{Q}$ output high.

The SN5473, SN54H73, and the SN54LS73A are characterized for operation over the full military temperature range of −55°C to 125°C. The SN7473, and the SN74LS73A are characterized for operation from 0°C to 70°C.

SN5473, SN54LS73A . . . J OR W PACKAGE
SN7473 . . . N PACKAGE
SN74LS73A . . . D OR N PACKAGE
(TOP VIEW)



'73
FUNCTION TABLE

INPUTS				OUTPUTS	
$\overline{CLR}$	CLK	J	K	Q	$\overline{Q}$
L	X	X	X	L	H
H	⎍	L	L	Q_0	$\overline{Q}_0$
H	⎍	H	L	H	L
H	⎍	L	H	L	H
H	⎍	H	H	TOGGLE	

'LS73A
FUNCTION TABLE

INPUTS				OUTPUTS	
$\overline{CLR}$	CLK	J	K	Q	$\overline{Q}$
L	X	X	X	L	H
H	↓	L	L	Q_0	$\overline{Q}_0$
H	↓	H	L	H	L
H	↓	L	H	L	H