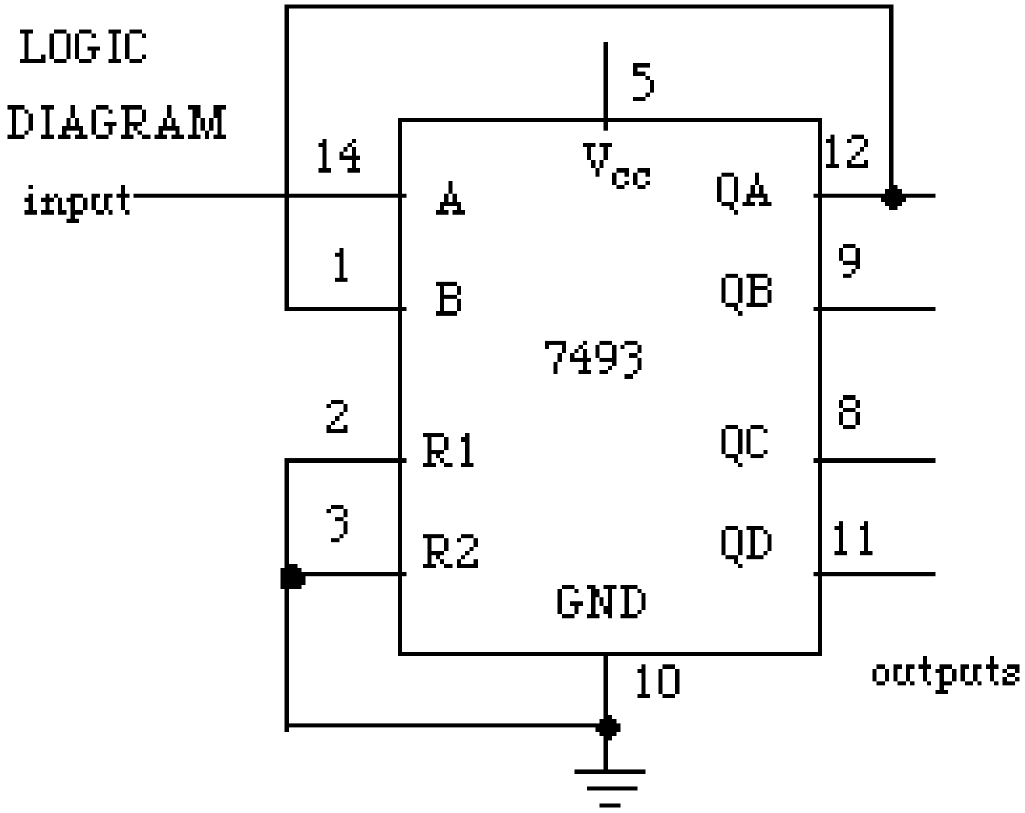
CS 240 Lab 3
Bits

- **Logic Diagrams**
- **Binary and Hexadecimal Numbers**
- **Review of Two’s Complement and Overflow**
- **Bitbucket/Mercurial complete Tutorial**
- **Finish Bit Puzzle Exercises from Lab 2**
- **Start work on Bits assignment**

Logic Diagrams

Not the same as pin-outs! Show information about the logical operation of the device.

Inputs on left side of diagram
Outputs on right
Voltage shown on top
Ground shown on bottom



Binary and Hexadecimal Numbers

Hex	Binary			
	QD	QC	QB	QA
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

Binary can be converted to decimal using positional representation of powers of 2:

$$0111_2 = 0 \text{ x } 2^3 + 1 \text{ x } 2^2 + 1 \text{ x } 2^1 + 1 \text{ x } 2^0, \quad \text{result} = 7_{10}$$

Decimal can be also be converted to binary by finding the largest power of 2 which fits, subtract, and repeat with the remainders until remainder is 0 (assigning 1 to the positions where a power of 2 is used):

$$6_{10} = 6 - 2^2 = 2 - 2^1 = 0, \quad \text{result} = 0110_2$$

Hex can be converted to binary and vice versa by grouping into 4 bits.

$11110101_2 = F5_{16}$ $\qquad 37_{16} = 00110111_2$

Two’s Complement and Overflow

Given n bits, the range of binary values which can be represented using

Unsigned representation: 0 –> $2^n - 1$

Signed representation: $- 2^{n-1}$ -> $2^{n-1} - 1$, MSB is used for sign

Two’s Complement (signed representation):

Most significant /leftmost bit (0/positive, 1/negative)

Example: given a fixed number of 4 bits:
1000_2 is negative.
0111_2 is positive.

Overflow

Given a fixed number of n available bits:
Overflow occurs if a value cannot fit in n bits.

Example: given 4 bits:
The largest negative value we can represent is -8_{10} (1000_2)
The largest positive value we can represent is $+7_{10}$ (0111_2)

Overflow in Addition

When adding two numbers with the same sign which each can be represented with n bits, the result may cause an overflow (not fit in n bits).

An overflow occurs when adding if:

- Two positive numbers added together yield a negative result, or
- Two negative numbers added together yield a positive result, or
- The Cin and Cout bits to the most significant pair of bits being added are not the same.

An overflow cannot result if a positive and negative number are added.

Example: given 4 bits:

$$\begin{array}{r}0111_2\\ \underline{+\ 0001_2}\\ 1000_2\end{array}\quad \text{overflow}\qquad \text{NOTE: there is not a carry-out!}$$

In two's complement representation, a carry-out does not indicate an overflow, as it does in unsigned representation.

Example: given 4 bits,

$$\begin{array}{r}1001_2\ (-7_{10})\\ \underline{+\ 1111_2\ (-1_{10})}\\ 1\ 1000_2\ (-8_{10})\end{array}\quad \text{no overflow, even though there is a carry-out}$$