Unit 14

Noise Margins, Interfacing, and Tri-States
Signal Types

- Recall even digital signals are *just voltages*...
- Analog signal
  - Continuous time signal where each voltage level has a unique meaning
- Digital signal
  - Continuous signal where voltage levels are mapped into 2 ranges meaning 0 or 1
Signals and Meaning

Each voltage value has unique meaning.

Analog

0.0 V

5.0 V

Each voltage maps to ‘0’ or ‘1’
(There is a small illegal range where meaning is undefined since threshold can vary based on temperature, small variations in manufacturing, etc.)

Digital

0.0 V

0.8 V

2.0 V

5.0 V

Logic 0

Logic 1

Illegal

Threshold Range
NOISE MARGINS, LEVEL SHIFTERS, & DRIVE STRENGTH
A Motivating Example

Example 1

- You connect an output port to an LED (light emitting diode) and connect everything correctly. The light should turn on when you set your output bit to a high voltage (logic '1').
- When you turn the system on the LED does not glow. You measure the voltage at the gate output with a voltmeter and find it is not 5V but 1.8V? Why isn't it a logic 1?
- The maximum current output ability from the output port is not high enough to adequately supply the LED which then drags the voltage down.

Example 2

- You buy two digital chips (say a microprocessor and GPS reader)
- You correctly wire them together and write software to turn 'on' a pin on the microprocessor to a '1' to enable the GPS reader
- When the software runs the GPS unit does not turn on. Why?
- Different circuit implementation techniques use different voltage levels to indicate '1' or '0' and may be incompatible

Lesson To Be Learned: Not all 1's or 0's are created equal!
The Digital Abstraction

- Digital is a nice abstraction of voltage and current
  - Lets us just think 'on' or 'off' but not really worry about the voltages and currents underneath

- Until NOW!!!

- Not all 1's and 0's are created equal
  - A '1' can be any 'HIGH' voltage (maybe in the range 2V-5V)
  - A '0' can be any 'LOW' voltage (maybe in the range 0V-0.8V)
  - So 3V and 5V both mean '1' but they aren't equal

- Similarly certain outputs of a chip may connect to other devices that require more current than the output can produce
  - Think of connecting a fire hose to your garden spigot
  - Or even worse your garden hose to a fire hydrant...it would shred it
  - In the same way, inputs and outputs of different devices must be matched to the demands/requirement of what they connect to
Digital Voltage Noise Margins

- Consider the **output** of one digital circuit feeding the **input** of another
  - Assume the devices are from different vendors (families of devices)
- There may be different limits and requirements of the two devices
  - Example: The output may produce 3V to mean logic '1' while the next device's **input** requires 5V to be used as logic '1'
- Analogy 1: Grades. Suppose the **cutoff for an A is 90%** (i.e. required input)
  - If you get a 91% (i.e. output result)... **GOOD**!
  - If you get an 89%... **BAD** from the cutoff's perspective.
- Analogy 2: Tickets. Suppose there are **100 available tickets to an event** (i.e. input limit)
  - If you are the 99th person (i.e. output result)... **GOOD**!
  - If you are the 101st person... **BAD**!
Digital Voltage Noise Margins

- Consider one digital gate feeding another

Output Range Interpretation

Input Range Interpretation

As long as $V_{OH} > V_{IH}$ and $V_{OL} < V_{IL}$ we are in good shape...

Electromagnetic interference & power spikes can cause this to break down
Class Activity

• Do an internet search for "74LS00 datasheet" (this is a chip w/ some 2-input NAND gates) and try to find any PDF and open it

• Skim the PDF and try to find:
  – VOH, VIH, VOL, VIL
Analogy

• Consider a sprinkler system...what will happen if you add 100 new sprinklers to your backyard?
• Pressure (voltage) will go way down and reduce water (current) flow coming out of each
Current Limitations

• When a circuit outputs a 'HIGH' ('1') it can only supply (source) so much current (think of your garden hose spigot) = $I_{\text{OH}}$

• When a circuit outputs a 'LOW' ('0') it can only suck up (sink) so much current = $I_{\text{OL}}$

• When a circuit receives a 'HIGH' signal on the input side it may need a certain amount of current to recognize the input as 'HIGH' = $I_{\text{IH}}$

• When a circuit receives a 'LOW' signal on the input side it may need a certain amount of current to recognize the input as 'LOW' = $I_{\text{IL}}$
Example

- Consider the example where device A's output connects to device B's input
  - Are the voltage requirements compatible?
  - How many device B inputs can a single device A output drive?

- Always use worst case of high or low output drive capability

<table>
<thead>
<tr>
<th>Dev.</th>
<th>VOH</th>
<th>VIH</th>
<th>VOL</th>
<th>VIL</th>
<th>IOH</th>
<th>IIH</th>
<th>IOL</th>
<th>IIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3.4V</td>
<td>3.3V</td>
<td>0.5V</td>
<td>1.0V</td>
<td>-4 mA</td>
<td>-1 mA</td>
<td>10 mA</td>
<td>2 mA</td>
</tr>
<tr>
<td>B</td>
<td>3.2V</td>
<td>3.0V</td>
<td>0.6V</td>
<td>0.7V</td>
<td>-2 mA</td>
<td>-1 mA</td>
<td>6 mA</td>
<td>2 mA</td>
</tr>
</tbody>
</table>

Voltage requirement are compatible! Dev. A VOH > Dev. B VIH AND Dev. A VOL < Dev. B. VIL

Dev. A's output can drive 4 Dev. B inputs
When outputting '1':
- \((\text{Dev. A IOH} / \text{Dev. B IIH}) = (-4 / -1) = 4\)
When outputting '0':
- \((\text{Dev. A IOL} / \text{Dev. B IIL}) = (10 / 2) = 5\)
Drive capability = min(4, 5) = 4
Consideration

- If we attach too many gates to one output it may not be enough to drive those gates.
- Need to make sure the current requirements and capabilities match.
- Let's say we connect one of the NAND gates on the 74LS00 chip to an input of N other NAND gates...
- Can it produce/suck up the required current...
- ...if $N = 6$?
- ...if $N = 12$?

If $I_{OH}$ or $I_{OL}$ is too low we can split the loads by placing intermediate buffers.
All In the Family

- There are many families of circuit devices that talk different language (Each has a different VOH, VIH, VOL, VIL, IOL, IIL, etc.)
- Examples:
  - CMOS
  - TTL
  - ECL
- Must make sure if you interface two different devices that they are compatible (i.e. VOH of device A is greater than VIH of device B) or use a buffer/amplifier/level shifter circuit to help them talk to each other
Arduino Limits

• Arduino outputs can sink (suck up) and source (produce) around a maximum of 20 mA on a pin

• Do an internet search for "Standard Servo Motor Datasheet" and find the maximum current it may need

• It doesn't seem like the Arduino would be able to drive the servo motor. How is it working?
  – Remember the 3-pin interface: R = Power, B = Ground, W = Signal
  – The signal is separate from the power
  – The power source is used to amplify the signal
Another Example

• Now consider a speaker system where the power and signal are provide together
  – Given our Arduino use 5V = Vcc and its current limitations per pin, how much power can we supply to the speaker?
  – 5V * 20 mA = 0.1W
  – You need an amplifier...

Power & Signal together
TRI-STATE GATES
Typical Logic Gate

- Gates can output two values: 0 & 1
  - Logic ‘1’ (Vdd = 3V or 5V), or Logic ‘0’ (Vss = GND)
  - But they are ALWAYS outputting something!!!
- Analogy: a sink faucet
  - 2 possibilities: Hot (‘1’) or Cold (‘0’)
- In a real circuit, inputs cause *EITHER* a pathway from output to VDD *OR* VSS

![Diagram of logic gates](image)
Output Connections

- Can we connect the output of two logic gates together?
- No! Possible short circuit (static, low-resistance pathway from Vdd to GND)
- We call this situation “bus contention”
Tri-State Buffers

• Normal digital gates can output two values: 0 & 1
  1. Logic 0 = 0 volts
  2. Logic 1 = 5 volts

• Tristate buffers can output a third value:
  3. Z = High-Impedance = "Floating"
     (no connection to any voltage source...infinite resistance)

• Analogy: a sink faucet
  3 possibilities:
  1.) Hot water,
  2.) Cold water,
  3.) NO water

- Hot Water = Logic 1
- Cold Water = Logic 0
- NO Water = Z (High-Impedance)
Tri-State Buffers

- Tri-state buffers have an extra enable input
- When disabled, output is said to be at high impedance (a.k.a. Z)
  - High Impedance is equivalent to no connection (i.e. floating output) or an infinite resistance
  - It's like a brick wall between the output and any connection to source
- When enabled, normal buffer

<table>
<thead>
<tr>
<th>En</th>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Z</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Tri-State Buffers

- We use tri-state buffers to share one output amongst several sources
- Rule: Only 1 buffer enabled at a time
Tri-State Buffers

- We use tri-state buffers to share one output amongst several sources.
- Rule: Only 1 buffer enabled at a time.
- When 1 buffer enabled, its output overpowers the Z’s (no connection) from the other gates.

Select source 1 to pass its data.

Output of 0 overpowers the Z.

Disabled buffers output ‘Z’.
Enable Polarity

- **Side note**: Some tri-states are designed to pass the input (be enabled) when the enable is 0 (rather than 1)
  - A inversion bubble is shown at the enable input to indicate the "low" polarity needed to enable the tristate

<table>
<thead>
<tr>
<th>En</th>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Z</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>En</th>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Z</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Communication Connections

- Multiple entities need to communicate
- We could use
  - Point-to-point connections
  - A shared bus (set of wires)
14.26

Bidirectional Bus

• 1 transmitter (otherwise bus contention)
• N receivers
• Each device can send (though 1 at a time) or receive
**Tri-State Gates**

- Big advantage: don’t have to know in advance how many devices will be connected together
  - Tri-State gates give us the option of connecting together the outputs of many devices without requiring a circuit to multiplex many signals into one
- Just have to make sure only one is enabled (output active) at any one time.
Problem: How can you use the serial I/O lines of the Arduino, which are also used for programming it?

Two active devices, both trying to output a signal, collide here.
Solution: Use a Tri-State gate to isolate the transmitter's data from the µC until programming is over.

Output of gate is floating until µC program makes Pxx a zero.