Practical PCB Design and Manufacture

Lab-9 Report: Trace to trace crosstalk

Objective / Purpose of the Lab:

- This lab exercise employs a variety of experiments and observations to examine the impact of cross talk between aggressor and victim signals in electrical circuits. In addition to measuring the effects of various wiring configurations and learning to identify signal and return paths, participants will investigate various geometrical setups to comprehend how simultaneous signal switching in aggressor lines induces noise in adjacent victim lines. The objectives include learning how to measure switching noise cross talk, assessing the effects of simultaneous I/O switching, figuring out the routing architecture that minimizes cross talk, and assessing noise scaling and cross talk variance with changes in physical wiring.

Component Listing:

- Arduino UNO (For power supply)
- Special crosstalk board
- 10x Probes with spring ground tips
- Oscilloscope

Board Design:

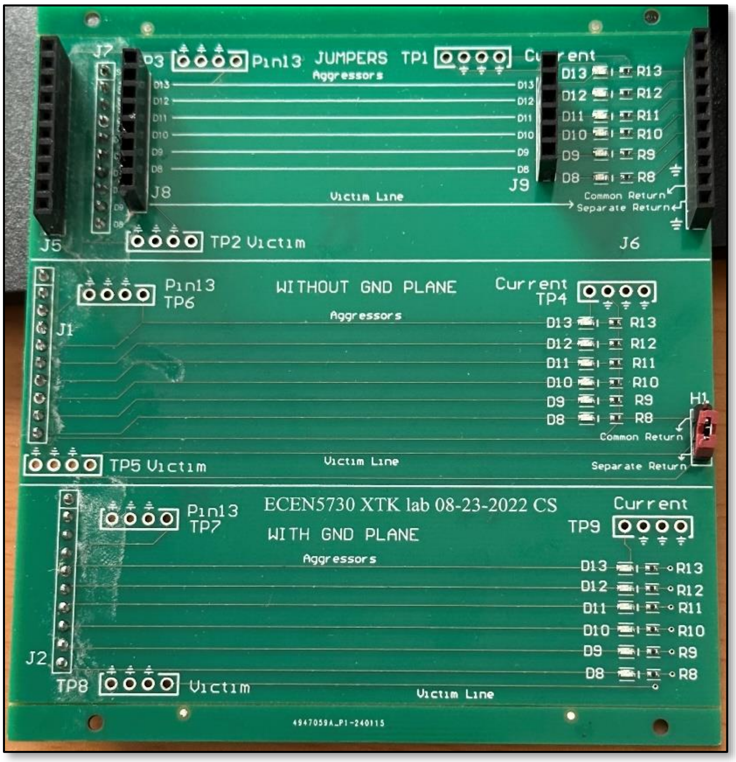

Fig – 1: Board layout

- This is a special crosstalk board that combines three different PCB layouts.
- The board's middle section has a shorting flag to switch between a shared path and a separate path, while the bottom section of the board has a ground plane.
- The header allows for the simultaneous switching of outputs through connecting to pins 8–13 on an Arduino Uno.
- All the pins, numbered 8 through 13, are turned on and off via the Arduino in time with each clock cycle.

Practical PCB Design and Manufacture

Lab-9 Report: Trace to trace crosstalk

Scope Shots with Analysis:

Part 1: Cross talk with a continuous return plane

Fig – 2: Noise for all LEDs connected

- The Fig 2 shows the difference when all the pins of the Arduino i.e. from 8 to 13 are toggling simultaneously.
- The yellow trace is the one when measured at pin 13 which is farthest from the victim line and closer to the aggressor line.
- The green trace is the noise when all the pins of the Arduino are toggling simultaneously.
- Here, we can see the noise is quite less about almost 321.6 mV.

The dI/dt in the aggressor loop is what causes the cross talk. When the signal transitions voltage levels, there are significant current variations at the voltage edges. We monitor the current across the 47 Ω resistor at the end of pin 13's trace. We can see the difference in noise in the Fig 3 and Fig 4 as shown below when the LED connected to the pin closest to the victim i.e. pin 8 (224.12 mV) is toggling and when the LED farthest i.e. pin 13 (80.4 mV) is toggling.

Fig – 3: Noise for led connected to pin 8

Fig – 4: Noise for led connected to pin 13

As we can see, the pin closest to the victim trace has the highest noise due to the stronger electromagnetic coupling between closely spaced conductors. This coupling includes:

- **Capacitive coupling:** where electric fields transfer noise, and
- **Inductive coupling:** where magnetic fields transfer noise.

The closer the aggressor and victim traces are, the stronger these interactions become, leading to higher noise levels on the nearest pin.

Practical PCB Design and Manufacture

Lab-9 Report: Trace to trace crosstalk

- The signal source will be an Arduino Uno. As aggressors, I am operating several digital input devices. The following is the sketch to simultaneously activate pins 12 and 13:
- As we can see from Fig 5 below, switching off LEDs from the one nearest to the victim trace to the farthest reduces noise in the victim trace gradually. This happens because both capacitive and inductive coupling—primary sources of electromagnetic interference—decrease as you eliminate nearby noise sources.
- The closest LEDs have the most significant impact on noise reduction, with each subsequent LED having a progressively lesser effect as their distance from the victim trace increases.
- This approach, effectively distancing or removing sources of EMI, is a practical strategy for mitigating noise in sensitive electronic circuits, especially in designs where signal integrity and electromagnetic compatibility are critical.

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
}

void loop()
{
  PORTB=B00111111;
  PORTB=B00000000;
}
```

Arduino Code

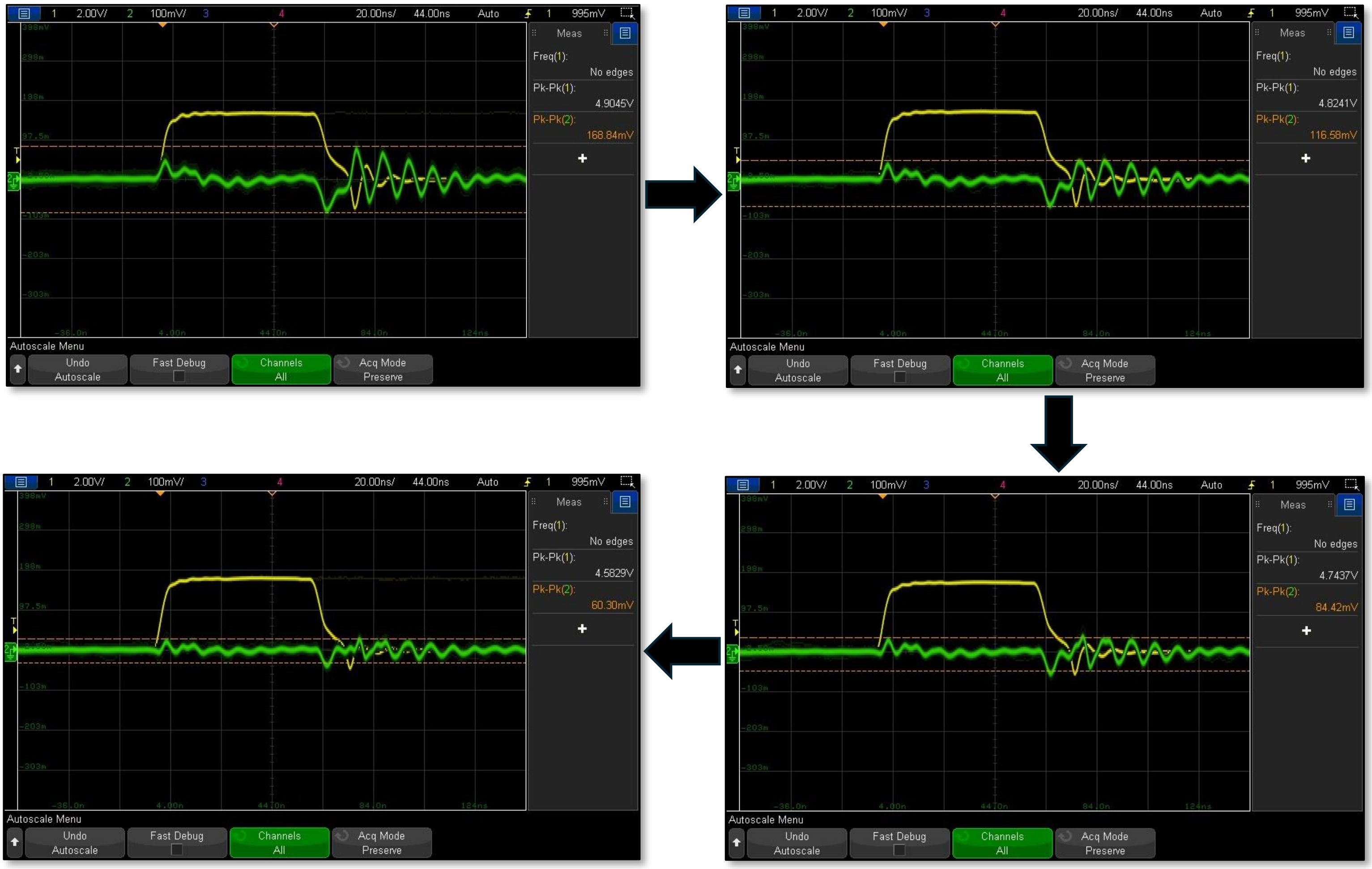

Fig – 5: Noise Reduction for LED's after turning them off from pin 8 to pin 13

Practical PCB Design and Manufacture

Lab-9 Report: Trace to trace crosstalk

Part 2: Cross talk with no plane, but an adjacent return trace

Common Return Path: Both the aggressor and victim traces use the same return path. This can reduce electromagnetic interference but introduces noise from one trace to another due to shared impedance.
Separate Return Paths: Each trace has its own return path. This helps to reduce noise coupling between the aggressor and victim but increases the system's overall susceptibility to external interference.
If minimizing interference between traces is crucial, separate returns is the best but if reducing overall electromagnetic emissions and susceptibility is more important, a common return path is better.
We can see in Fig 6 and Fig 7 below the difference in noise when we use a common return path and when a separate return path is used.

Fig – 6: Common return path

Fig – 7: Shared return path

Conclusion / Inference:

In this lab to measure the crosstalk between the return paths of the aggressor and victim, the key conclusions are:

- The closer the routing of the components, the crosstalk is minimized due to the self-inductance of the loop path.
- When using a common/shared return path, there is increased crosstalk between the aggressor and victim traces, as the shared return path can facilitate the transfer of noise from one trace to another.
- When using a separate return path, this approach can significantly reduce crosstalk between the aggressor and victim traces since the dedicated return path can isolate the signals better, preventing the transfer of noise from one circuit to another.

Practical PCB Design and Manufacture

Lab-9 Report: Trace to trace crosstalk