## Hardware-Oriented Security and Trust Project

## Trojan Embedding and Detection on FPGAs using Digilent VII-Pro boards

The members of the class will be partitioned into two groups and each group will serve two roles. One role as a **Red Team** and one role as a **Black Team**.

The basic idea is to choose a design, e.g., a VHDL description of an AES encryption algorithm, that both teams use. Each **black** team inserts Design-for-Hardware-Trust (DHT) features into the design and returns the modified design (as VHDL) to the **red** team. The **red** team synthesizes the design and creates two versions, one with three Trojans (probably using the FPGA edit feature in ISE) and one with no Trojans (Trojan-free). The **red** team generates the bitstreams for both designs and gives them to the **black** team. The **black** teams apply their Trojan detection strategies and attempt to identify which bitstream has the Trojans.

The **black** teams should analyze the designs to determine the DHT features they want to add, e.g., dummy FFs for enhanced controllability and observability. They add these functions to the VHDL so that they are visible to the **red** team.

The **red** team also analyzes the design to determine how to add the Trojans such that they defeat the encryption engine, e.g., make it trivial to break the encryption, leak out plain text through a side channel, or turn off encryption altogether. The changes MUST impact the security of the algorithm in some way and therefore they must be designed to produce a functional change to the algorithm under some specific conditions. At least one of the modifications must be 'trigger-based', i.e., the **red** team provides a special sequence of text, or a side-channel trigger (pushbutton combination), that activates the Trojan. Bear in mind that the functional output of both versions MUST be identical until the trigger condition is met otherwise the **black** team will trivially distinguish between them.

The **black** team's objective is to determine which bitstream contains the Trojans and either activates it (or each of them) directly or provides data that indicates an anomaly is present. A functional demonstration that the Trojan has been discovered involves showing that the logic behaviors of the two bitstreams are different for at least one input.

A successful demonstration that shows the presence of an anomaly involves showing that the delay along some paths is longer in one bitstream than in another. We will have multiple copies of the FPGAs available. For the anomaly demonstration, you need to characterize delays using each bitstream on multiple copies of the FPGAs and show that a statistical difference exists. Power analysis may also be used but may be more difficult to implement given the pin-out restrictions on the board.

Either team can use the pattern generator and logic analyzer that I have in my lab. We will develop a time sharing schedule if conflicts exist. I have recently discovered that it is possible to make power measurements on the XUP boards by inserting a ammeter in series across any of the triple jumpers in the 'power region' of the board (light green). We can also drive the chip with an external clock using the oscillator footprint connections labeled Y2 on the board, which should allow

you to carry out delay tests in which you strobe the clock until you get a failure in the capture flops.

As discussed in class, the **red** teams will submit their design to Colby and he will encrypt the bitstream to prevent the **black** teams from reverse engineering the bitstream to find the modifications that the **red** teams have made. The V2Pro has a hardware description engine on board.