IPWM: A Public-Key Watermarking Tool for IP Designs

Intellectual property (IP) block reuse is essential for facilitating the design process of System-on-a-Chip (SOC). Sharing IP blocks in such a competitive market poses significant high security risks. IPWM is an automatic tool for watermarking sequential IP designs. The implemented tool is based on the ***Coinciding Transitions*** approach, which relies on the usage of coinciding transitions to increase the watermark robustness as well as decreasing the overhead it might cause on the system.

A Prototype of the tool was implemented using C++ under UNIX environment. The design accepts kiss2 standard (as its FSM representation) which can be generated by many tools such as SIS. Figure 1 shows the different implementation blocks. IPWM is composed of four main blocks. It starts by building a tree for the FSM representation using the ***FSM builder*** block. The ***signature generation*** block provides the signature to the watermarker after hashing it, while the random input and next states needed are provided using a ***random generator*** built in our tool. The three components are added in the ***watermarker*** block; where the user can choose the algorithm to watermark his/her design. Finally, the produced watermarked design is converted again to kiss2 format using a ***Kiss-to-HDL*** block, and the key that provides the signature added is produced as well.

The watermark insertion algorithm attempts to coincide a part of the watermark on the state transition graph (STG) to increase the watermark robustness. This is done by searching different outputs of each visited state in the STG, and comparing it to a part of the generated signature in order to map this signature on the system outputs. Starting form any randomly chosen state (S_x), the watermark will be added to the STG according to the following algorithm (more details can be found in [1]):

1) Compare the outputs of the state S_x to the generated signature to check if they coincide.
2) In case one of the outputs is equal to the watermark bits, this transition will be considered part of our watermark.
3) If the signature sequence is not equal to any of the outputs, then the inputs of S_x will be checked to determine if there is any free input that can be used to add an extra transition. The next state in this case will be chosen randomly, with preference given to states with free transitions.
4) In case all the inputs are already being used, an extra input bit e_{iwm} is added to the system to extend the FSM. This input bit will have the same logic value for all existing transitions. For instance, a logic value '0' assigned to the already existing transitions and logic value '1' will be used for the watermark transition added. The next state will be chosen randomly.
5) The algorithm will loop until the embedding all the signature bits is done.

Reference:
[1] A. Abdel-Hamid, S. Tahar, and E.M. Aboulhamid: A Public-Key Watermarking Technique for IP Designs; *Proc. IEEE/ACM Design Automation and Test in Europe* ***(DATE'05)***, Munich, Germany, March 2005.