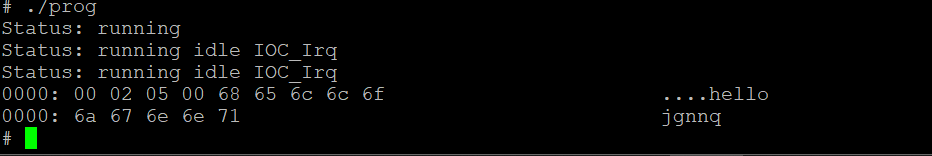
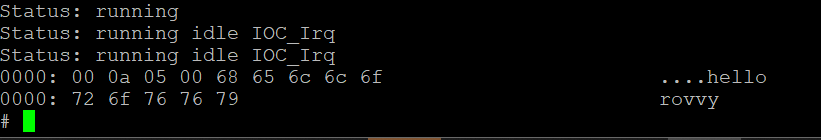
TP1

Quand nous avons booté, la carte nous a demandé un identifiant ainsi qu’un mot de passe. Nous avons saisi root pour les deux, et nous avons eu accès à un shell linux. Nous nous sommes déplacés dans l’explorateur de ficher avec les commades « ls » , « cd »… et nous avons exécuté le programme hello\_word avec la commande « ./hello\_word »

TP2

Une fois les actions effectuées, nous avons bien vu que nous enoyons « hello », et nous recevons « rovvy ». Après analyse de la trame envoyé, nous avons observé qu’en faisant varier le deuxièbe bit, le programme fonctionne toujours, cependant, le message reçu change : en remplaçant 10 par 2, le message reçu n’est plus « rovvy » mais « jgnnq ». On en déduit donc que l’harware accelerator fait chiffrement par un code de césar. Le code César est un chiffrement basé sur un décalage de l'alphabet (déplacement des lettres plus loin dans l'alphabet), il s'agit d'une [substitution monoalphabétique](https://www.dcode.fr/substitution-monoalphabetique), c'est-à-dire qu'une même lettre n'est remplacée que par une seule autre (toujours identique pour un même message).

TP3

Nous avons éprouvé des difficultés pour cette partie. Car lors des tests, nous n’arrivionn pas à booter le bon fpga.bit. Après de nombreux tests, nous avons remarqué que la corbeille de la carte (le dossier trash) contenait un fichier fpga.bit et que c’était celui la qui bootait. En le supprimant, nous avons réussi à résoudre nos problèmes.

Une fois le code finit et implémenté avec vivado HLS, nous l’avons intégré au FPGA.bit avec vivado et une fois installé sur la carte, nous avons en effet observé que notre code utilisant l’hardawre accelerator nous donnait bien un décallage d’un octet.

PROJET

Pour ce projet, nous avons décidé de faire un hardare accelerator qu’image processing. Il prend en entrée 2 images et renvoie en sortie une seule image qui est le mélange des deux entrées.

Sources :

<https://www.dcode.fr/chiffre-cesar>