

Sprint 7

issues terminées pour le sprint

Jean-Noël : gestion d'un socket en C [WIP]

Difficultés rencontrées

Faire marcher le socket sur Linux ET sur Windows simultanément (gestion différente par l'OS). Beaucoup de templates de code disponibles sur internet mais pas adapté à notre projet. Phase de conversion et d'adaptation qui a été nécessaires

Fonctionnalité

Permet d'envoyer un message d'un client et à un serveur en c via l'appel Scanf, et ce peut importe l'OS utilisé.

[TODO] => lier la fonction de socket au format du bitstream, l'envoi de données ainsi que l'envoi du dictionnaire.

Hugo : packaging d'une version d'EVEEX python possible

Difficultés rencontrées

Problème d'ouverture du package, il manquait des dépendances lors de l'ouverture des fichiers. Le problème a été résolu en changement la méthode d'importation des fichier (import EVEEX.huffman au lieu de import Huffman)

Fonctionnalité

Permet la création d'un package d'une version de l'algorithme sur python. Ce package permet l'importation facile sur une machine tierce des fonctions nécessaires à l'algo, et intègre un fichier requirements.txt permettant d'installer via pip tout les paquets externes nécessaires.

Hugo : ajout de la compression vidéo sur l'algorithme python

difficultés rencontrées

Problème de format lorsque la taille de l'image n'est pas un multiple de la taille des macro blocs. La solution a été de changer de package d'ouverture de la vidéo.

fonctionnalité

Permet la compression EVEEX d'une flux vidéo (sans son), IE de plusieurs images qui se suivent. Les performances ne sont pas temps réelles, car python ne permet pas d'aller suffisamment vite.

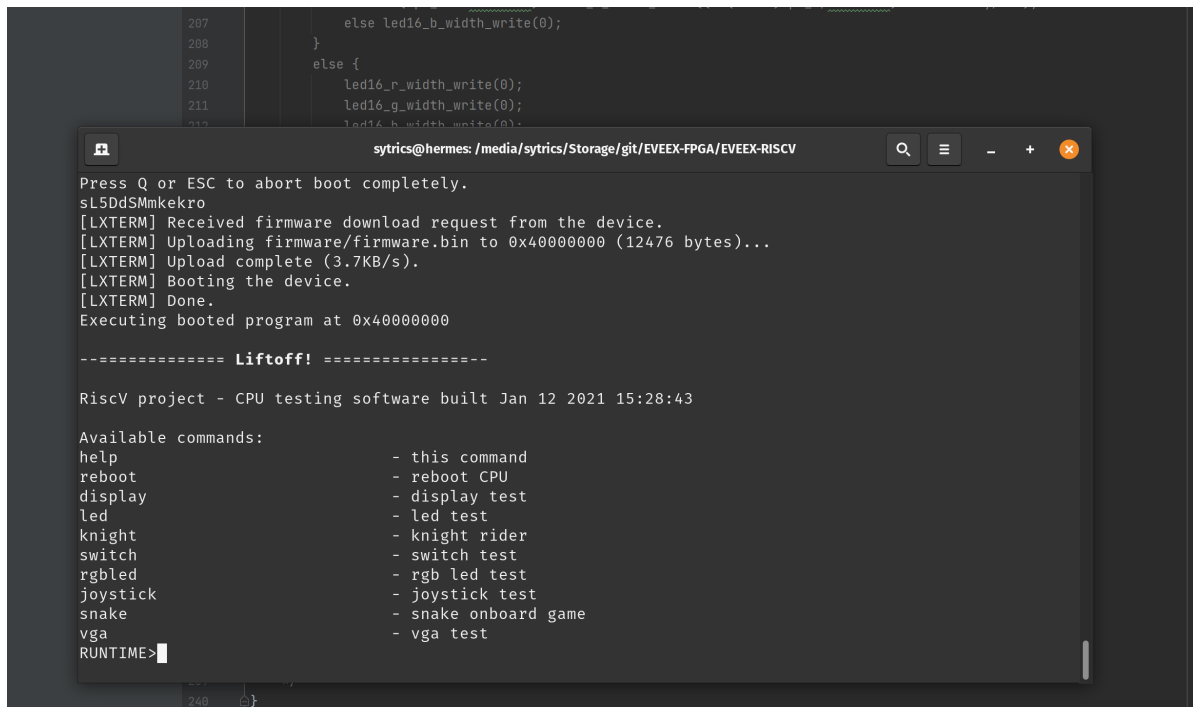
Guillaume : comprendre, utiliser et adapter le projet ENSTAB-RISCV et l'implémentation d'un SOC au sein d'un fpga

Difficultés rencontrés

Gestion de la RAM difficilement compréhensible, avec un manque de connaissance sur les différentes technologies RAM (Sram, dram) et leur implémentation hardware au sein du FPGA. La compréhension du makefile ainsi que du linker a pris quelques heures également.

Fonctionnalité

Permet d'intégrer au sein d'un fpga un SOC riscv, ainsi qu'un firmware disposant de plusieurs fonctions de test du matériel. le SOC est contrôlable depuis un terminal Linux et dispose d'assez de RAM pour exécuter un code simple (16ko).



```
207         else led16_b_width_write(0);
208     }
209     else {
210         led16_r_width_write(0);
211         led16_g_width_write(0);
212         led16_b_width_write(0);
213     }
}

sytrics@hermes: /media/sytrics/Storage/git/EVEEX-FPGA/EVEEX-RISCV
Press Q or ESC to abort boot completely.
sL5DdSMmkekro
[LXTERM] Received firmware download request from the device.
[LXTERM] Uploading firmware/firmware.bin to 0x40000000 (12476 bytes)...
[LXTERM] Upload complete (3.7KB/s).
[LXTERM] Booting the device.
[LXTERM] Done.
Executing booted program at 0x40000000

----- Liftoff! -----

RiscV project - CPU testing software built Jan 12 2021 15:28:43

Available commands:
help                - this command
reboot              - reboot CPU
display             - display test
led                 - led test
knight              - knight rider
switch              - switch test
rgbled              - rgb led test
joystick            - joystick test
snake               - snake onboard game
vga                 - vga test
RUNTIME>
```

Hussein : Implémentation de la DCT / DCT inverse en C

Difficultés rencontrés

Application de la DCT sur une image, et non sur une matrice. Quelques problèmes pour la prise en compte de la taille des matrices (qui pourra être potentiellement variable dans le futur).

Fonctionnalité

Permet la transformation DCT d'une matrice de taille N, ainsi que la DCT inverse. Elle intervient dans le chaîne de compression.

```
C:\Users\saadhu\CLionProjects\encoder-decoder\cmake-build-debug\encoder_decoder.exe
```

```
Origin Matrix :
```

```
244 243 118 154 126 246 137 184
178 127 196 155 184 184 147 245
114 116 134 231 145 166 214 112
178 127 196 155 184 184 247 245
200 224 133 145 105 167 237 118
137 263 118 154 126 246 137 89
178 127 196 255 184 184 147 245
114 116 134 231 145 166 247 198
```

```
Coefficients Matrix :
```

```
1385.3 -62.7 22.3 19.7 -20.8 50.7 -7.6 -82.0
 5.5 29.0 47.8 38.4 4.2 -36.1 42.0 15.2
22.2 -15.4 -7.4 -21.1 69.1 -21.6 44.7 -43.9
 6.5 129.7 61.6 83.7 -65.1 -101.9 9.4 -83.9
26.7 -14.0 86.5 27.1 -28.3 26.5 -36.9 15.5
53.4 -11.7 -12.6 -14.6 48.8 -2.8 39.7 -6.5
-73.1 33.9 -42.3 67.1 -86.5 7.2 -79.1 -79.2
-24.7 36.1 -11.0 3.8 6.7 61.6 -45.1 -24.4
```

```
Reconstructed Matrix :
```

```
244 243 118 154 126 246 137 184
178 127 196 155 184 184 147 245
114 116 134 231 145 166 214 112
178 127 196 155 184 184 247 245
200 224 133 145 105 167 237 118
137 263 118 154 126 246 137 89
178 127 196 255 184 184 147 245
114 116 134 231 145 166 247 198
```

```
Difference Origin/Reconstructed:
```

```
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
```

```
Process finished with exit code 0
```

[TODO] => gérer la transformation DCT / inverse sur une image/macro bloc

Alexandre : Encodage de Huffman et création de types non natifs en C

difficultés rencontrées

Grosses fuites de mémoires, une journée complète pour tracker toutes les fuites par valgrind, ainsi que les corruptions.

Maintenant les types non natifs sont implémentés et ne génère pas de fuite/corruption de mémoire.

fonctionnalité

Encoder une chaîne de caractère selon la méthode de l'arbre binaire de Huffman, et retourner un dictionnaire d'encodage avec pour chaque symbole son encodage binaire.

```
Printing huffman tree :
      |
      +-----+-----+
      |               |
      +-----+-----+       +-----+-----+
      |               |       |               |
      +---+---+   +---+---+   +---+---+   +---+---+
      (r)   (a)   (s)   (t)       |       (e)   ( )   (.)
                                   +---+---+
                                   (p)   (h)

Encoding dictionary :
000 --> r
001 --> a
010 --> s
011 --> t
1000 --> p
1001 --> h
101 --> e
110 -->
111 --> .
```

Suite de l'activité pour le sprint 8 (pour l'instant)

- finir le socket en C (l'incorporer au projet)
- support des images par la transformation DCT
- Changement de SOC pour passer sur un fonctionnement VeriscV + linux basé buildroot
- encodage d'un dictionnaire de Huffman basé sur plusieurs frames (moyenne) en python.

A plus long terme (visée sprint 9) :

- gestion de l'interpolation d'image en python
- Faire la chaîne d'encodage en entier en C
- préparer un SOC avec le support d'une camera I2C et de mémoire DDRAM
- découpage en macrobloqs dynamiques

