Documentation Technique

1) Fichier opcode.py

a) Variables

opcodeMasque

Type : tableau (int)
Taille : 35
Utilisation : Contient les différents masque qui seront utilisé sur l'opcode pour retrouver son id correspondant

opcodeId

Type : tableau (int)
Taille : 35
Utilisation : Contient les 35 id des opcodes utilisés en CHIP-8 qui vont permettre de connaître l'action à effectuer

opcode

Type : int
Taille : maximum 4 octets (65535)
Utilisation : C'est la concaténation de 2 valeurs (2 octet maximum chacune) contenus dans la mémoire, on obtient ainsi l'action à effectuer et ses valeurs.

action

Type : int
Taille : maximum 34
Utilisation : Sa valeur correspond à l'action à effectuer, elle est déterminée arbitrairement et chaque valeur correspond à l'opcodeId associé. Par exemple pour action à 4 on a l'opcodeId 0x2000 qui correspond à un saut.

b) Fonctions

analyse

Paramètres : Aucun
Variables retournées : action, opcode

Utilisation : Premièrement la fonction va créer l'opcode à partir de la valeur du Pointer Counter (PC), on prend donc la valeur en mémoire à l'adresse stocké dans PC puis l'on concatène la valeur en mémoire à l'adresse PC+1, ainsi on crée la variable opcode.
Ensuite on va chercher l'action correspondante à l'opcode obtenu précédemment en faisant un ET logique entre l'opcode et un masque, si l'on obtient l'opcodeId correspond on a notre action à effectuer.

interpretation

Paramètres : Aucun
Variables retournées : action, opcode
Utilisation : Premièrement on va récupérer l'action et l'opcode grâce à la fonction "analyse" puis on va effectuer une des 35 instructions selon la valeur de la variable "action"

valueNNN

Paramètres : opcode
Variables retournées : une valeur de maximum 3 octets (4095)
Utilisation : La fonction va retourner la valeur en décimale des 3 derniers octets de l'opcode

valueNN

Paramètres : opcode
Variables retournées : une valeur de maximum 2 octets (255)
Utilisation : La fonction va retourner la valeur en décimale des 2 derniers octets de l'opcode

valueN

Paramètres : opcode
Variables retournées : une valeur de maximum un octet (15)
Utilisation : La fonction va retourner la valeur en décimale du dernier octet de l'opcode

valueX

Paramètres : opcode
Variables retournées : une valeur de maximum un octet (15)
Utilisation : La fonction va retourner la valeur en décimale du premier octet de l'opcode

valueY

Paramètres : opcode
Variables retournées : une valeur de maximum un octet (15)
Utilisation : La fonction va retourner la valeur en décimale du deuxième octet de l'opcode

2)Fichier memoire.py

a) Variables

memoire

Type : tableau (int)
Taille : 4096
Utilisation : Tableau qui représente la mémoire entière, ainsi il y a 4096 cases qui peuvent contenir maximum 2 octets (255). Par exemple le programme en lui même est stocké à partir de la 500ème case.

V

Type : tableau (int)
Taille : 16
Utilisation : Contient la valeur des 16 registres V (nommé de 0 à F), chacun contenant maximum 2 octets (255)

I

Type : int
Taille : maximum 4 octets (65535)
Utilisation : Registre qui contient une adresse mémoire

DT

Type : int
Taille : 255
Utilisation : Le Delay Timer va permettre de “compter” le temps étant donné qu’il décrémente toute les 16ms

ST

Type : int
Taille : 255
Utilisation : Le Sound Timer va permettre de faire un son tant qu’il n’est pas égal à 0, il décrémente toute les 16ms

PC

Type : int
Taille : maximum 4 octets (65535)
Utilisation : Le Pointer Counter permet de savoir à quel endroit de la mémoire nous sommes rendu dans le programme

SP

Type : int
Taille : 15
Utilisation : Le Stack Pointer est utilisé pour savoir à quel endroit du tas nous sommes

saut

Type : liste (int)
Taille : 16
Utilisation : Le Tableau contient les adresses mémoires pour retournés d'un saut

tabEcran

Type : tableau (int)
Taille : 64*32
Utilisation : Tableau contenant 32 listes (les lignes) qui ont chacune 64 valeurs qui peuvent être 0 (pixel éteint) ou 1 (pixel allumé)

tabTouche

Type : liste (int)
Taille : 16
Utilisation : Liste contenant la valeur des touches, 0 si elle est relâchée ou 1 si elle est enfoncée

b) Procédure

chargerCaractere

Paramètres : aucun
Utilisation : Procédure qui va charger en mémoire les sprites des chiffres de 0 à 9 et des lettres de A à F sous forme de binaire

3)Fichier lecture.py

a) Procédures

lecture

Paramètres : le nom du fichier
Utilisation : On vérifie que le fichier CHIP8 existe, si oui on l'ouvre en lecture. On stocke ensuite le programme en mémoire à partir de l'adresse 512

sauvegarde

Paramètres : le nom du fichier, une variable pour forcer l'écriture
Utilisation : On vérifie si le fichier n'existe pas déjà, si oui on va renvoyer une erreur (sauf si le paramètre pour forcer est True). On va ensuite écrire dedans, à chaque ligne une valeur de la mémoire, des registres, le tas puis on finit par le tableau des pixels. Pour finir on ferme le fichier.

chargement

Paramètres : le nom du fichier
Utilisation : On vérifie que le fichier de sauvegarde existe, si oui on l'ouvre. Ensuite on va faire l'inverse que pour la fonction de sauvegarde, on va lire chaque ligne puis stocké la valeur dans la mémoire, les registres, le tas et enfin le tableau des pixels puis on ferme le fichier.

4)Fichier EmulateurChip8.py

a) Les objets de la fenêtre

fenetreChip8

C'est la variable qui contient la fenêtre principale, elle a un nom (Emulateur CHIP-8), elle ne peut pas être redimensionnée. De plus à la fin du fichier on applique la fonction "mainloop" sur la fenêtre, cela permet de l'afficher.

bouton (Lancer, Mémoire, Sauver, Charger, ->)

Ils ont chacun un nom, une fonction/procédure à lancer quand il est appuyé et enfin un emplacement dans la grille de la fenêtre.

entrée (Nom du programme, nom de la sauvegarde, les registres)

Ils ont chacun un label pour leur donner un nom, une variable qui récupère l'information écrite par l'utilisateur et un emplacement dans la grille.

checkbox (Pas à Pas)

Elle a un nom, une commande à lancer lorsque qu'il y a un changement de la checkbox, une variable qui récupère la valeur de cette checkbox et enfin un emplacement dans la grille.

touches

La fenêtre reçoit l'appuie (<Key>) et le relâchement des touches (<KeyRelease>)

b) Procédures

fonctionnement

Paramètre : aucun
Utilisation : C'est la procédure principale, elle va servir à boucler toutes les 4ms. Tous d'abord on récupère l'action et l'opcode associé grâce à la fonction "interpretation" du fichier opcode. Si on récupère une action qui n'a pas pu être faite dans cette fonction on va la faire, ainsi si on récupère l'action 1 on efface l'écran, si on récupère l'action 23 on va dessiner un sprite et si on récupère l'action 27 on va attendre qu'une touche soit appuyée. Ensuite on rafraichit les registres, on décompte les timers puis enfin on ajoute 2 à PC (On passe à la prochaine instruction). Pour finir si on n'est pas en mode Pas à Pas on va attendre 4 secondes puis relancer la procédure.

decompteSoundDelay

Paramètre : aucun
Utilisation : Cette procédure décrémente DT et ST s'ils sont supérieurs à 0.

effacerEcran

Paramètre : aucun
Utilisation : On dessine un rectangle noir sur tout l'écran puis on met le tableau des pixels à 0.

dessinerSprite

Paramètres : X, Y et N.
Utilisation : On met tout d'abord VF à 0. Ensuite on va boucler N fois. On récupère notre nombre binaire dans la mémoire à l'adresse I plus le numéro de fois qu'on a bouclé. On calcul ensuite la coordonnée en Y où l'on va dessiner. On va boucler sur le nombre binaire pour savoir quel pixel doit changer d'état (si le bit est égal à 1). On calcul la coordonnée en X. Enfin on regarde si le pixel était éteint (On l'allume), si il était allumé on l'éteint et on met VF à 1 (Il y a une collision).

refreshRegistre

Paramètre : aucun
Utilisation : Cette procédure change la valeur affiché des registres avec leur nouvelle valeur.

refreshEcran

Paramètre : aucun
Utilisation : On va parcourir tout le tableau des pixels, si on tombe sur un 1 on allume le pixel sinon on l'éteint.

toucheAppui / toucheRelache

Paramètre : event
Utilisation : On va regarder la touche appuyée ou relâchée (grâce à l'objet event qui nous donne le caractère) et on va ensuite modifier la liste qui représente les touches (1 pour appuyer et 0 pour relâcher).

stepChange

Paramètre : aucun
Utilisation : La procédure va empêcher l'appui sur le bouton pour avancer d'une étape si la checkbox n'est pas cochée et l'autoriser si elle l'est.

pcAvant

Paramètre : aucun

Utilisation : Cette procédure appelle la procédure de fonctionnement pour faire une étape.

lancement

Paramètre : aucun
Utilisation : Cette procédure va permettre de réinitialiser toutes la mémoire, lancer le programme demandé par l'utilisateur puis d'appeler la fonction principale (fonctionnement). Si le programme nommé par l'utilisateur n'existe pas on va afficher un popup d'erreur.

affichageMemoire

Paramètre : aucun
Utilisation : Cette procédure va créer une nouvelle fenêtre pour afficher les valeurs contenus en mémoire. Tout d'abord on crée une nouvelle fenêtre nommée "Mémoire CHIP-8" (elle peut se redimensionner). Elle est dotée d'une scrollbar et les valeurs de la mémoire sont stockées dans une liste, chaque ligne contient 16 valeurs (il y a donc 256 lignes).

sauvegarde

Paramètre : aucun
Utilisation : Cette procédure va lancer la fonction de sauvegarde en lui passant en paramètre le nom choisis par l'utilisateur. Si on obtiens une erreur de la procédure on va demander à l'utilisateur s'il veut supprimer l'ancienne sauvegarde du même nom.

charge

Paramètre : aucun
Utilisation : Cette procédure va lancer la fonction de chargement en lui passant en paramètre le nom choisis par l'utilisateur. Ensuite on rafraîchit l'écran et les registres avec les valeurs de la sauvegarde. On finit par lancer la fonction principale (fonctionnement). Si la sauvegarde nommé par l'utilisateur n'existe pas on va afficher un popup d'erreur.