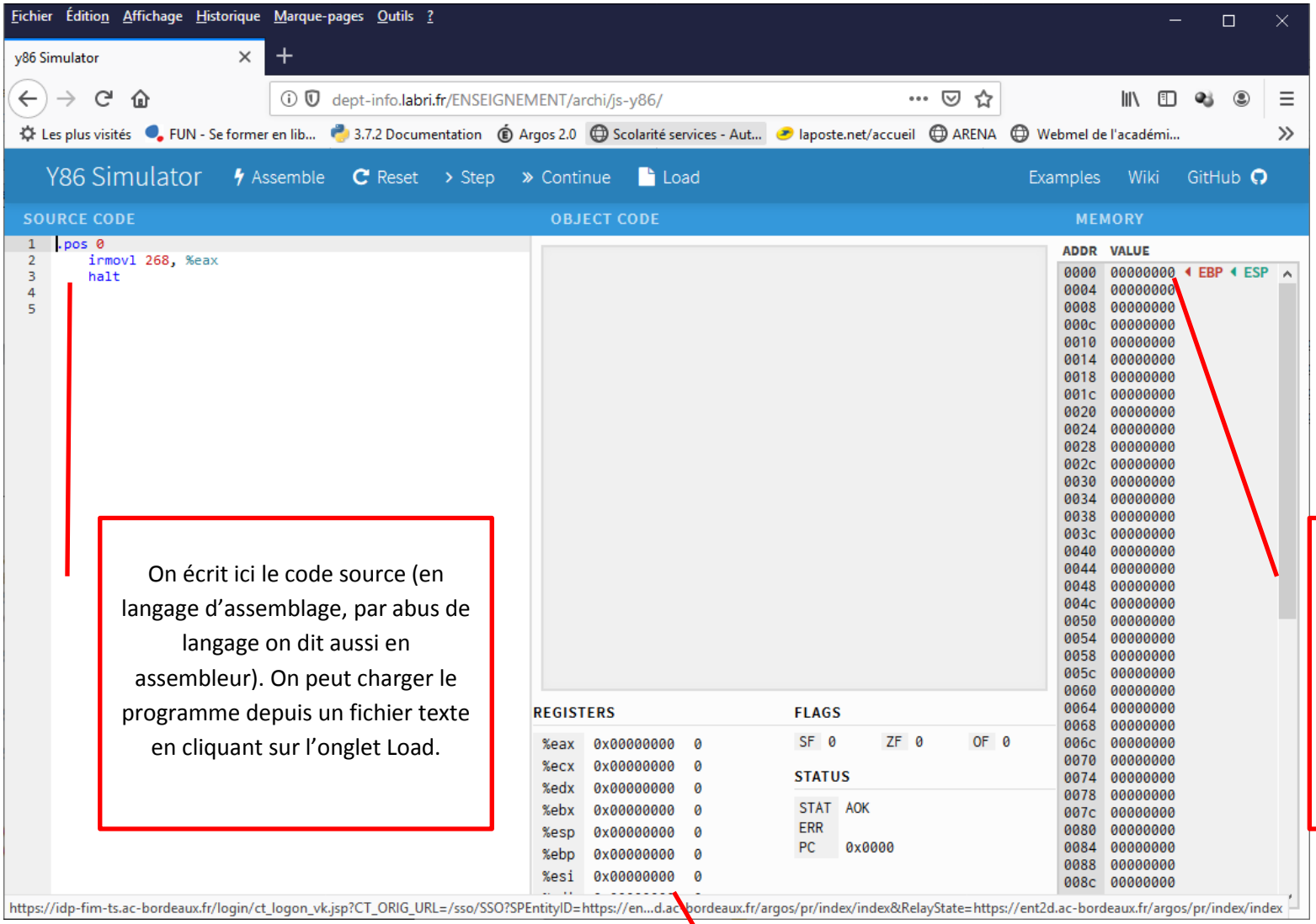
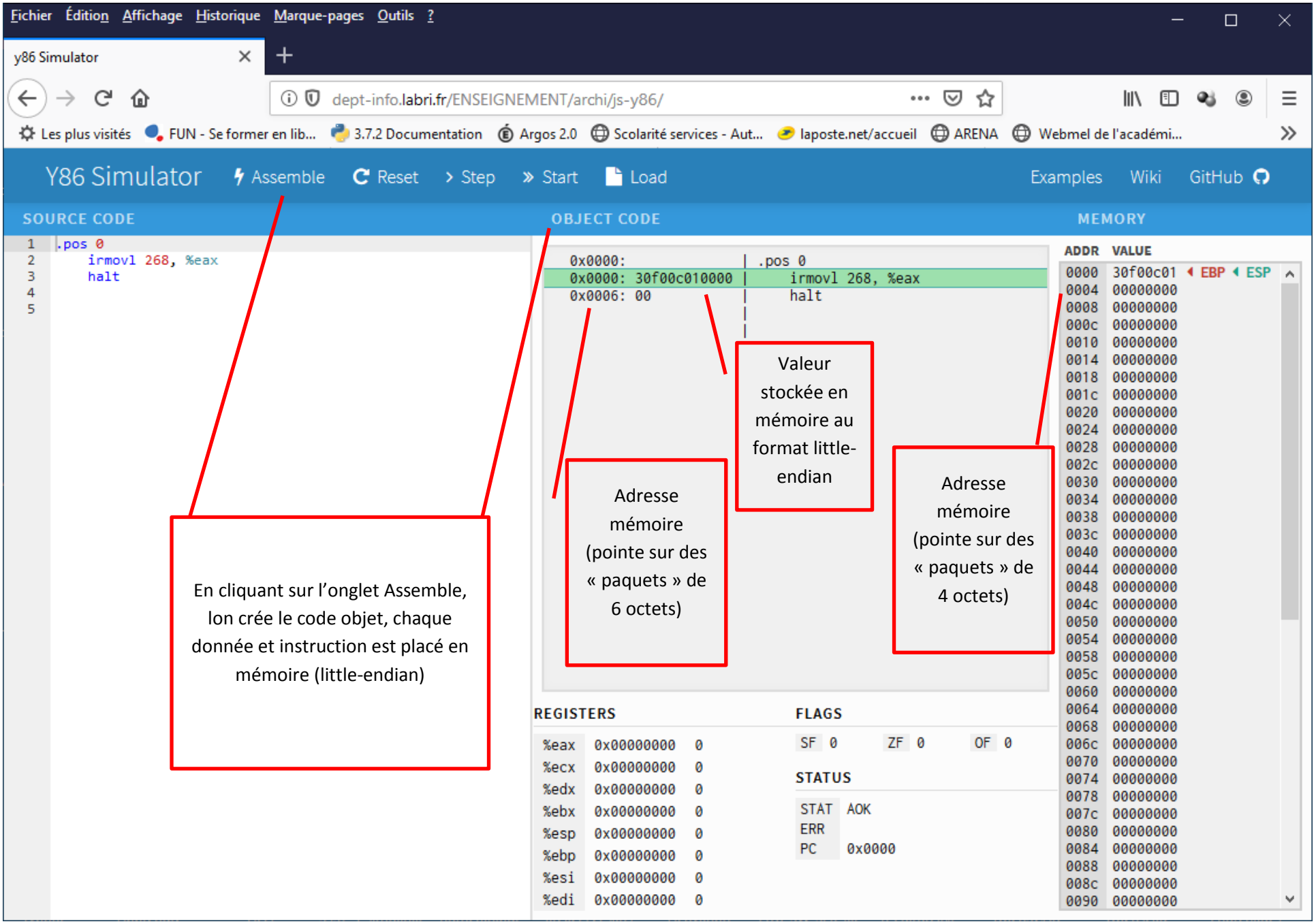
Exemple d'utilisation du simulateur de processeur Y86

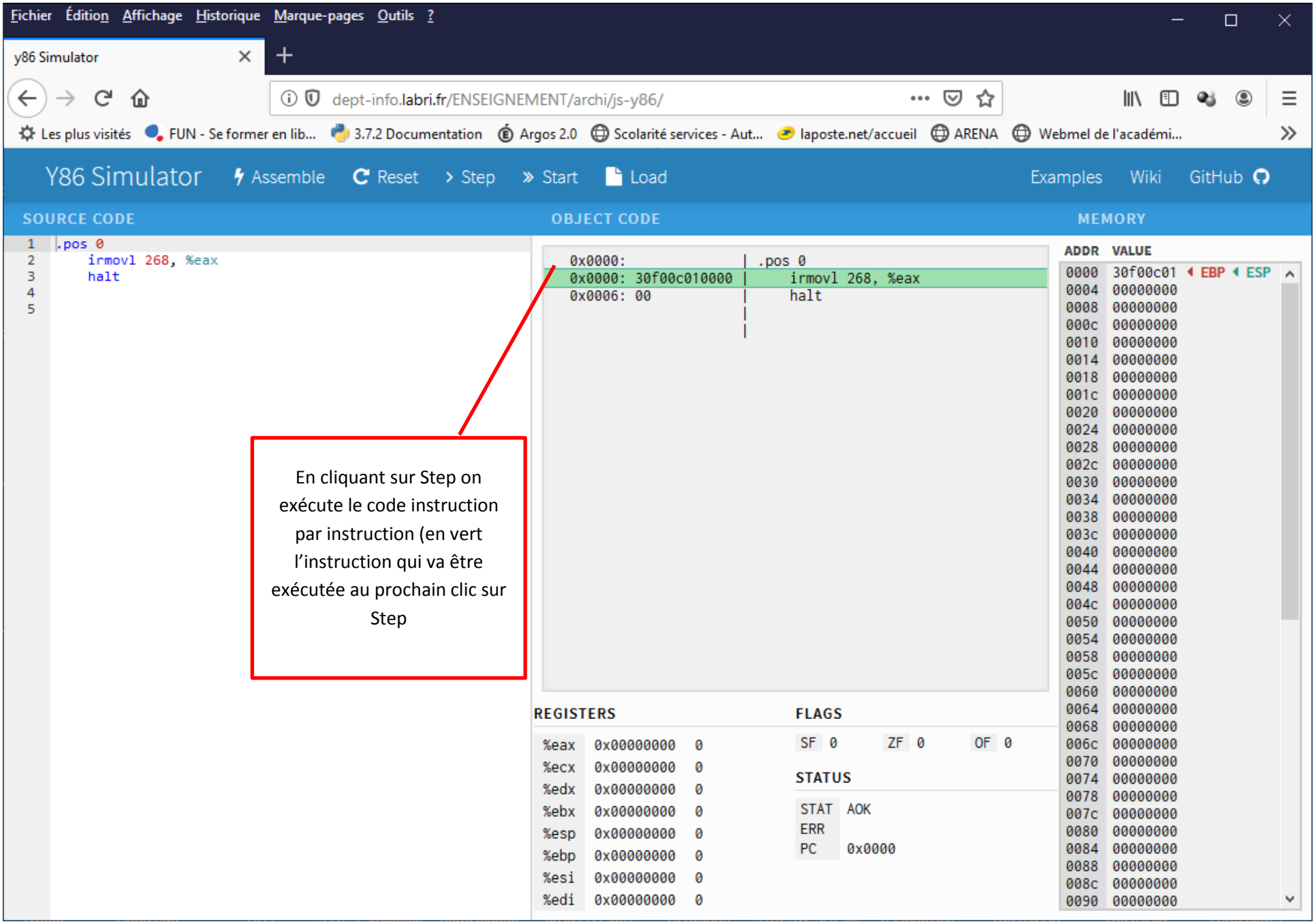


On écrit ici le code source (en langage d'assemblage, par abus de langage on dit aussi en assembleur). On peut charger le programme depuis un fichier texte en cliquant sur l'onglet Load.

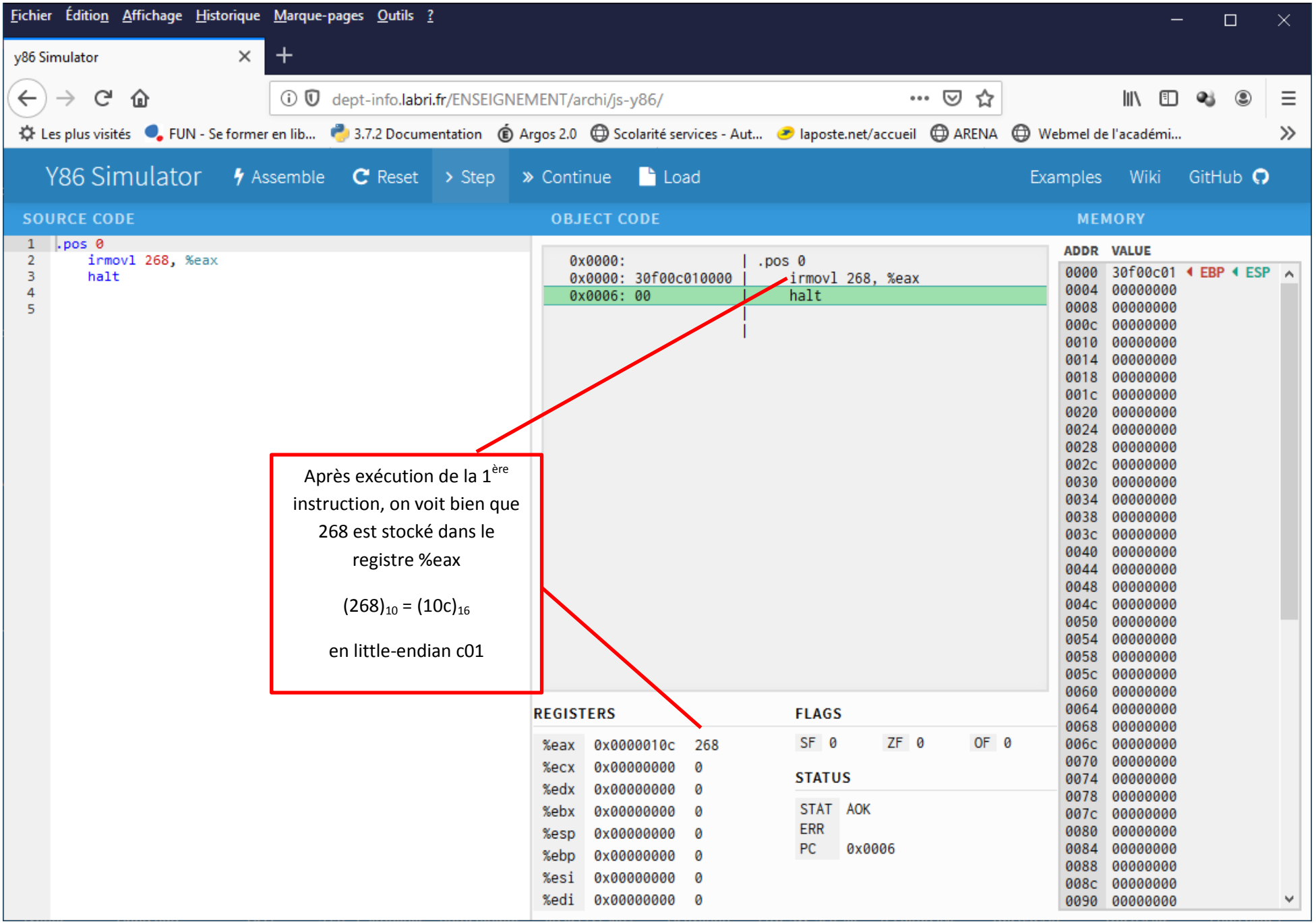
Les mots mémoires sont alignés sur 32 bits et utilise la convention little-endian (stockage des octets dans l'ordre inverse, c'est-à-dire en commençant par l'octet de poids faible).

Le Y86 utilise 8 registres de 32 bits


Fichier Édition Affichage Historique Marque-pages Outils ?
y86 Simulator
dept-info.labri.fr/ENSEIGNEMENT/archi/js-y86/
Les plus visités
FUN - Se former en lib...
3.7.2 Documentation
Argos 2.0
Scolarité services - Aut...
laposte.net/accueil
ARENA
Webmel de l'académi...
Y86 Simulator
Assemble
Reset
Step
Start
Load
Examples
Wiki
GitHub
SOURCE CODE
OBJECT CODE
MEMORY
1 .pos 0
2 irmovl 268, %eax
3 halt
4
5
0x0000: | .pos 0
0x0000: 30f00c010000 | irmovl 268, %eax
0x0006: 00 | halt
ADDR VALUE
0000 30f00c01 EBP ESP
0004 00000000
0008 00000000
000c 00000000
0010 00000000
0014 00000000
0018 00000000
001c 00000000
0020 00000000
0024 00000000
0028 00000000
002c 00000000
0030 00000000
0034 00000000
0038 00000000
003c 00000000
0040 00000000
0044 00000000
0048 00000000
004c 00000000
0050 00000000
0054 00000000
0058 00000000
005c 00000000
0060 00000000
0064 00000000
0068 00000000
006c 00000000
0070 00000000
0074 00000000
0078 00000000
007c 00000000
0080 00000000
0084 00000000
0088 00000000
008c 00000000
0090 00000000
REGISTERS
%eax 0x00000000 0
%ecx 0x00000000 0
%edx 0x00000000 0
%ebx 0x00000000 0
%esp 0x00000000 0
%ebp 0x00000000 0
%esi 0x00000000 0
%edi 0x00000000 0
FLAGS
SF 0 ZF 0 OF 0
STATUS
STAT AOK
ERR
PC 0x0000
En cliquant sur l'onglet Assemble, lon crée le code objet, chaque donnée et instruction est placé en mémoire (little-endian)
Adresse mémoire (pointe sur des « paquets » de 6 octets)
Valeur stockée en mémoire au format little-endian
Adresse mémoire (pointe sur des « paquets » de 4 octets)



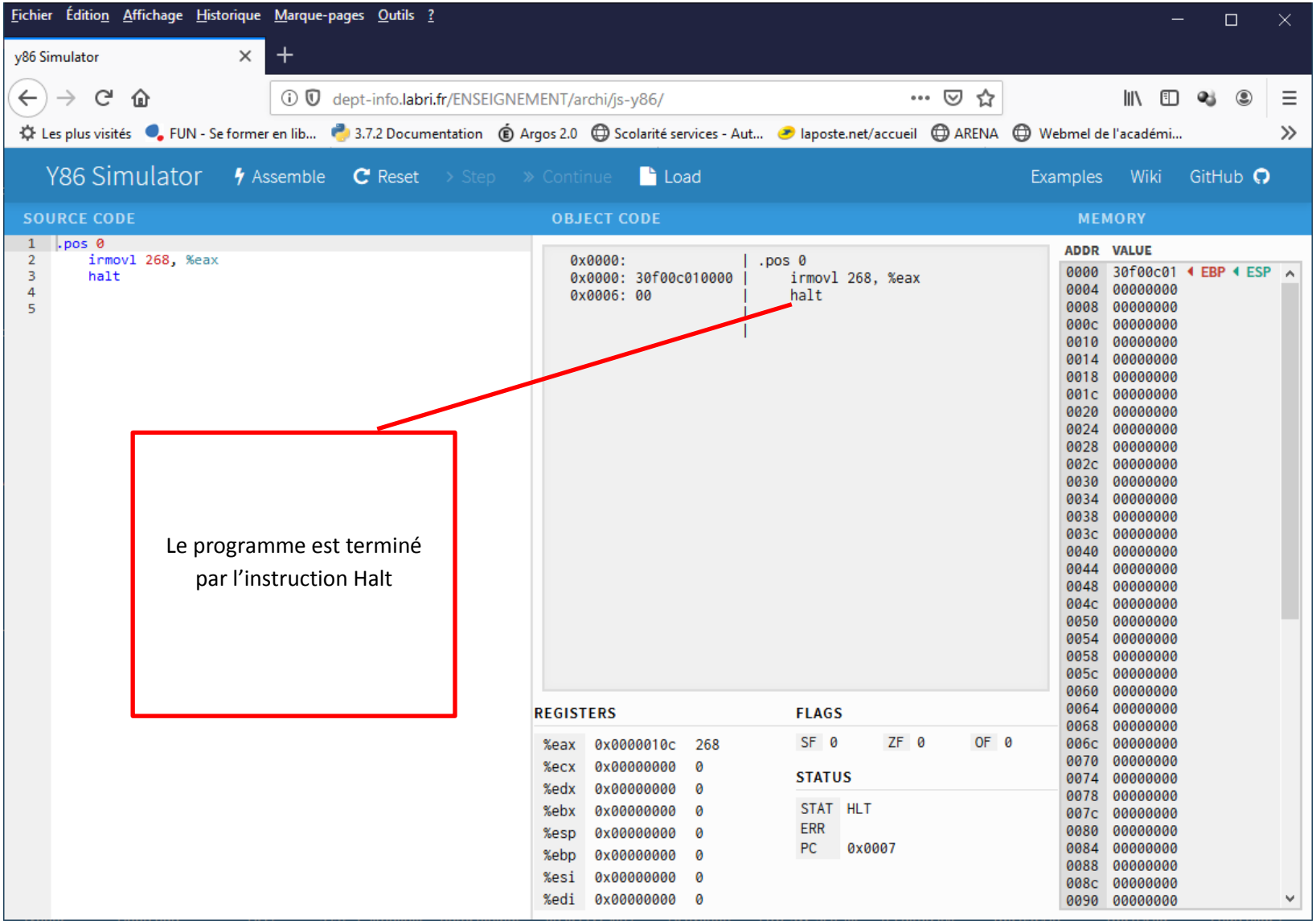
En cliquant sur Step on exécute le code instruction par instruction (en vert l'instruction qui va être exécutée au prochain clic sur Step



Après exécution de la 1ère instruction, on voit bien que 268 est stocké dans le registre %eax

$(268)_{10} = (10c)_{16}$

en little-endian c01


Fichier Édition Affichage Historique Marque-pages Outils ?
y86 Simulator
dept-info.labri.fr/ENSEIGNEMENT/archi/js-y86/
Les plus visités
FUN - Se former en lib...
3.7.2 Documentation
Argos 2.0
Scolarité services - Aut...
laposte.net/accueil
ARENA
Webmel de l'académi...
Y86 Simulator
Assemble
Reset
Step
Continue
Load
Examples
Wiki
GitHub
SOURCE CODE
OBJECT CODE
MEMORY
1 .pos 0
2 irmovl 268, %eax
3 halt
4
5
0x0000: | .pos 0
0x0000: 30f00c010000 | irmovl 268, %eax
0x0006: 00 | halt
ADDR VALUE
0000 30f00c01 EBP ESP
0004 00000000
0008 00000000
000c 00000000
0010 00000000
0014 00000000
0018 00000000
001c 00000000
0020 00000000
0024 00000000
0028 00000000
002c 00000000
0030 00000000
0034 00000000
0038 00000000
003c 00000000
0040 00000000
0044 00000000
0048 00000000
004c 00000000
0050 00000000
0054 00000000
0058 00000000
005c 00000000
0060 00000000
0064 00000000
0068 00000000
006c 00000000
0070 00000000
0074 00000000
0078 00000000
007c 00000000
0080 00000000
0084 00000000
0088 00000000
008c 00000000
0090 00000000
REGISTERS
%eax 0x0000010c 268
%ecx 0x00000000 0
%edx 0x00000000 0
%ebx 0x00000000 0
%esp 0x00000000 0
%ebp 0x00000000 0
%esi 0x00000000 0
%edi 0x00000000 0
FLAGS
SF 0 ZF 0 OF 0
STATUS
STAT HLT
ERR
PC 0x0007
Le programme est terminé par l'instruction Halt