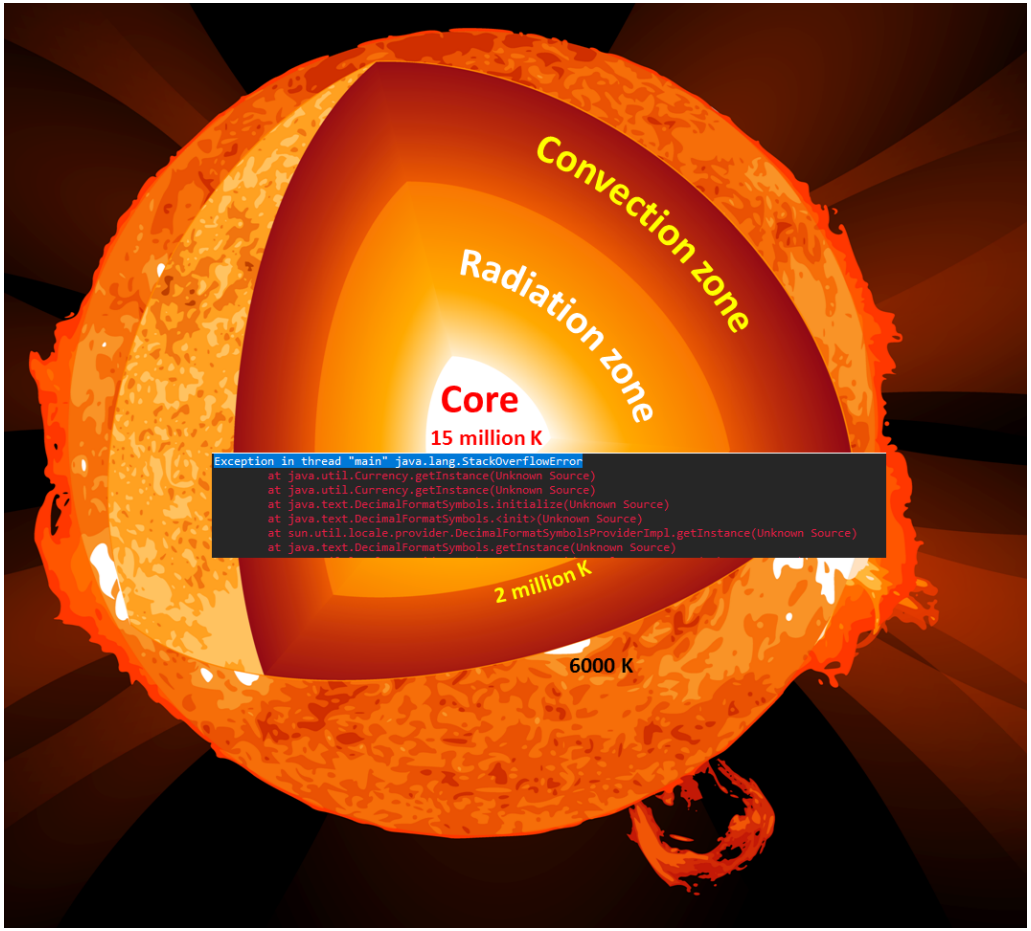
Wskaźniki i pamięć komputera*

Bartłomiej Kliś

(*i inne niskopoziomowe gówno)

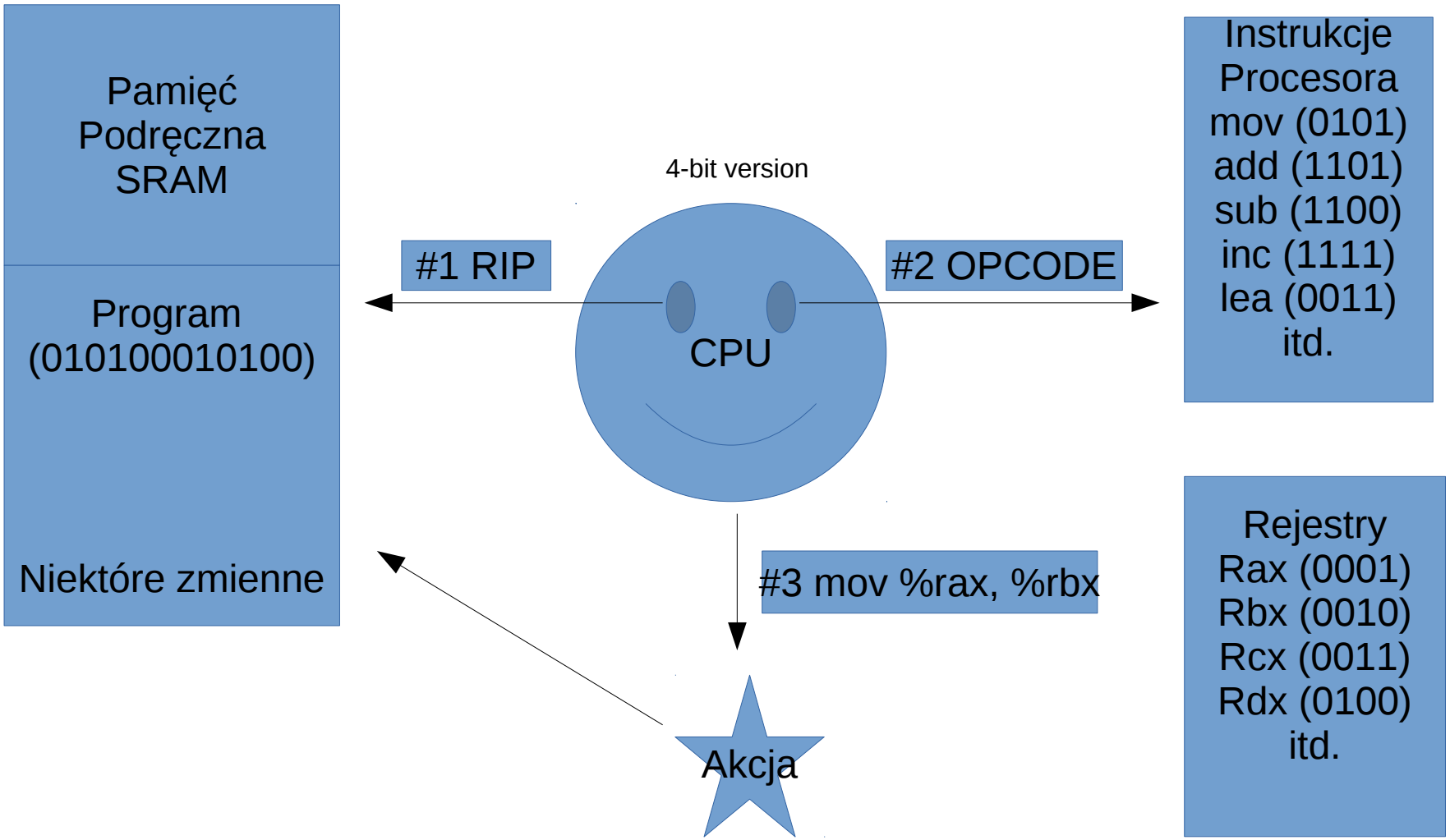
Co w tym ciekawego?

- Wykorzystywanie do granic możliwości własnego sprzętu i nie tylko (procesor i jego wątki, karta graficzna->CUDA, peryferia, IoT)
- Pisanie kodu którego nikt nie rozumie i uważa za przeżytek*
- Minimalizacja rozmiarów programów (oszczędność której prawie nikt nie szanuje)
- Hacking, można rozwalić każdy inny program panując nad chaosem.
- Znajomość niskopoziomowych aspektów programowania przekłada się na pisanie świetnych programów w wysokopoziomowych językach oraz pozwala na tzw. cross programing.
- Można pisać własne języki oraz stosowne do nich kompilatory
- Można stracić zdrowie psychiczne i w rezultacie życie pisząc przez dekadę własny system operacyjny (RIP Terry A. Davis, TempleOS).



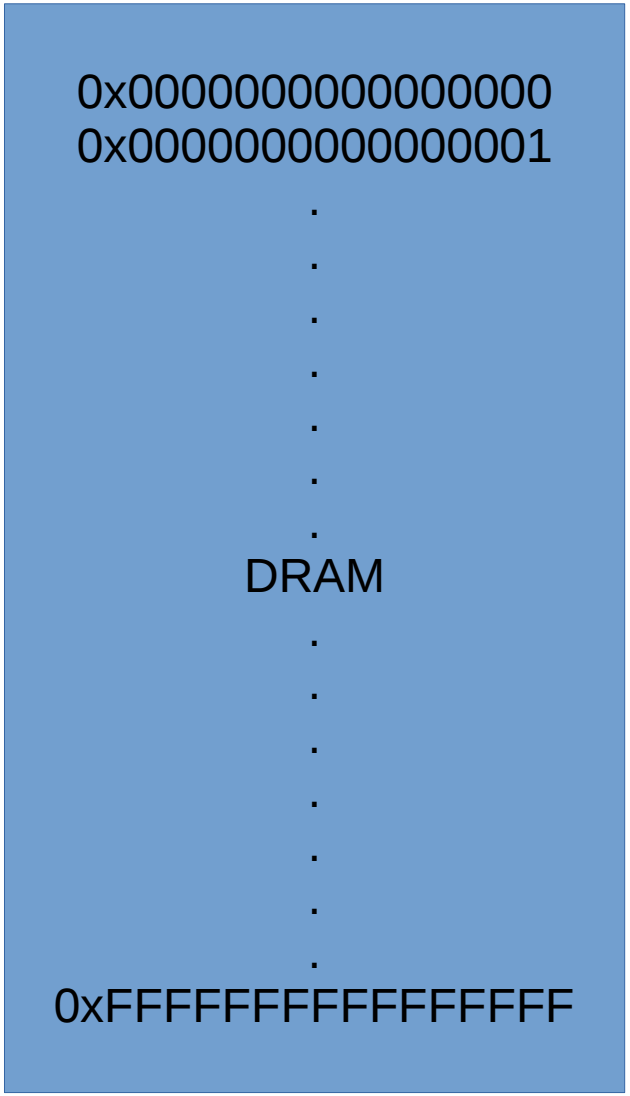
(*bo żyje świecie w którym słońce płonie dzięki superkomputerowi liczącym całą symulację w javie)

Komputer



100x lepiej niż ja: Richard Feynman Computer Heuristics Lecture: https://www.youtube.com/watch?v=EKWGGDXe5MA

Pamięć komputera



- Wszystkie programy, zmienne, pliki i dane są ładowane do pamięci o dostępie swobodnym (RAM)
- Z pamięci tej instrukcje programu przerzucane są do pamięci podręcznej procesora
- Operacje na pamięci to sedno działania każdego programu, mimo że nie zawsze piszemy w czymś co pozwala to obserwować

Wskaźniki

- Zgodnie z nazwą „wskazują” one na coś w programach. Fizycznie są to zmienne przechowujące adress jakiejś innej zmiennej. W współczesnych komputerach są formatu *unsigned long int* o rozmiarze 8 bajtów, wynika to z faktu że obecne komputery są 64 bitowe.
- Maksymalny adress wynosi

  0xFFFFFFFFFFFFFFFF = 2^64 b = 16 EiB (exbi)

  pozwala to teoretycznie na adresację pamięci mającą takie rozmiary.

Kod mi padł, padł mi kod - Debugger

- Podstawowe komendy:
- run (r) – uruchamia program „r (argumenty)”
- backtrace (bt) – wskazuje funkcje które zostały wywołane po drodze do punktu w którym obecnie się znajduje
- list (l) -wyświetla kod programu
- info args - podaje argumenty funkcji na której zatrzymał się debugger)
- info locals - podaje zmienne wewnątrz funkcji/bloku
- break (b) – ustawia breakpoint w którym program ma być zatrzymany podczas debuggowania b (miejsce)
- step (s) – przeskakuje o instrukcję (si – przeskakuje o instrukcję maszynową)
- next (n) – przeskakuje o funkcję
- continue (c) - wraca do wykonywania programu
- x – wypisuje obszar pamięci pod wskaźnikiem

  x/(liczba)(typ: x, c, d, f, p)(wielkość: b, h, w, g)

GDB: The GNU Project Debugger

Zasoby

- Literatura:
  - Jon Erickson „Hacking the art of exploitation”, 0x200 Programing
  - Richard Blum „Professional Assembly Programing”
- Wykłady:
  - Richard Feynman Computer Heuristics Lecture: https://www.youtube.com/watch?v=EKWGGDXe5MA
- Twórcy z internetu:
  - Bisqwit: https://www.youtube.com/channel/UCKTehwyGCKF-b2wo0RKwrcg
  - GynvaelColdwind: https://www.youtube.com/channel/UCjS2aGCvsnhExcWRAI8T4Pw
  - Javidx9: https://www.youtube.com/channel/UC-yuWVUplUJZvieEligKBkA
  - CLNohr: https://www.youtube.com/channel/UCG7yIWtVwcENg_ZS-nahg5g
  - bitluni's lab: https://www.youtube.com/channel/UCp_5PO66faM4dBFbFFBdPSQ
  - Null Byte: https://www.youtube.com/channel/UCgTNupxATBfWmfehv21ym-g
  - HackerSploit: https://www.youtube.com/channel/UC0ZTPkdxlAKf-V33tqXwi3Q/videos
  - Applied Science: https://www.youtube.com/channel/UCivA7_KLKWo43tFcCkFvydw