Wybrany system kontroli wersji:

- Launchpad https://launchpad.net/

Wybrana metodyka

- TDD http://en.wikipedia.org/wiki/Test-driven_development

Wymagania funkcjonalne wersji 0.0.1

- widok główny zawiera domyślnie 4 okna:
- okno kodu - okno, w którym użytkownik może wprowadzić kod asemblera, zawiera menu (przyciski):
- uruchom
- uruchom ze śledzeniem instrukcji
- wykonaj następną instrukcję
- przerwij program
- wznów program
- zakończ program

- okno pamięci - pokazuje stan pamięci
- okno rejestrów - pokazuje stan rejestrów
- okno schematu - pokazuje predefiniowany schemat mikrokontrolera

- użytkownik może ukryć/pokazać każde z okien
- użytkownik może kazać programowi wykonanie wprowadzonego kodu asemblera (wybiera opcję "uruchom", "uruchom ze śledzeniem instrukcji" lub "wykonaj następną instrukcję")
- program wykonuje kod wprowadzony w oknie kodu zmieniając stan pamięci, rejestrów i wyjścia

- użytkownik może na bieżąco śledzić w oknie kodu wykonywane instrukcje (wybiera opcję "uruchom ze śledzeniem instrukcji", przydałoby się opóźnić "skok" do następnej instrukcji, by animacja była płynna)

- użytkownik może zatrzymać działanie programu w dowolnej chwili (wybiera opcję "przerwij program") a następnie wznowić jego działanie z tego samego momentu (wybiera opcję "wznów program")
- użytkownik może zakończyć działanie programu w dowolnej chwili (wybiera opcję "zakończ program")
- użytkownik może wykonać jedną, następną instrukcję (wybiera opcję "wykonaj następną instrukcję")
- program powinien być kompatybilny z rzeczywistym procesorem (wprowadzony kod powinien działać tak samo w naszym emulatorze, w RIDE i na rzeczywistym procesorze)

Jak to zrobimy (i w jakiej kolejności ;))?

- stworzymy najpierw emulator procesora 8051 (moduł czytający z pamięci kolejne opkody i wykonujący operacje na pamięci, rejestrach itd.)
- kod asemblera będzie tłumaczony na kod maszynowy (asemblacja) i umieszczany w pamięci
- zaprojektujemy GUI używając odpowiedniej biblioteki (np. jQuery UI): okienka kodu, pamięci i rejestrów.
- początki interfejsu do tworzenia układów (flash)

Oczywiście zanim przystąpimy do pisania właściwego kodu będziemy, zgodnie z TDD, pisać odpowiednie testy (może potem wrzuci się tu linka z jakimś artykułem o dobrych i złych praktykach pisania testów w TDD).