Postavljanje razvojnog okruženja za Contiki i postavljanje Contiki slika na Texas Instruments CC2560

Seminarski rad, Bežične Mreže Osjetila, Ak. God. 2019/2019
Mentor: doc. dr. sc. Damir Arbula
Dražen Milković, Erik Smoljan, Sandi Šegota

1 Postavljanje razvojnog okruženja

1.1 Virtualni stroj

Najjednostavniji način postavljanja razvojnog okruženja za Contiki je korištenje virtualnog stroja dostupnog sa službenih stranica. Preuzmemo virtualni stroj i pokrenemo ga korištenjem VMWare Player programa. (https://sourceforge.net/projects/contiki/files/Instant%20Contiki/).
U virtualnom stroju se nalaze svi alati potrebni za kompilaciju Contiki na podržanim mikrokontrolerima, kao i Cooja simulator.

1.2 Instalacija na Linux OS

Potrebno je prvo instalirati potrebne preduvjete za pokretanje programa.

```
$ sudo apt update
$ sudo apt install build-essential doxygen git curl wireshark
python-serial srecord rlwrap
```

Zatim je potrebno instalirati compiler za ARM. Ovo možemo postići naredbom:

```
$ sudo apt install gcc-arm-eabi-none
```

Ovisno o Contiki verziji i verziji kompajlera u repozitorijima moguće je da će dolaziti do problema kod kompilacije. U tom slučaju potrebno je skinuti verziju kompajlera koja potvrđeno radi i instalirati je ručno. Stariju verziju kompajlera možemo preuzeti sa stranice:

https://launchpad.net/gcc-arm-embedded/5.0/5-2015-q4-major/+download/gcc-arm-none-eabi-5_2-2015q4-20151219-linux.tar.bz2

Nakon ekstrakcije arhive ona kreira direktorij naziva gcc-arm-none-eabi-5_2-2015q4. Taj direktorij je potrebno dodati u našu putanju naredbom:

```
$ export
PATH=$PATH:/path/to/directory/gcc-arm-none-eabi-5_2-2015q4
```

Također može biti korisno postaviti pristup USB ulazima/izlazima bez korištenja sudo naredbe, za potrebe postavljanja slika na sustav. Ovo postižemo tako što trenutnog korisnika postavimo kao člana grupa plugdev i dialout, odnosno naredbama:

```
$ sudo usermod -a -G plugdev <user>
$ sudo usermod -a -G dialout <user>
```

Na samom kraju, preuzimamo Contiki iz git repozitorija, te inicijaliziramo submodule.

```
$ git clone https://github.com/contiki-os/contiki
$ cd contiki
$ git submodule update --init --recursive
```

1.2.1 Instalacija Cooja simulatora

Cooja simulator je koristan alat koji nam omogućava testiranje i debugiranje programa bez potrebe za fizičkim mikrokontrolerom. Iako Cooja simulator nije nužan alat, veoma je koristan kod razvoja pa smo njegovo postavljanje i kratke upute za korištenje odlučili uključiti u ovu dokumentaciju. Kako bismo koristili simulator potrebno je instalirati slijedeće alate:

```
$ sudo apt install default-jdk ant
```

Zatim postavimo varijablu JAVA_HOME u datoteci .profile, u home direktoriju korisnika.

```
$ echo 'export JAVA_HOME="/usr/lib/jvm/default-java"' >> ~/.profile
```

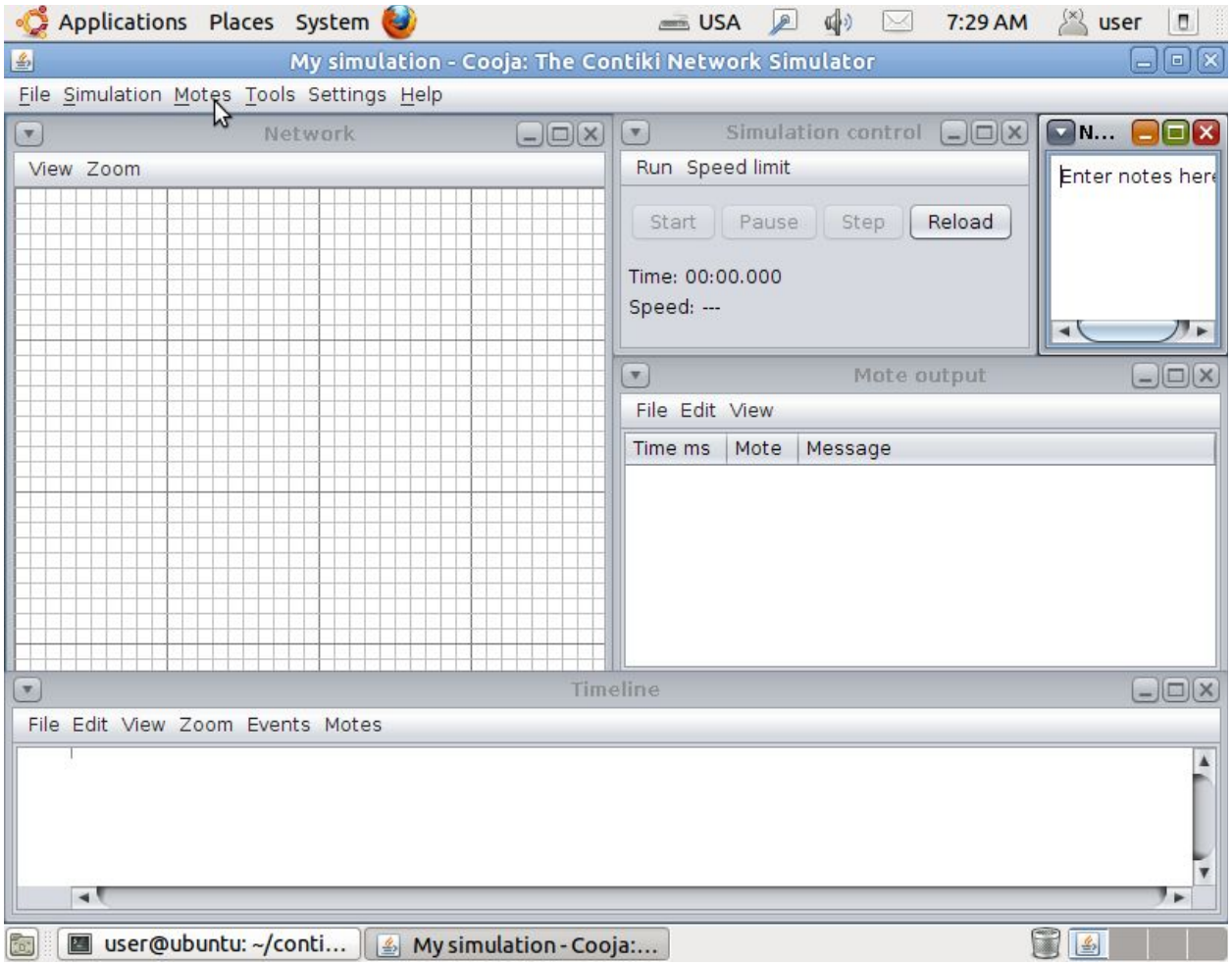
Ukoliko Cooja simulator nije kompatibilan sa Java verzijom koja je postavljena kao zadana na našem sustavu, možemo promijeniti verziju korištenjem naredbe:

```
$ update-alternatives --config java
```

Cooja simulator pokrećemo naredbom ant run iz direktorija u kojemu se nalazi.

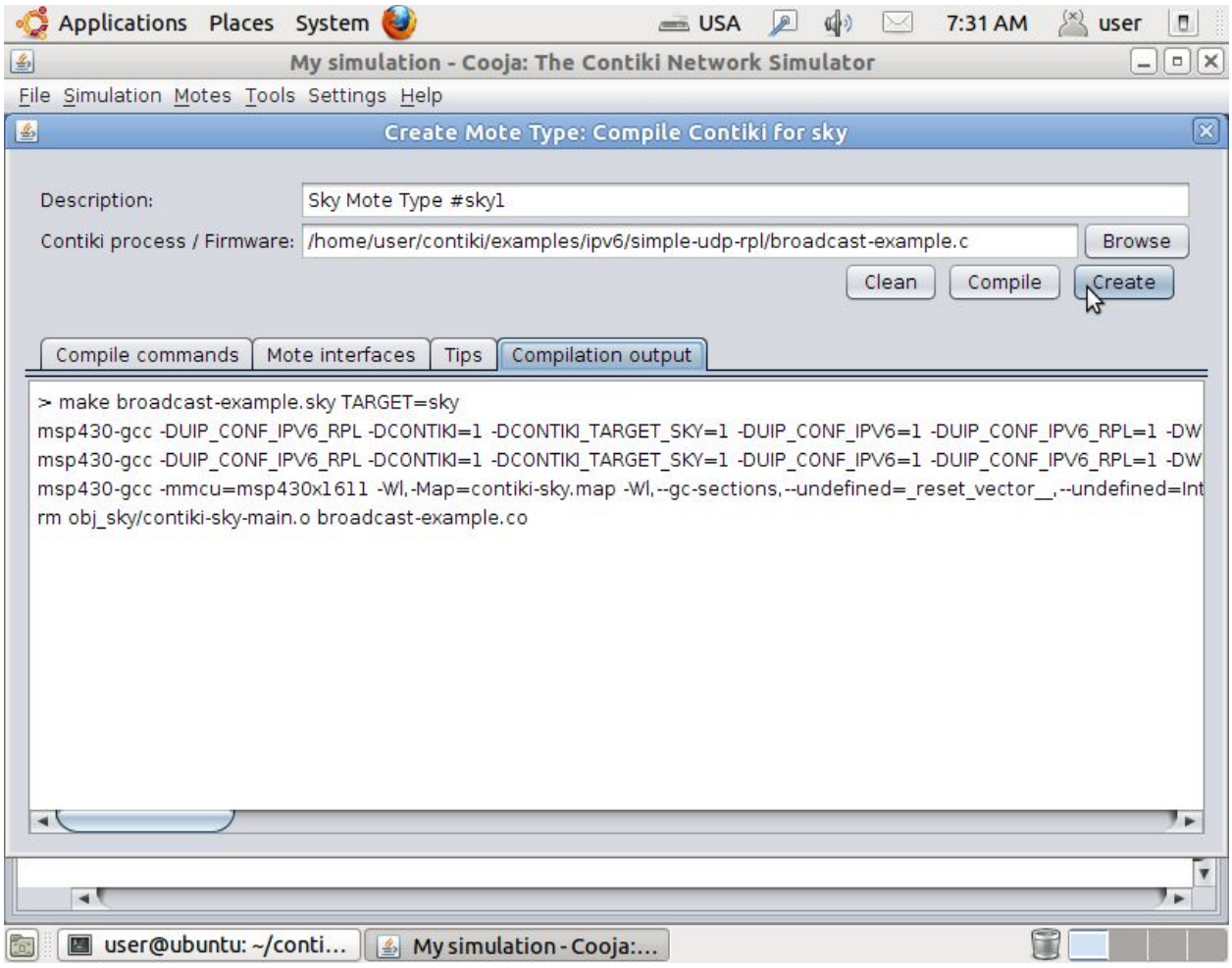
```
$ cd /path/to/contiki/tools/
/path/to/contiki/tools/cooja$  ant run
```

U Cooja simulatoru možemo postaviti jednu simulaciju odabirom *File->New Simulation*.



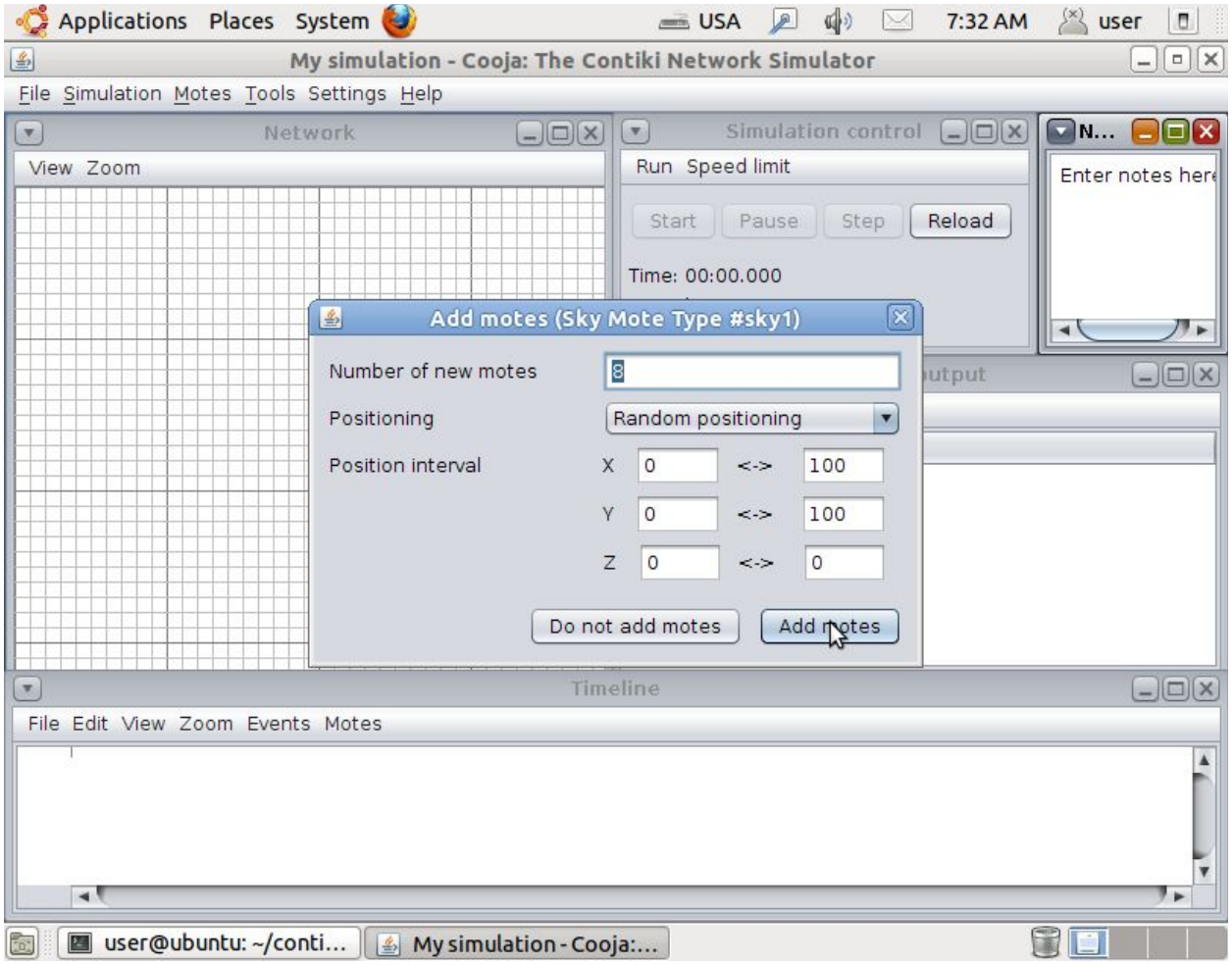
Slika 1. Izgled novootvorene simulacije u Cooja simulatoru

Zatim dodajemo novi mote klikom na *Motes -> Add motes.* (Slika 2.)



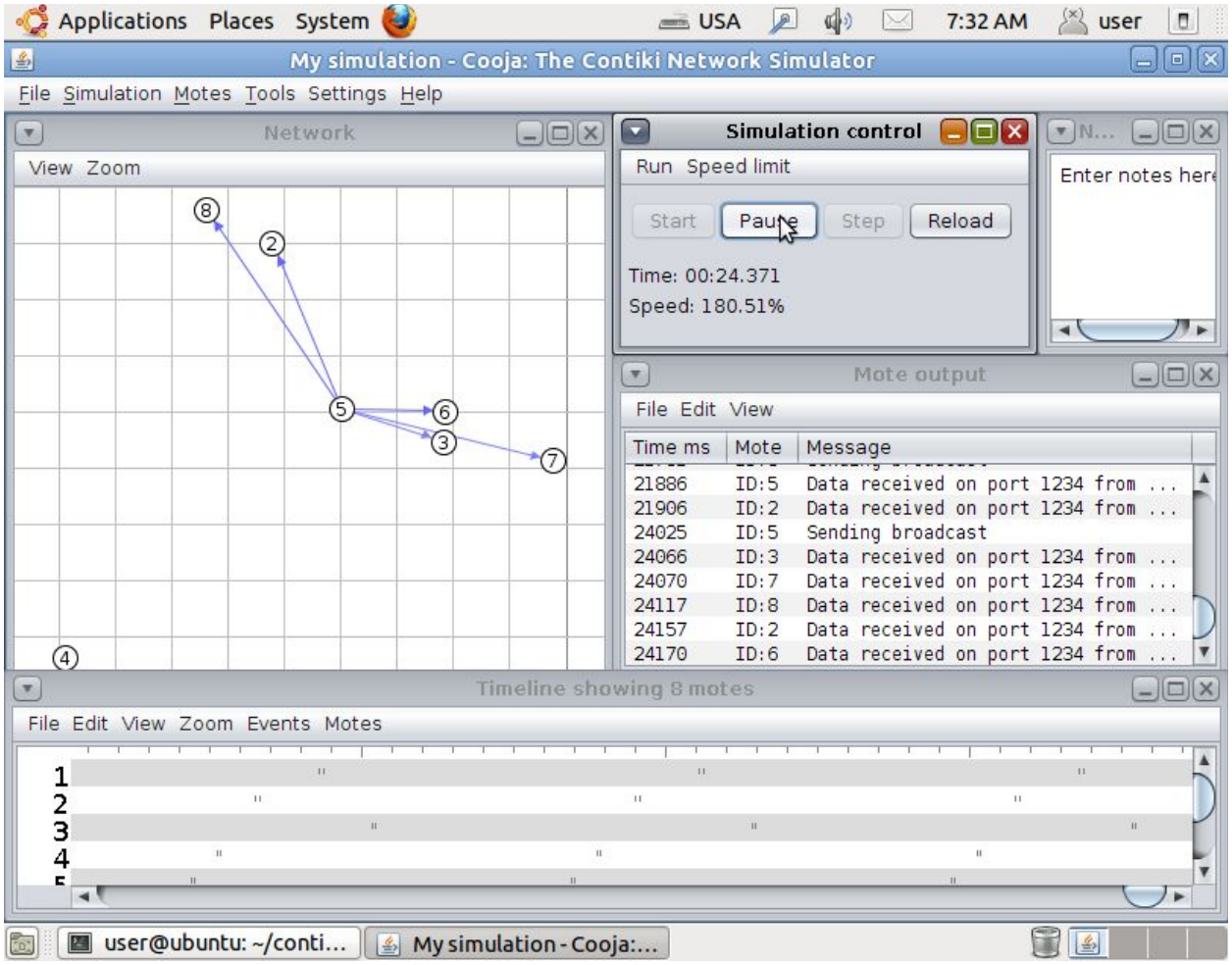
Slika 2.Dodavanje novog mote tipa i kompilacija firmwarea

Trenutno ne postoji mote za mikrokontroler koji smo mi koristili, ali možemo koristiti Cooja mote tip. Mana ovoga što ne možemo koristiti naredbe specifično vezane uz naš sustav. Iako, često je i ovakav drugi tip motea dovoljno dobar za testiranje logike sustava. Bitno je napomenuti da je potrebno izbjegavati korištenje drugih tipova moteova, vezanih uz specifične mikrokontrolere - pošto prilagodba firmwarea za njih može uzrokovati poteškoće kod prijenosa na naš tip mikrokontrolera (npr. zbog mote-specific headera). Dodanom moteu zadajemo putanju do C programa u kojem je razvijen firmware. Taj program se kompajlira, te kada je taj postupak završen klikom na “Create” otvaramo dijalog u kojemu biramo broj moteova koji želimo postaviti u simulaciju (Slika 3.)



Slika 3. Dodavanje moteova u simulaciju

U simulaciji možemo postaviti brzinu izvršavanja simulacije, izmjenjivati prikaz (npr. dodati prikaz poruka ili mote identifikatora) te pokretati i zaustavljati simulaciju. Tako, "*Mote Output*" pod prozor prikazuje ispis podataka na svim moteovima (npr. korištenjem naredbe printf u C programu), Timeline prikazuje komunikaciju i RF događaje u vremenu. Također, postoji praktičan prozor za dodavanje bilješki vezanih uz simulaciju. Ovo je prikazano na Slici 4.



Slika 4. Izgled Cooja simulacije u tijeku

Otvorena simulacija se sprema u CSC formatu. Kod ponovnog pokretanja Cooja simulatora možemo otvoriti ranije postavljenu simulaciju sa *File->Open Simulation*.

2 Izrada i postavljanje slike na sustav

Ako su prijašnji koraci bili uspješni imamo spremno okruženje za razvoj. Možemo ga testirati korištenjem nekog od primjera iz <contiki-path>/examples direktorija, ili vlastitog programa.

2.1 Izrada slike

Kako bismo izradili sliku sustava otiđemo u direktorij u kojemu se program nalazi. U tom direktoriju bi se trebali nalaziti programski kod, *.c formata i Makefile. Ako sastavljamo programski kod za određeni mikrokontroler, tada se tamo može nalaziti i project_conf.h header dokument koji sadržava naredbe specifične za taj sustav.

Kako bismo izradili sliku koristimo naredbu make. Ako pokrenemo naredbu make bez dodatnih argumenata ona će izraditi sliku za nativni sustav. Ako želimo sliku koju ćemo postaviti na naš sustav moramo specificirati argumente TARGET i BOARD. U osnovi, ovdje se radi o cross-kompilaciji. TARGET označava sustav na kojemu će se naš firmware izvršavati. BOARD označava verziju mikrokontrolera, tj. pločicu na koju postavljamo sustav. Ako želimo vidjeti koji su nam targeti/pločice na raspolaganju to možemo vidjeti u direktoriju <contiki-path>/platform/. Sam board argument možemo provjeriti unutar tog direktorija, unutar direktorija našeg targeta, npr. <contiki-path>/platform/srf06-cc26xx .

Za naš sustav ove vrijednosti su:

- TARGET = srf06-cc26xx
- BOARD = srf06/cc2650

Pa je naredba koju koristimo za kompilaciju:

```
$ make TARGET=srf06-cc26xx BOARD=srf06/cc2650
```

U našem slučaju BOARD argument je jednak default argumentu, pa ga ne moramo koristiti. Odnosno, dovoljno je samo pokrenuti sljedeću naredbu.

```
$ make TARGET=srf06-cc26xx
```

Kada bismo koristili neku drugu verziju CC2650 kontrolera, npr. sensortag ili launchpad, morali bi koristiti odgovarajući argument - BOARD=sensortag, odnosno BOARD=launchpad.
Također, prije kompilacije, savjetuje se brisanje datoteka kreiranih kod prethodnog kompajliranje. Ovo se može postići naredbom:

```
$ make TARGET=srf06-cc26xx clean
```

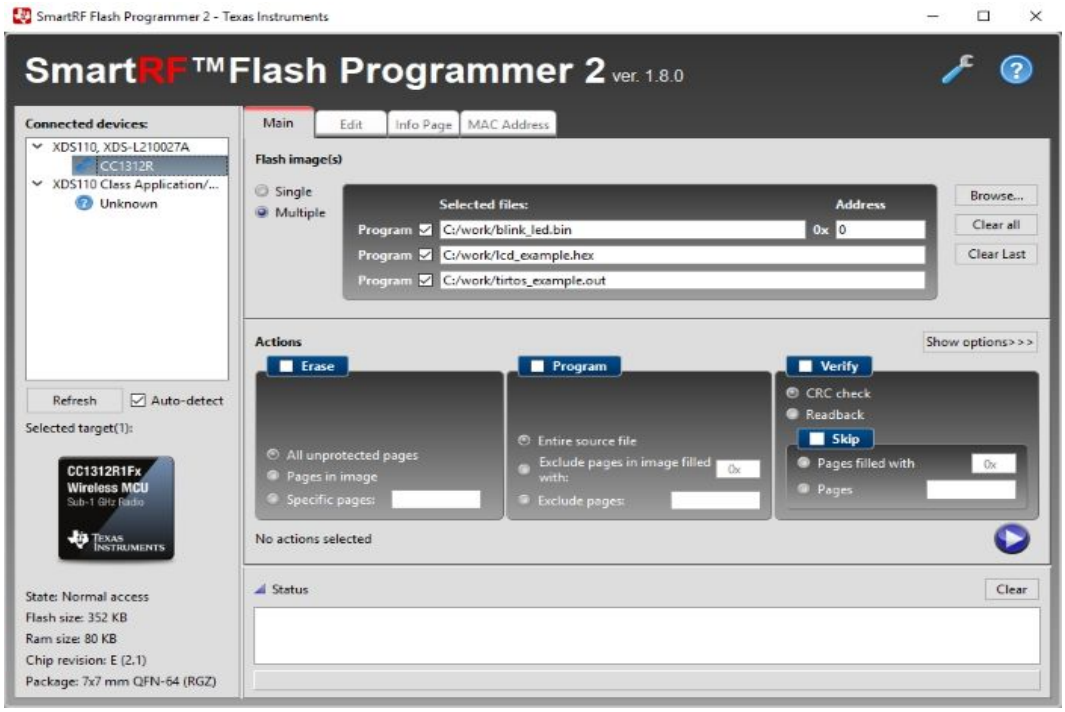
Ova naredba će kompajlirati naš kod prema poostojećem Makefileu, te ako je kompilacija uspješna, kreirati slike sustava u formatu HEX, BIN i ELF. Ove slike možemo postaviti na sustav i pokrenuti naš firmware. Ovaj postupak je opisan u nastavku.

2.2 Postavljanje slike na sustav

Sliku na TI CC2650 možemo postaviti korištenjem službenih alata *Uniflash* (dostupan za Linux, macOS i Windows) i *SmartRF Flash Programmer 2* (Dostupan za Windows platformu) . Preporuča se korištenje *SmartRF Flash Programmer 2* (http://www.ti.com/tool/flash-programmer) alata, s čime se slažemo pošto smo tokom izrade rada imali problema sa nestabilnošću alata prilikom pokušaja korištenja *Uniflash* alata, te smo odlučili koristiti *SmartRF Flash Programmer 2.* Oba alata su besplatna i dostupna sa stranica Texas Instruments:

- Uniflash - http://www.ti.com/tool/uniflash
- Texas Instruments - http://www.ti.com/tool/flash-programmer

SmartRF Flash Programmer 2 alat dolazi u .zip arhivi s čarobnjakom za instalaciju. Tokom instalacije se instaliraju i svi potrebni driveri potrebni za pokretanje. Nakon preuzimanja i instalacije *SmartRF Flash Programmer 2* (nadalje: *SmartRF*) alata. SmartRF alat je vidljiv na Slici 5.

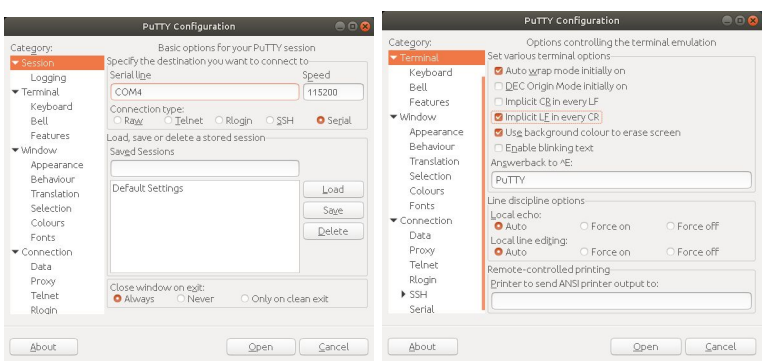


Slika 5. Sučelje SmartRF Flash Programmer 2 programa

Nakon što spojimo naše mikrokontrolere na računalo putem USB priključka SmartRF bi ih trebao automatski prepoznati(moguće je da će Windows sustav prvo instalirati dodatne drivere za kontrolere). Nakon što su kontroleri povezani, možemo ih programirati na sljedeći način:

1. Pod *"Flash Images"* unesemo ili odaberemo putanju do slike sustava u binarnom formatu (BIN). Moguće je koristiti i više slika istovremeno.
2. Iz izbornika lijevo, lijevim klikom odaberemo mikrokontroler na koji želimo postaviti sliku.
3. Pod *"Activities"* lijevim klikom odaberemo *"Erase"* (obriši memoriju mikrokontrolera prije postavljanja slike), *"Program"* (postavi sliku na sustav) i *"Verify"* (provjeri je li slika uspješno i točno postavljena).
    a. Nije nužno uvijek koristiti *"Erase"* i *"Verify*", ali je njihovo korištenje preporučeno kako bi se izbjegli neočekivani problemi.
4. Kliknemo na gumb za programiranje - plavi krug sa bijelim trokutom
5. U dijelu prozora *"Status"* će se pokazati trenutne akcije kod postavljanja slike kao i progress bar.
6. Ako je Progress bar potpuno zelen, programiranje je uspješno.
    a. Ukoliko nije, provjeriti sliku, njenu lokaciju, kontroler te vezu na kontroler.

Nakon instalacije možemo provjeriti pokreće li se program, odnosno ispisuju li se poruke koje se prema kodu za firmware trebaju ispisivati (npr. *printf("Hello World!")*). Za ove potrebe, pošto se već nalazimo na Windows OS, preporučamo korištenje programa PuTTY - besplatnog SSH i telnet klijenta te simulatora serijske veze.
U PuTTY odaberemo lokaciju na koju se želimo spojiti. Zatim odaberemo port na kojem je naš mikrokontroler spojen (Vidljivo u Windows sustavu pod *Control Panel -> Devices*) - npr. COM4 ili COM6. Bitno je i postaviti brzinu čitanja serijske linije, kako bismo dobili čitljivi output - u našem slučaju je brzina 115200 baud. Također je korisno u lijevom izbornikom, pod *"Terminal",* odabrati opciju *"Implicit LF in every CR"*, što će kod svakog *Carriage Return* simbola automatski unijeti i *Line Feed* simbol. Ovom opcijom izbjegavamo pomicanje teksta udesno kod svake nove linije u Terminal prozoru naše konekcije. Ove postavke su vidljive na Slici 6.
Nakon postavljanja potrebnih postavki pokrenemo našu sesiju klikom na tipku "Open" u donjem desnom kutu prozora što će otvoriti terminal i prikazati nam poruke koje naš firmware ispisuje.



Slika 6. Putty prozor sa potrebnim postavkama