

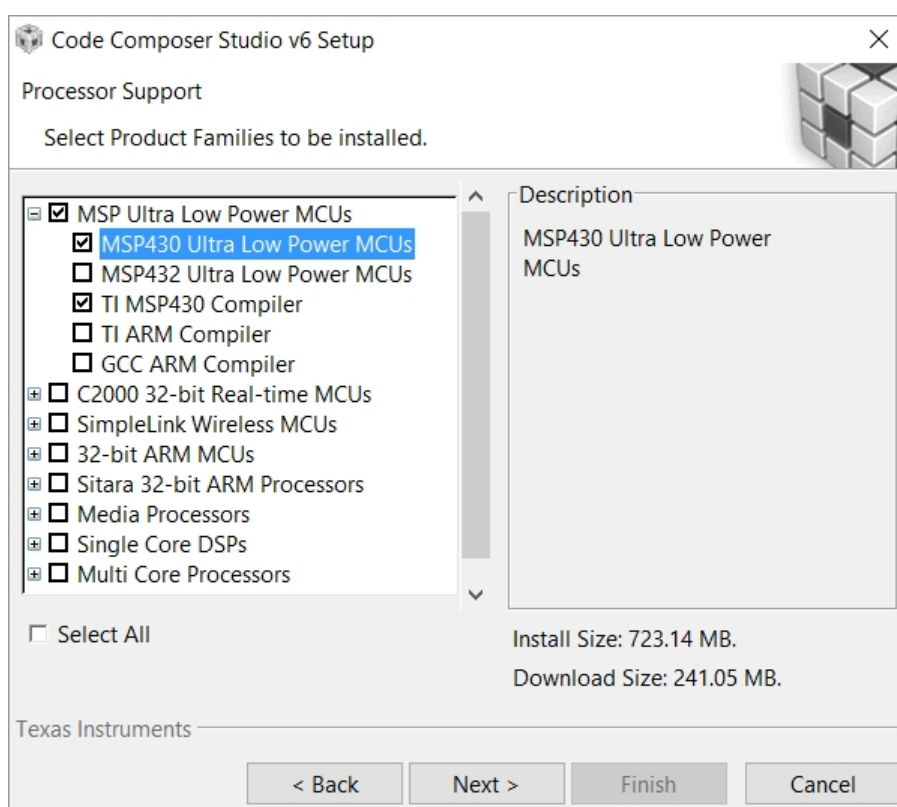
Ghid de utilizare a mediului de dezvoltare Code Composer Studio v.6 pentru MSP430

1. Instalare

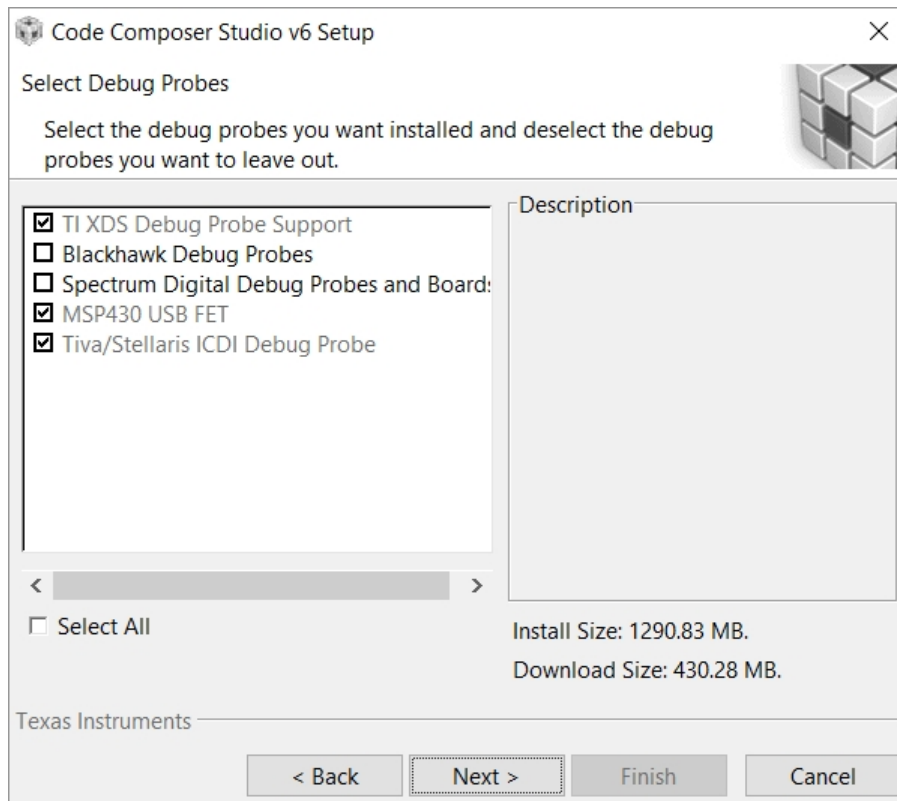
Cea mai recentă versiune a mediului de dezvoltare Code Composer Studio (CCS) poate fi descărcată de pe site-ul producătorului – Texas Instruments (TI) – de la adresa: <http://www.ti.com/tool/ccstudio>. În cele ce urmează se va discuta despre versiunile 6.x.

Se recomandă dezactivarea aplicației antivirus (dacă există pe sistemul pe care lucrați) pe durata instalării în special dacă aceasta exercită un control excesiv asupra proceselor executate. Lansați kit-ul de instalare descărcat pentru a porni procesul de instalare și urmați pașii inițiali privind acceptarea termenilor de utilizare și selecția locației de instalare.

În pasul dedicat selecției produselor ce urmează a fi instalate alegeți produsele destinate familiei MSP430. Figura următoare ilustrează selecția produselor necesare în acest pas.



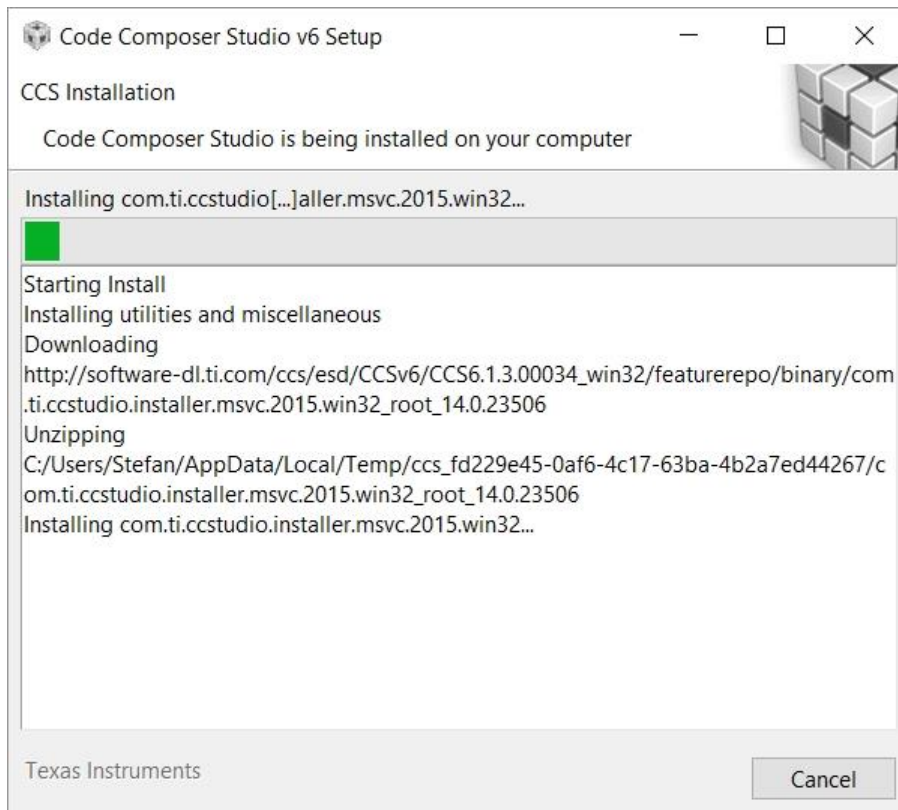
Pasul următor este dedicat alegerii driverelor necesare pentru interfețele de debug pe care le veți folosi. Asigurați-vă că opțiunea pentru **MSP430 USB FET** este selectată.



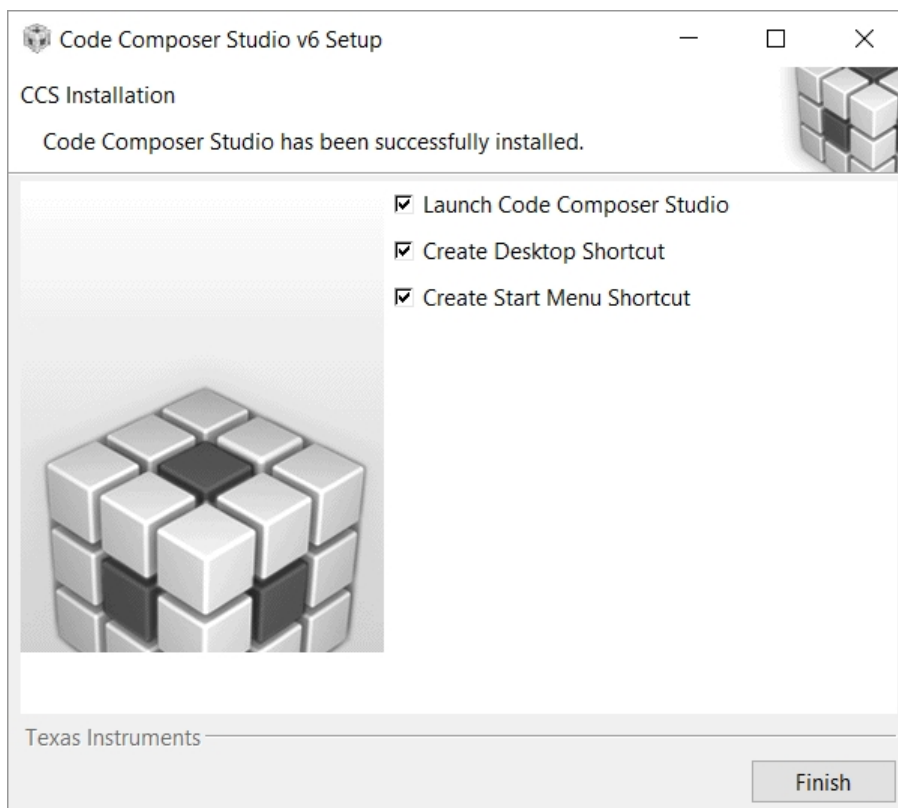
În continuare se pot alege produse suplimentare precum compilatorul GCC pentru MSP430. Alegerea acestui compilator este opțională, compilatorul oferit de Texas Instrument fiind suficient pentru îndeplinirea sarcinilor proiectului.



Instalarea propriu-zisă începe odată cu apăsarea butonului *Finish* și necesită o conexiune la internet pentru că implică descărcarea unor pachete de pe site-ul producătorului.

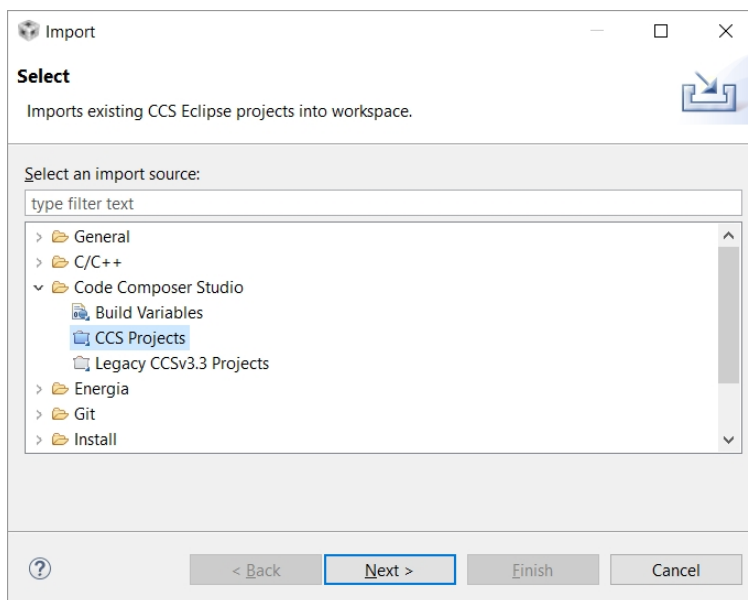


Fereastra urmatoare apare dupa finalizarea instalarii oferind posibilitatea de a crea scurtaturi si de a lansa Code Composer Studio.

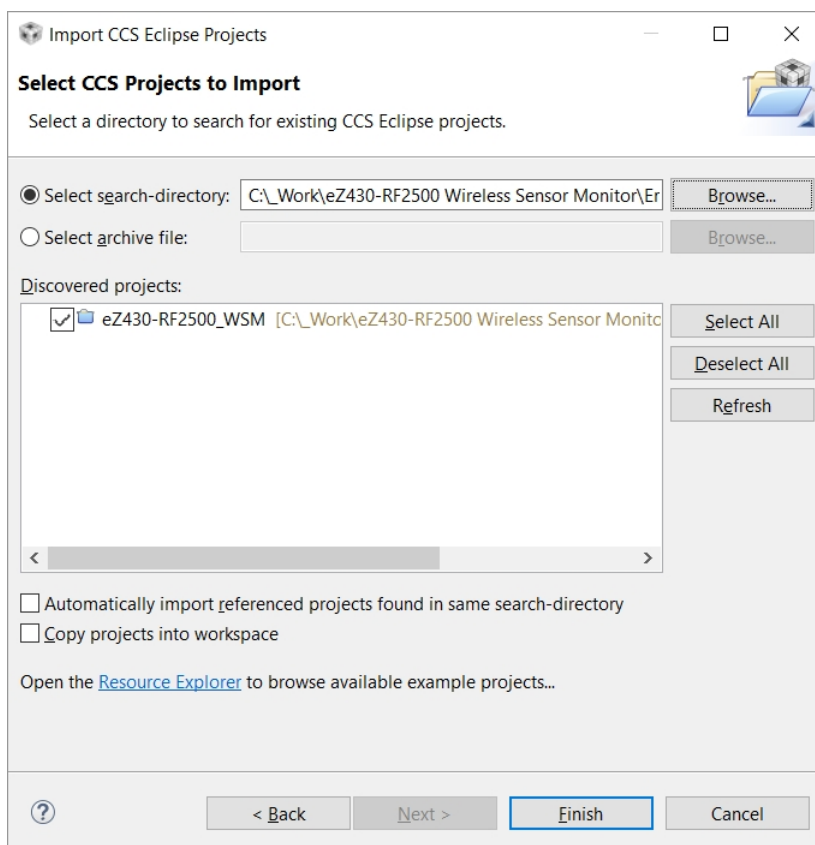


2. Importarea proiectelor

Pentru a importa un proiect existent în CCS6 selectați funcția *Import* din meniul *File*. Din fereastra nou deschisă selectați *Code Composer Studio > CCS Projects* și apăsați butonul *Next* ca în imaginea de mai jos.



În continuare selectați calea către proiectul deja existent ca în figura de mai jos și apăsați butonul *Finish*. Dacă se dorește copierea fișierelor proiectului în directorul current de lucru selectați *Copy projects into workspace*.

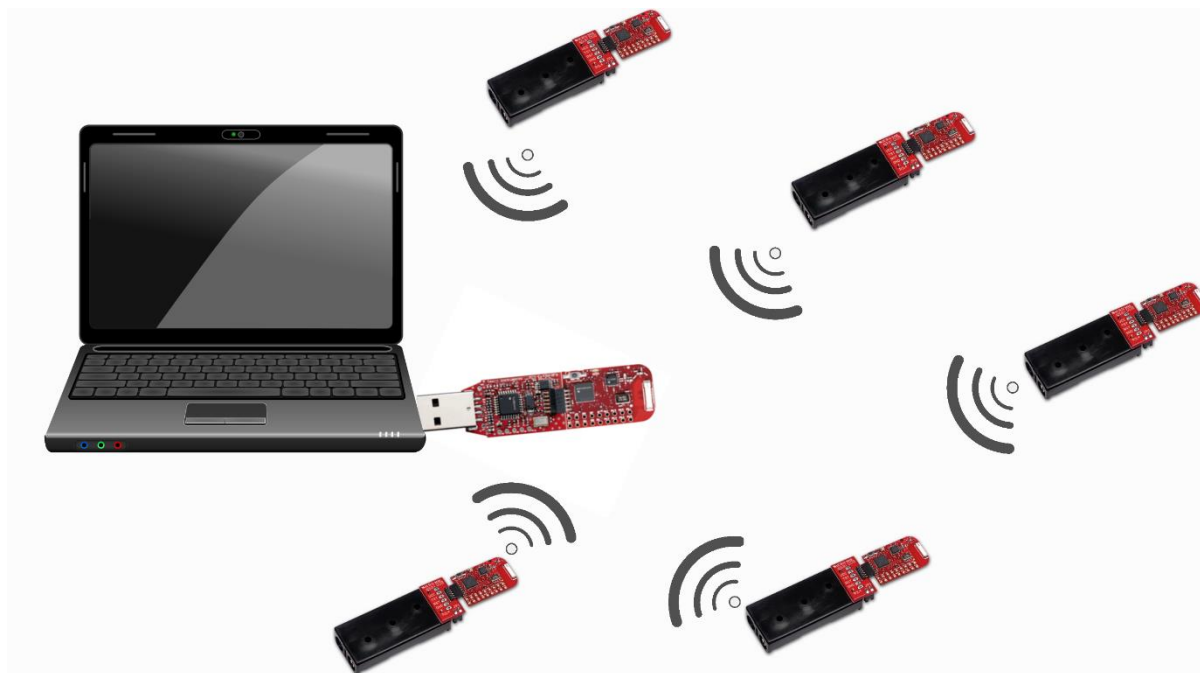


3. Utilizarea proiectului demo EZ430-RF2500

3.1. Descrierea proiectului demonstrativ

Pentru a ușura introducerea în dezvoltarea de aplicații folosind kit-ul EZ430-RF2500, TI oferă pe pagina dedicată produsului un exemplu de proiect ce poate fi implementat pe această platformă.

Exemplul de proiect deservește o aplicație cu o rețea de senzori ce monitorizează temperatura în diverse puncte ale perimetrului în care aceștia sunt plasați. Fiecare sensor măsoară periodic temperatura și o transmite către o unitate master conectată la un computer. Unitatea master transmite toate informațiile primite prin interfața serială către o aplicație ce afișează datele recepționate de la toți senzorii.



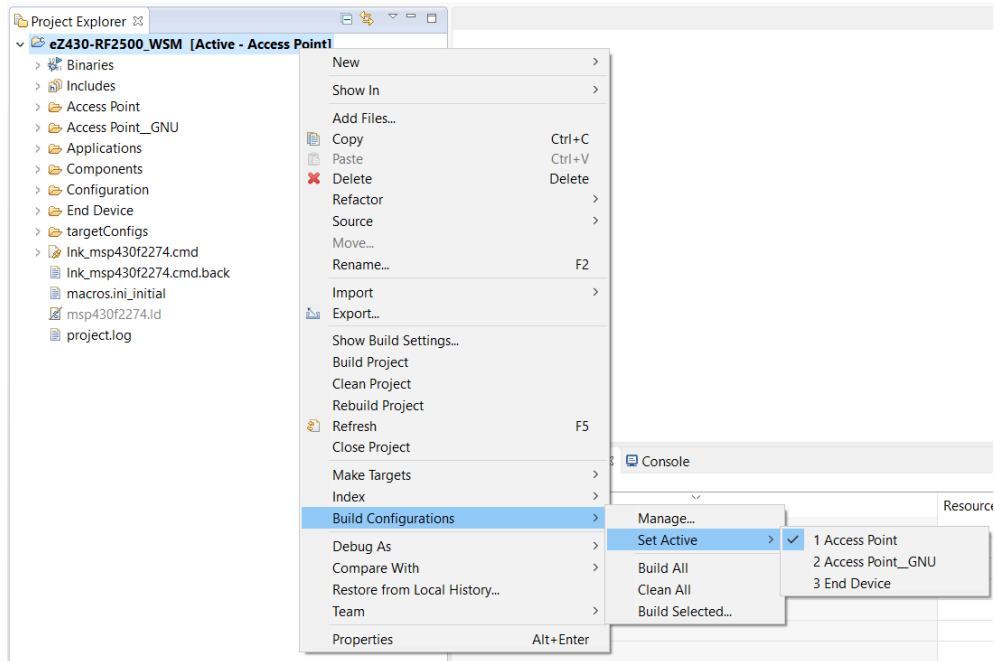
Imagina de mai sus ilustrează configurația sistemului deservit de aplicația demonstrativă. Sistemul este compus din trei componente SW:

- *Aplicația de monitorizare* ce rulează pe PC
- *Aplicația Access-point* ce rulează pe dispozitivul MSP430 master conectat la PC
- *Aplicația End-device* ce rulează pe dispozitivele MSP430 slave

3.2. Importarea și configurarea proiectului în CCS 6

Descărcați proiectul de pe pagina de produs EZ430-RF2500 (<http://www.ti.com/tool/ez430-rf2500#whatsIncluded>). Pachetul descărcat conține cele 3 componente SW amintite în secțiunea anterioară. Aplicația pentru PC se regăsește în folderul PC sub forma unui installer. Aplicațiile pentru microcontroller le găsim în folderul Embedded în două versiuni. O versiune este construită pentru CCS iar cealaltă pentru mediul de dezvoltare IAR Embedded Workbench.

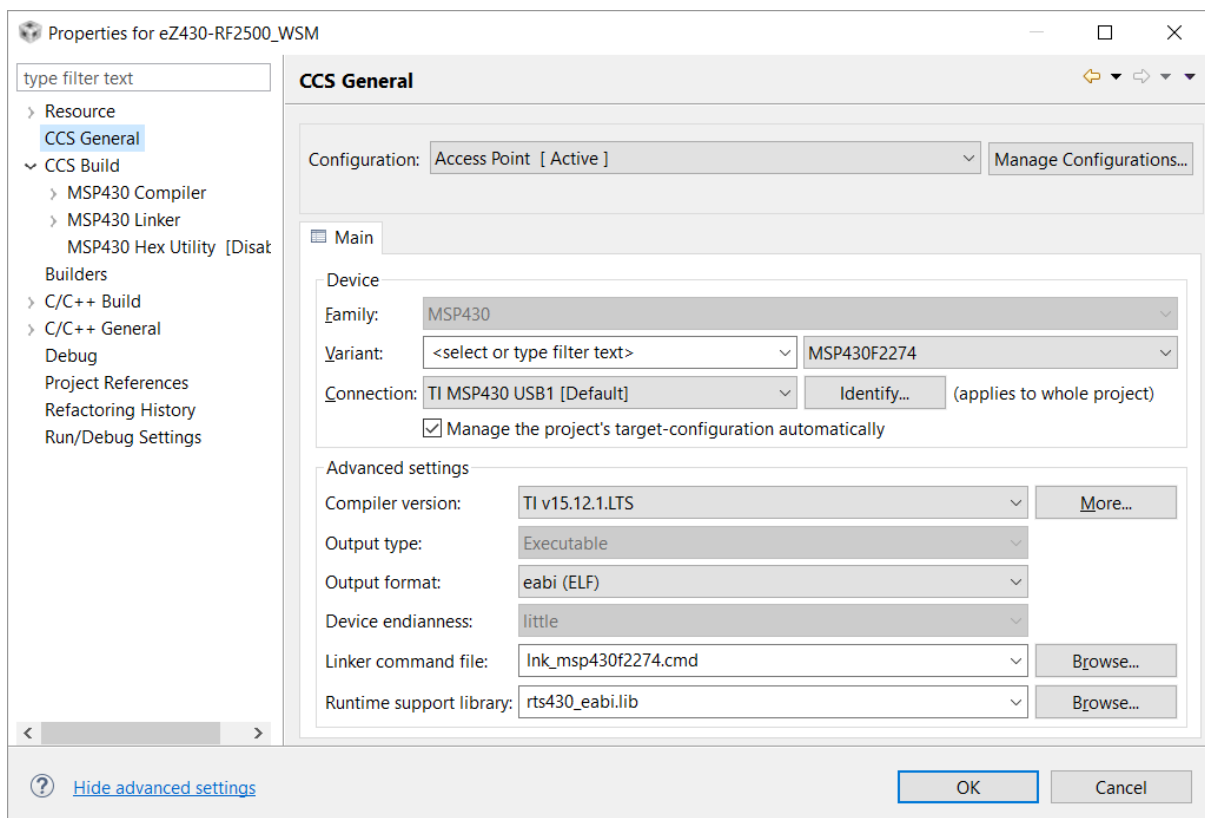
Extrageți conținutul pachetului descărcat și importați în CCS proiectul destinat acestui mediu de dezvoltare folosind pașii descriși în secțiunea 2. Proiectul conține atât codul sursă pentru componenta Access-point cât și cel pentru End-device. Pentru a selecta ca proiect activ unul din cele două faceți click dreapta pe titlul proiectului din secțiunea *Project Explorer* selectând mai apoi din meniul deschis opțiunea *Build Configurations > Set Active* și configurația dorită, ca în imaginea următoare.



Ca alternativă aceeași setare se poate face diin meniul *Project > Build Configurations > Set Active*.

Proiectul descărcat a fost construit cu o versiune anterioară de CCS. Pentru a putea fi utilizat cu versiunea curentă de CCS proiectul necesită adaptarea unor setări de compilator. În câmpul *Project Explorer* faceți click dreapta pe proiectul deschis si selectati optiunea properties.

În fereastra deschisa la sectiunea General asigurați-va ca aveți setările precum în imaginea următoare.



Astfel, conform imaginii setările din următoarele trei câmpuri trebuie modificate după cum urmează:

- *Compiler version* – versiunea compilatorului trebuie configurată ca versiunea cea mai recentă disponibilă a compilatorului Ti;
- *Output format* – formatul fișierului rezultat în urma compilării trebuie să fie *eabi (ELF)*;
- *Runtime support library*.- librăria utilizată este *rts430_eabi.lib*.

Aceste schimbări sunt necesare pentru ambele configurații ale proiectului. Pentru schimbarea proprietăților configurația poate fi selectată din câmpul *Configuration* din cadrul ferestrei de proprietăți. Asigurați-vă că adaptați schimbările menționate anterior pentru ambele configurații ale proiectului.

3.3. Compilare, linkeditare, programare și debug

Pentru a genera codul obiect pentru una din componente selectați-o drept configurație activă și faceți click pe iconița reprezentând un ciocan 🛠️. Ca alternativă puteți porni procesul din meniul *Project > Build Project*.

Adaptarea fișierului de linkeditare

În urma compilării și linkeditării vor fi generate 2 warninguri. Unul din ele se referă la crearea secțiunii “.data”, iar cel de-al doilea este legat de vectorul de întreruperi. Ambele warninguri se datorează diferențelor dintre comportamentul compilatorului pentru care a fost inițial configurat proiectul și cel al versiunii nou configurate. Nerezolvarea celor două warninguri duce la probleme în execuția codului. Mai exact problemele manifestate sunt blocarea aplicației End-Point în faza de configurare a stack-ului de comunicație și neincluderea rutinelor de tratare ale întreruperilor în codul obiect generat.

Pentru a rezolva problema înlocuiți fișierul de linkeditare aflat în directorul proiectului la calea relativă `\eZ430-RF2500_WSM\lnk_msp430f2274.cmd` cu cea mai nouă versiune a acestui fișier pe care o puteți găsi în directorul de instalare a CCS6: `ccsv6\ccs_base\msp430\include\lnk_msp430f2274.cmd`.



Pentru a vă asigura că folosiți o versiune adecvată a fișierului de linkeditare verificați dacă în zona “SECTIONS” este definită secțiunea „.data” precum este ilustrat în secvența de mai jos.

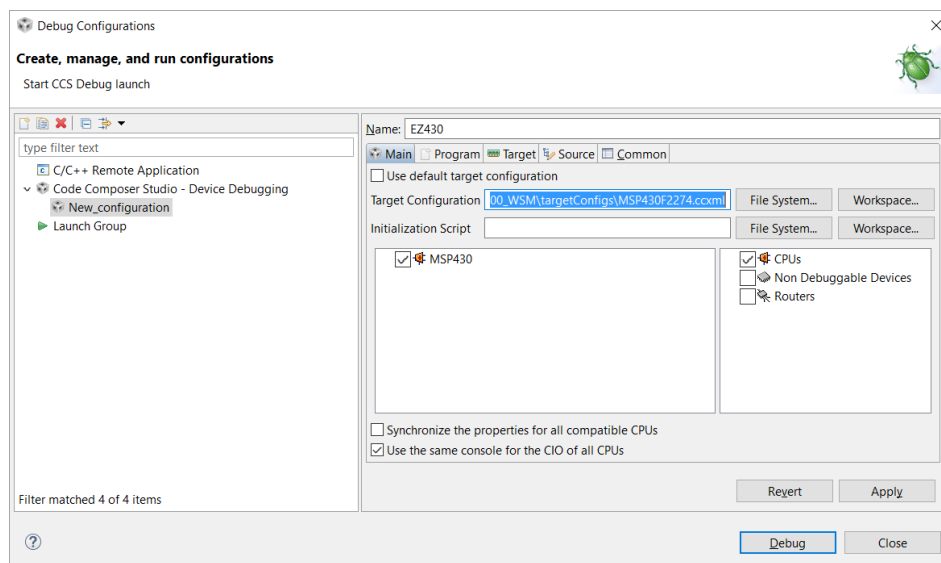
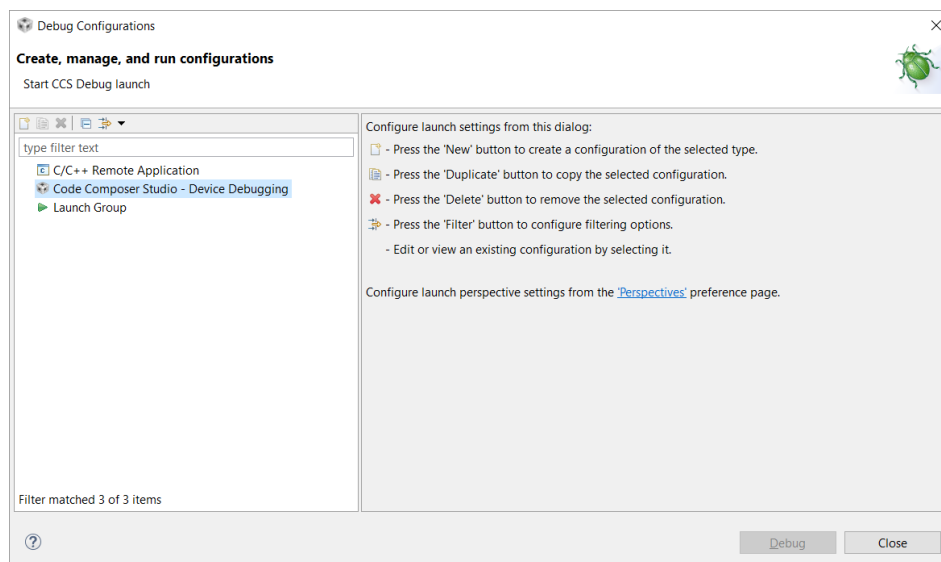
```
SECTIONS
{
    .bss      : {} > RAM           /* Global & static vars */
    .data     : {} > RAM           /* Global & static vars */
    .TI.noinit : {} > RAM           /* For #pragma noinit */
    .systemem : {} > RAM           /* Dynamic memory allocation area */
    .stack    : {} > RAM (HIGH)    /* Software system stack */
}
```

De asemenea trebuie să vă asigurați că fișierul de linkeditare conține definiții pentru vectorul de întreruperi precum în exemplul de mai jos.

```
/* MSP430 Interrupt vectors */
.int00      : {} > INT00
.int01      : {} > INT01
PORT1       : { * ( .int02 ) } > INT02 type = VECT_INIT
PORT2       : { * ( .int03 ) } > INT03 type = VECT_INIT
.int04      : {} > INT04
ADC10       : { * ( .int05 ) } > INT05 type = VECT_INIT
USCIAB0TX   : { * ( .int06 ) } > INT06 type = VECT_INIT
USCIAB0RX   : { * ( .int07 ) } > INT07 type = VECT_INIT
TIMERA1     : { * ( .int08 ) } > INT08 type = VECT_INIT
TIMERA0     : { * ( .int09 ) } > INT09 type = VECT_INIT
WDT         : { * ( .int10 ) } > INT10 type = VECT_INIT
.int11      : {} > INT11
TIMERB1     : { * ( .int12 ) } > INT12 type = VECT_INIT
TIMERB0     : { * ( .int13 ) } > INT13 type = VECT_INIT
NMI         : { * ( .int14 ) } > INT14 type = VECT_INIT
.reset      : {} > RESET /* MSP430 Reset vector */
```

Programare

Pentru a programa aplicația pe dispozitivul embedded și pentru a face debug trebuie să creați o configurație de debug. Deschideți fereastra de creare și editare a configurațiilor de debug din meniul *Run > Debug Configurations...* sau din submeniul iconiței . Pentru a crea o configurație de debug selectați din lista opțiunea *Code Composer Studio – Device Debugging* și apăsați butonul *New* .



Introduceți un nume relevant pentru configurația nou creată în câmpul *Name* și selectați un fișier pentru câmpul *Target Configuration*. Fișierul de configurare pentru dispozitivul țintă îl veți găsi între fișierele proiectului urmând calea relativă `\\Embedded\\CCS\\eZ430-RF2500_WSM\\targetConfigs\\MSP430F2274.ccxml`. Apăsați butonul *Apply* pentru a salva schimbările efectuate.

Pentru a lansa sesiunea de debug apăsați butonul *Debug* din fereastra *Debug Configurations* sau selectați *Debug* din meniul *Run*. Procesul pornit astfel începe cu stabilirea conexiunii cu dispozitivul embedded și programarea microcontrollerului cu aplicația generată prin procesul de build.