ARHITECTURA MICROCONTROLERELOR XMC 4500

1. Descriere generală

Seria **XMC 4500** face parte din familia de microcontrolere industriale XMC 4000 bazate pe procesorul ARM Cortex-M4. Aceste dispozitive sunt optimizate pentru controlul motoarelor electrice, conversie de putere, conectivitate industrială, și automatizări.

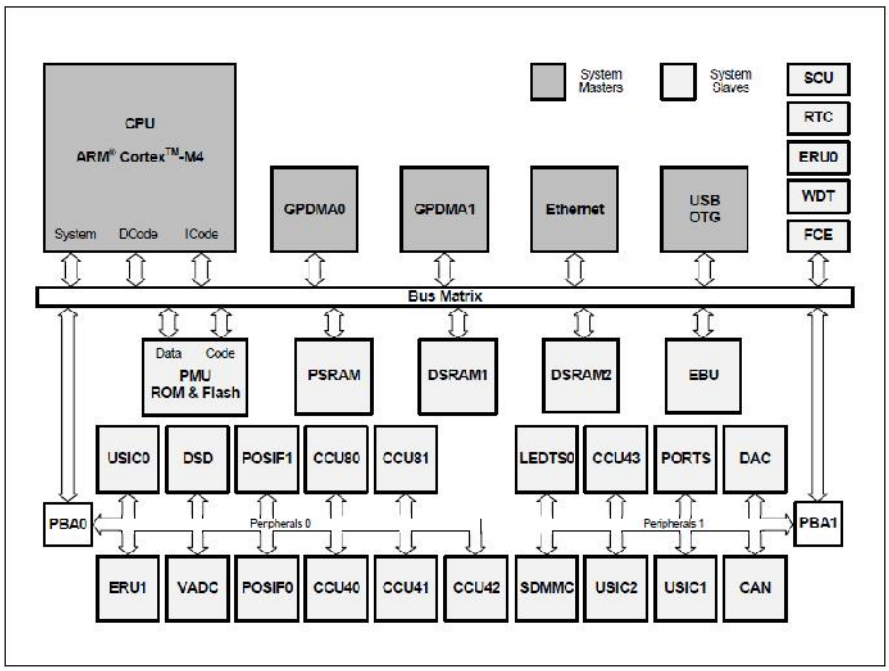

Figura 1. Blocurile funcționale și conectarea lor în sistemul XMC 4500

1.1 Subsistemul CPU

- Nucleul procesorului:
  - processor ARM Cortex-M4 pe 32 de biți;
  - set de instrucțiuni pe 16 sau 32 de biți;
  - instrucțiuni DSP/MAC;
  - timer de sistem pentru suport de sistem de operare;
- Unitate de virgulă mobilă;
- Unitate de protecție a memoriei;
- Controler de întreruperi înlănțuite;
- Două blocuri de transfer DMA;
- Unitate de cerere a evenimentelor (pentru servicii interne sau externe);
- Bloc de detecție a erorilor multiple (CRC).

1.2 Memorii on chip

- 16 ko ROM (boot);
- 64 ko memorie de program de mare viteză;

- 64 ko memorie de date de mare viteză;
- 32 ko memorie de mare viteză pentru comunicație;
- 1024 ko memorie flash cu 4 ko memorie cache.

1.3 Dispozitive periferice pentru comunicație

- modul Ethernet 10/100 Mbit;
- modul USB;
- interfață CAN (Controller Area Network);
- 6 interfețe seriale (configurabile în diferite standarde seriale);
- interfață pentru comunicarea om-mașina (LED și touch);
- interfață pentru carduri de memorie externă (SD și SDMMC);
- bus extern pentru conectarea unor memorii externe.

1.4 Periferice pentru semnale analogice

- 4 convertoare ADC pe 12 biți cu câte 8 canale fiecare;
- Demodulator Sigma Delta cu 4 canale;
- 1 convertor DAC pe 12 biți cu 2 canale.

1.5 Periferice pentru control industrial

- 2 unități de captură și comparare pentru controlul motoarelor;
- 4 unități de captură și comparare folosite ca timere de uz general;
- 2 interfețe de determinare a poziției;
- timer de tip watchdog;
- senzori de temperatură;
- ceas de timp real;
- unitate de control a sistemului.

1.6 Linii de intrare – ieșire

- modul pentru porturi programabile;
- intrări tri-state;
- interfață de test JTAG (Joint Test Action Group);
- suport pentru depanare.

2. Arhitectura procesorului ARM Cortex-M4

2.1 Caracteristici generale

Procesorul ARM Cortex-M4 este construit pe un nucleu de procesor foarte performant, bazat pe o structură pipe-line cu 3 stagii, cu arhitectură Harvard, fiind ideal pentru aplicații embedded cu cerințe ridicate de procesare.

Datorită unui set de instrucțiuni eficient și a unui design optimizat pentru aplicații de control eficiente energetic, procesorul este extrem de eficient din punct de vedere al consumului de putere.

Pentru a face față cerințelor de complexitate crescută a sistemelor de control embedded, procesorul permite prelucrări în virgulă fixă într-un singur ciclu, operații SIMD (Single Instruction Multiply Data) de înmulțire și înmulțire cu acumulare, logică de saturare, și operații de împărțire hardware.

Pentru creșterea vitezei și a densității codului, i s-a implementat un set de instrucțiuni simplificat și comprimat bazat pe tehnologia Thumb-2.

Procesorul implementează și un sistem de întreruperi cu priorități multiple, folosind 122 noduri de întreruperi și 64 de nivele de prioritate. [18]

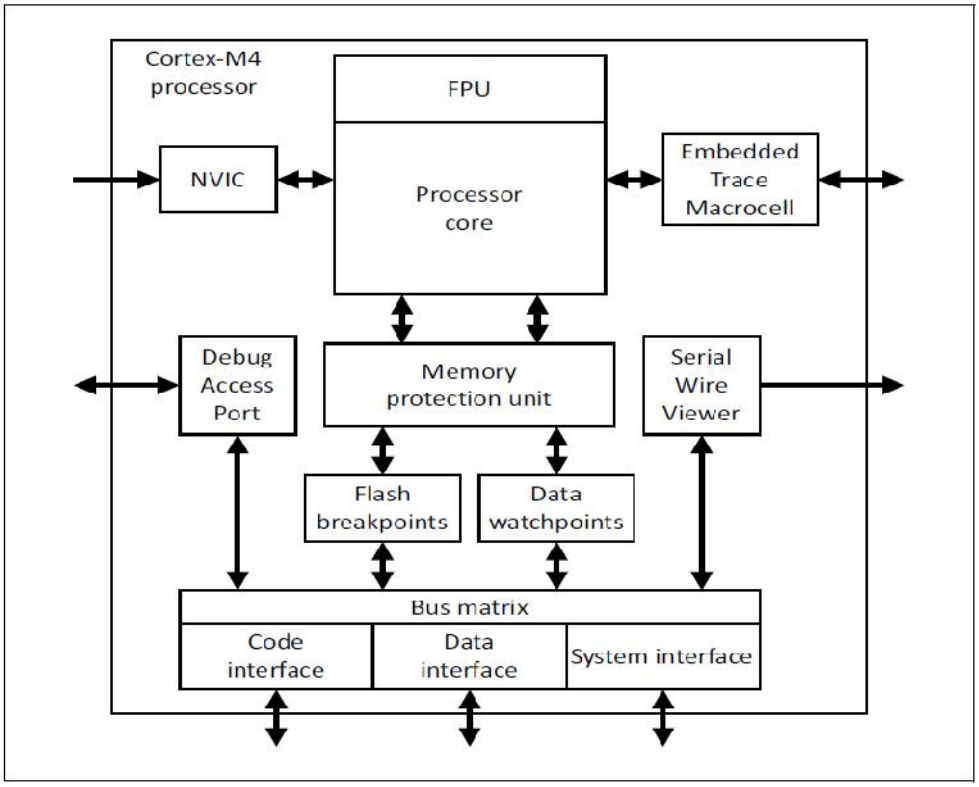

Figura 2. Blocurile funcționale ale procesorului ARM Cortex-M4

Blocurile funcționale ale procesorului ARM Cortex-M4 sunt prezentate în figura 2:

- Nucleul procesorului (Processor Core);
- Unitatea de virgulă mobilă (Floating Point Unit);
- Controlerul de întreruperi (Nested Vectored Interrupt Controler);
- Unitatea de protecție a memoriei (Memory Protection Unit).

2.2.2 Modelul de programare

Procesorul ARM Cortex-M4 prezintă următoarele moduri de lucru :

- modul **thread** – se execută un program din aplicație. În acest mod se intră imediat după reset;
- modul **handler** – se tratează excepții. După tratarea unei excepții, se revine în modul **thread**.

Programele se execută pe 2 nivele de privilegiere:

- nivelul **neprivilegiat** – programele nu pot executa anumite instrucțiuni de acces la registre speciale, nu se poate folosi timer-ul de sistem și blocul de control al sistemului, pot avea acces restricționat la memorie;
- nivelul **privilegiat** – se intră pe acest nivel din modul **thread** dacă actualizează adecvat un registru de control. Modul **handler** este întotdeauna pe nivelul **privilegiat**.

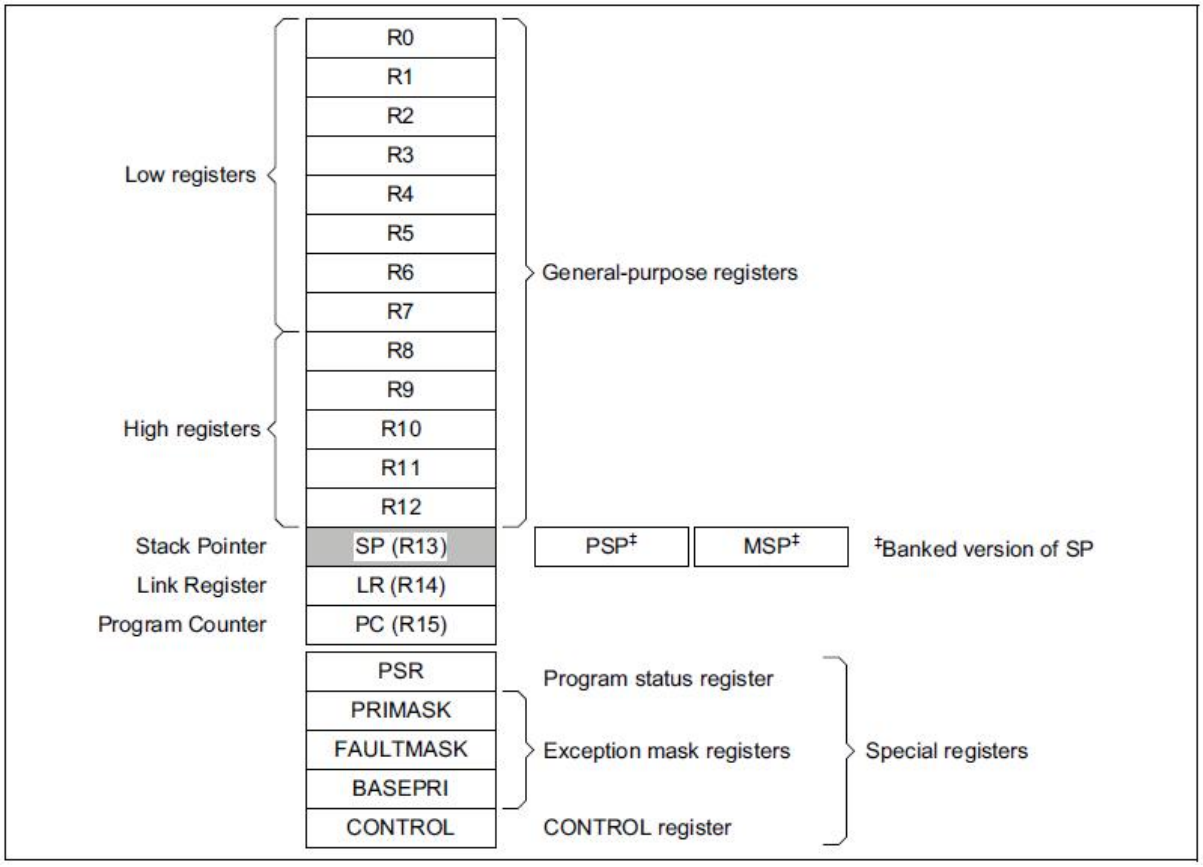

Figura 3. Registrele procesorului ARM Cortex-M4; Sursa: [21]

În figura 3 sunt prezentate toate registrele procesorului ARM Cortex-M4. Acestea se împart în:

- registre de date (registre generale) pe 32 de biți;
- registre pentru controlul programului;
- registre pentru controlul stivei;
- registre speciale, care indică starea procesului, după cum urmează:
  - **PSR** - Program Status Register
  - **ASPR** - Application Program Status
  - **IPSR** - Interrupt Program Status
  - **EPSR** - Execution Program Status
  - **PRIMASK** - Priority Mask Register on
  - **FAULTMASK** - Fault Mask Register
  - **BASEPRI** - Base Priority Mask Register
  - **CONTROL** - CONTROL register

3. Modelul de memorie

Procesorul are o hartă a memoriei fixată implicit care oferă până la 4Go de memorie adresabilă. Această hartă a memoriei este ilustrată în figura 4.

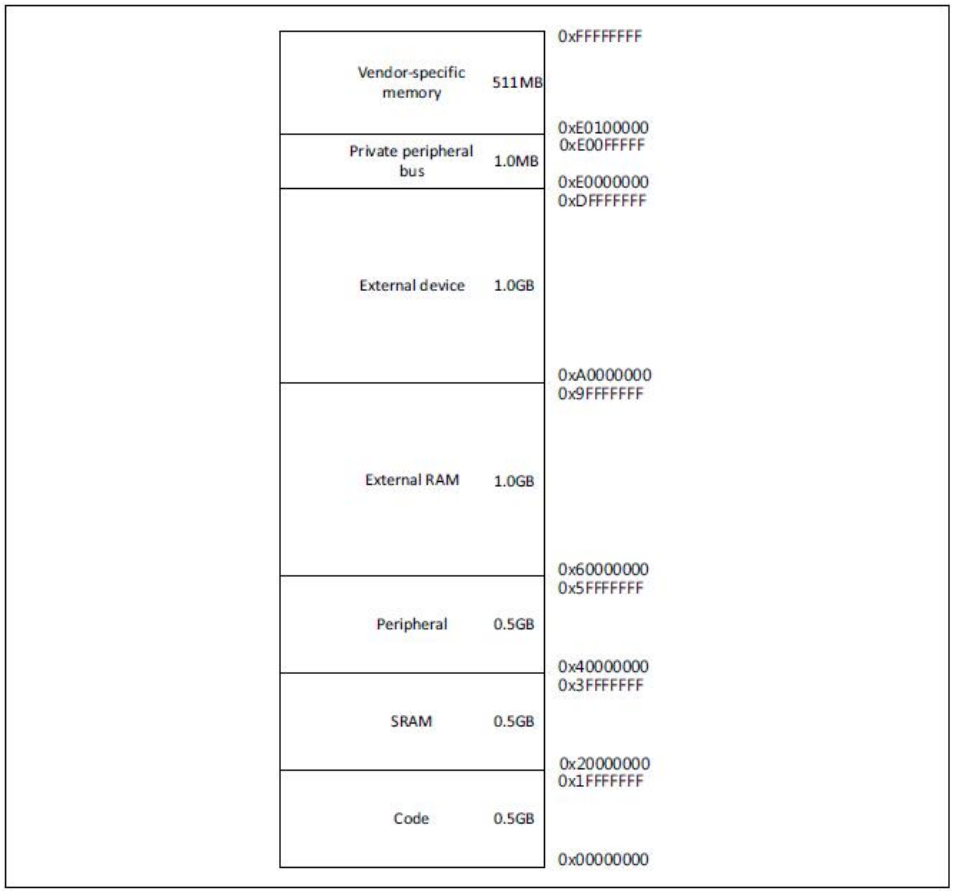

Figura 4 Harta memoriei

Există zone de memorie rezervate pentru cod, date, și dispozitive de intrare – ieșire. Prin modul de programare al blocului MPU fiecare zonă de memorie poate avea atribute specifice:

- **Normal** – se pot reordona accesele la memorie pentru creşterea eficienţei sau executa citiri speculative din memorie;
- **Device** – se conservă ordinea acceselor la memorie, relativ la accese de tip **Device** sau **Strongly-ordered**;
- **Strongly-ordered** – se conservă ordinea acceselor la memorie, relativ la toate tipurile de acces;
- **Execute never** – nu se pot accesa instrucţiuni.

4 Interfaţa Ethernet

Interfaţa Ethernet MAC (ETH) este un periferic ce suportă transfer de date la rate de 10/100 MBit/s în conformitate cu standardul IEEE 802.3-2002. ETH poate fi utilizat pentru a implementa aplicaţii conectate la internet prin IPv4 şi IPv6. ETH oferă suport şi pentru IEEE 1588, sincronizare temporală, pentru a putea implementa protocoale Ethernet în timp real.

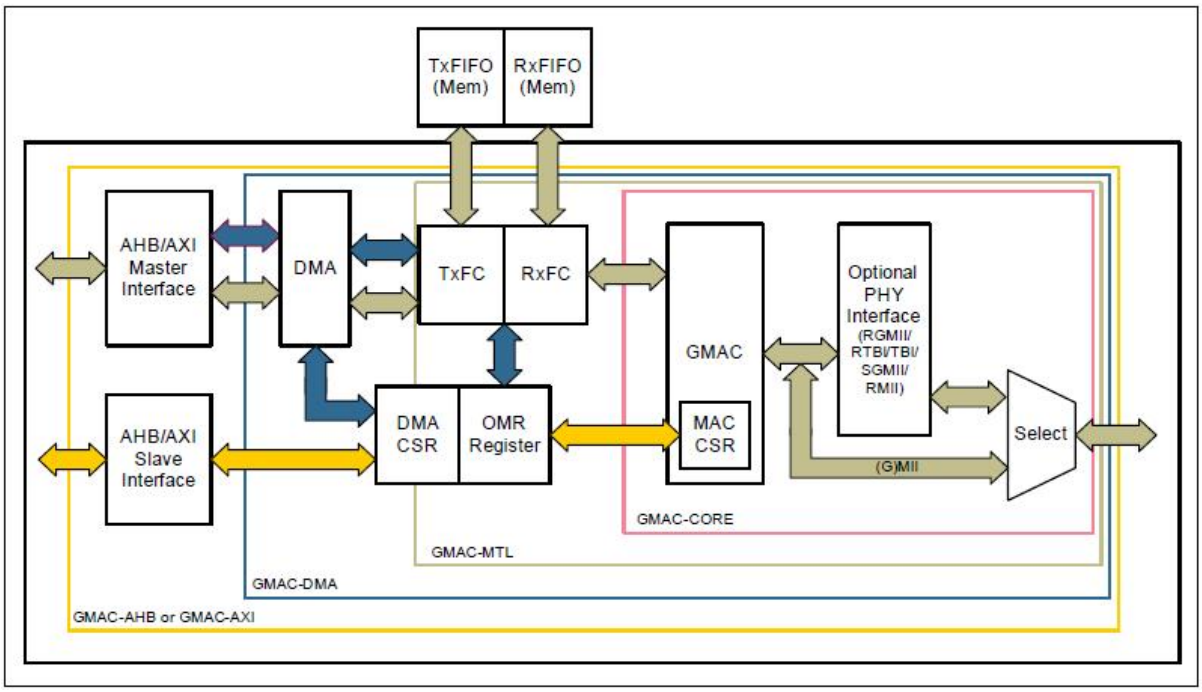

Figura 5. Schema bloc ETH

Perifericul ETH este compus din următoarele unităţi funcţionale majore:

- **Nucleul ETH (ETH Core)** – primeşte pachete de date de la utilizator şi le trimite către interfaţa nivelului fizic (PHY) printr-o interfaţă de tip MII (Media Independent Interface) sau RMII (Reduced Media Independent Interface);
- **Nivelul de tranzacţie ETH MAC (MTL)** – acţionează ca o punte între aplicaţie şi nucleul ETH; MTL pune la dispoziţie 2 cozi FIFO de câte 2ko fiecare, folosite ca buffer pentru cadrele trimise şi recepţionate. Aplicaţia poate scrie cadrele direct către MTL (modul direct) sau, mai normal, va folosi unitatea embedded ETH DMA;

- **Unitatea ETH DMA** – permite aplicației să definească o regiune din RAM ca buffer de transmisie și recepție. Transferurile DMA sunt inițiate de descriptori DMA care se găsesc tot în RAM;
- **Modulul timpului de sistem** – permite marcarea timpului pe cadrele transmise și recepționate;
- **Setul extins de countere de management al MAC** – oferă statistici detaliate pentru bus.

5 Placa de evaluare XMC 4500 Relax Kit-V1

XMC 4500 Relax Kit-V1 este concepută și dezvoltată pentru a evalua capabilitățile microcontrolerului **XMC 4500** și a mediului de dezvoltare **Dave**. XMC 4500 Relax Kit-V1 extinde capabilitățile microcontrolerului prin posibilitatea rulării unui web server embedded. Se pot stoca pagini web HTML pe un card microSD sau se poate controla microcontrolerul prin intermediul browserului web de pe PC.

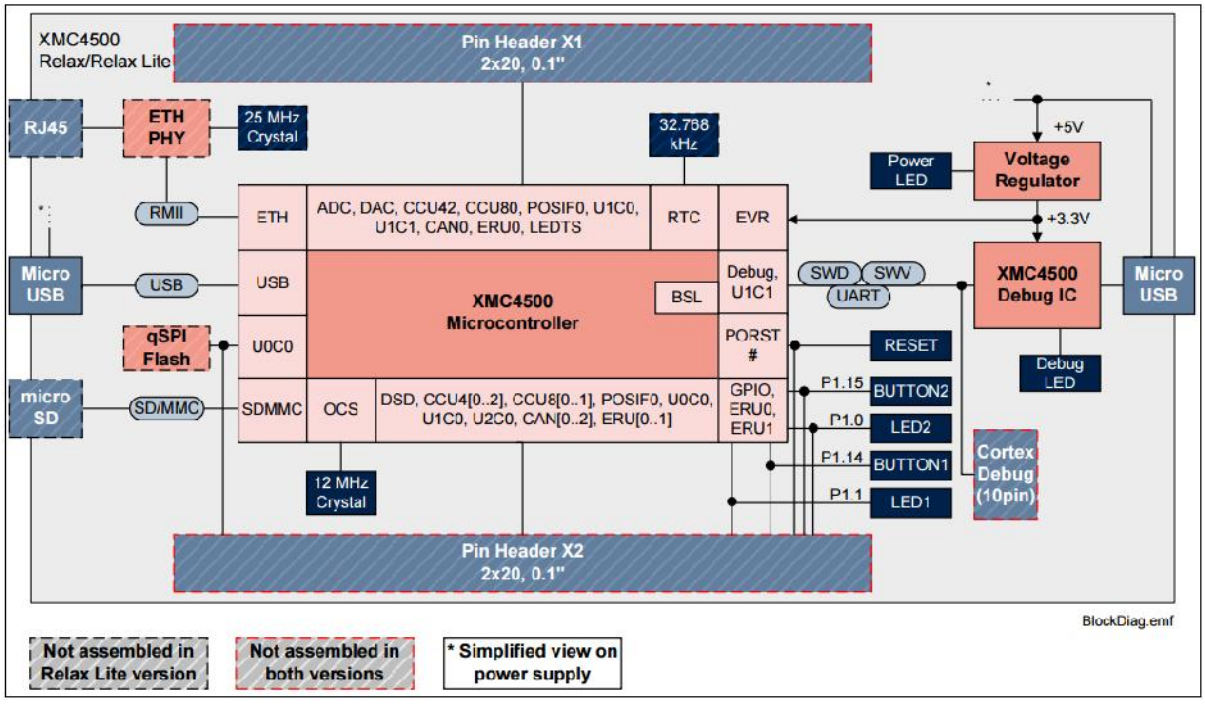

Figura 6 Schema bloc a plăcii XMC 4500 Relax Kit-V1

XMC 4500 Relax Kit-V1 este constituită din următoarele blocuri funcționale:

- Microcontrolerul XMC 4500;
- Blocul de depanare (un al doilea microcontroler XMC 4500 și interfața USB);
- Interfața Ethernet;
- Două conectoare de 40 de pini (pentru conectare cu alte echipamente);
- Sursa de alimentare;

- Două butoane (conectate pe portul P1, biții 14 și 15) și două LED-uri (conectate pe portul P1, biții 1 și 0) pentru utilizator;
- Interfața USB;
- Slot microSD.

PROTOCOLUL LWIP

1. Descriere generală

LwIP (lightweight) este o stivă TCP/IP open source foarte folosită, concepută pentru sisteme embedded. LwIP a fost dezvoltat inițial de Adam Dunkels la Swedish Institute of Computer Science, iar în momentul de față o vastă rețea mondială de dezvoltatori se ocupă de dezvoltare și întreținere.

LwIP este folosit de foarte mulți producători de sisteme embedded, cum ar fi: Altera (în sistemul de operare Nios II), Analog Devices (pentru chip-ul Blackfin DSP), Xilinx, Honeywell (în cadrul unora din sistemele lor de avionică certificate FAA), Freescale Semiconductor (software de Ethernet Streaming pentru microcontrolerele automotive), și nu în ultimul rând Infineon Technologies.

Scopul implementării LwIP TCP/IP este de a reduce folosirea de resurse, cu avantajul de a dispune în continuare de TCP în totalitate. Acest lucru face LwIP avantajos pentru sisteme embedded cu doar câteva zeci de kilooctețiliberi RAM și spațiu de câteva zeci de kilooctețiROM pentru cod[30].

Aplicațiile embedded au început să prezinte interes datorită îmbunătățirii considerabile a performanțelor procesoarelor actuale. O astfel de aplicație o reprezintă rețelele de senzori. În Fig. 1 este reprezentată schema bloc a unui nod de rețea de senzori embedded.

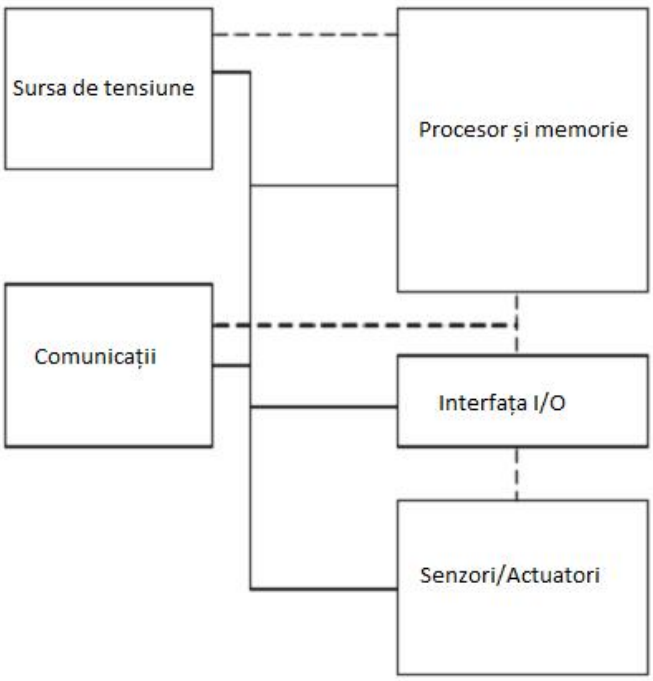

Figura 1 Structura nodului ESN

O rețea de senzori este utilizată pentru a monitoriza evenimente fizice. Aceste evenimente vor fi recepționate de utilizatori îndepărtați conectați la internet conform Fig. 2.

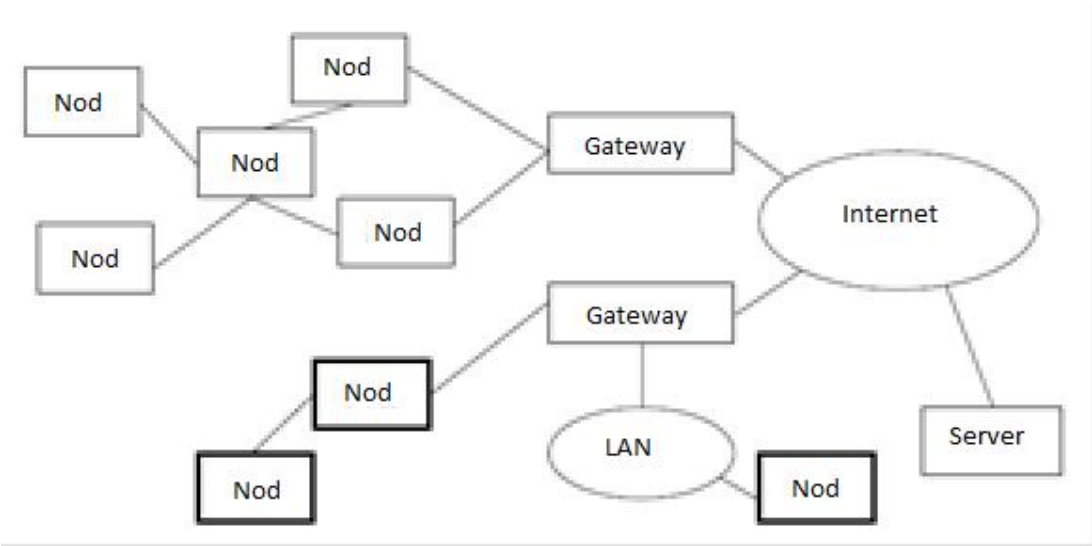

Figura 2. Exemplu de arhitectură a unei rețele de senzori

Nodurile ESN pot fi conectate într-o rețea individuală (rețea de senzori wireless – WSN) printr-un protocol specific. La marginea acetei rețele individuale se poate insera un nod (un computer personal) care să aibă rolul de gateway către o rețea obișnuită precum rețelele TCP/IP. Nodurile ESN nu sunt conectate direct la internet.

S-a dorit ca fiecare nod ESN să se poată conecta direct la internet folosind protocolul TCP/IP, în locul conectării printr-un gateway. În acest fel, un utilizator îndepărtat are acces mai rapid la nodurile ESN.

Datorită resurselor reduse ale nodurilor ESN, este imposibil să se ruleze implementări din Linux sau Windows ale TCP/IP pe aceste sisteme. Problema principală este reprezentată de constrângerile de memorie. Soluția găsită a fost o implementare mai simplă a stivei TCP/IP, care să poată rula pe sisteme cu memorie mică. Această implementare poartă numele de Lightweight IP.

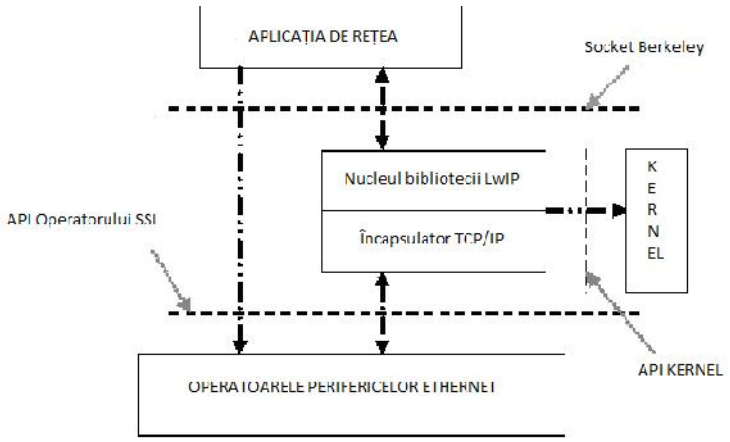

Figura 3. Schema arhitecturii LwIP

Stiva LwIP este dezvoltată cu ajutorul unui sistem de operare. Limitările de resurse și caracteristicile aplicațiilor rețelelor de senzori embedded impun cerințe specifice sistemului de operare care rulează pe nodurile rețelei, în special în privința managementului eficient al memoriei și a programării proceselor. [30]

2 Caracteristicile LwIP

Stratul Internet:

- IP – inclusiv expedierea de pachete prin mai multe interfețe de rețea;
- ICMP – întreținerea şi depanarea rețelei;
- IGMP – pentru gestionarea traficului multicast.

Stratul Transport :

- UDP – inclusiv extensii experimentale UDP-lite;
- TCP – controlul congestiei, estimarea RTT, şi recuperare rapidă / retransmitere rapidă.

Stratul Aplicație :

- DNS;
- SNMP;
- DHCP.

Stratul legătură de date :

- PPP;
- ARP.

MEDIUL DE DEZVOLTARE DAVE IDE

1 Descriere generală

Mediul de dezvoltare DAVE este o platformă de dezvoltare software gratuită şi foarte performantă de generare de cod folosind aplicaţii predefinite bazată pe folosirea unor aplicaţii predefinite – **DAVE Apps**.

DAVE Apps sunt aplicaţii orientate software care acoperă o arie mare de aplicabilitate cum ar fi generarea semnalelor PWM, conversia analog digitală şi altele. Biblioteca DAVE Apps abstractizează orice aplicaţie şi poate fi privită ca o clasă care permite instanţierea sa de fiecare dată când este necesar pentru a crea biblioteca necesară. Un rezolvitor al resurselor hardware supraveghează folosirea resurselor chip-ului pentru a evita orice tip de conflict hardware. Odată ce biblioteca a fost creată, utilizatorul poate controla perifericele prin intermediul unor interfeţe bine definite.

1. Caracteristici generale

DAVE include:

- Un IDE bazat pe Eclipse CDT;
- Unelte de compilare GNU C;
- Depanator;
- Un plug-in de generare a codului (Code Engine) cu interfaţă grafică;
- Un rezolvitor de resurse care administrează folosirea resurselor chip-ului;
- Un manager al bibliotecilor pentru a downloada şi folosi DAVE Apps.

2. Configurarea mediului de dezvoltare

În urma instalării mediului de dezvoltare, urmează configurarea acestuia. În acest scop se efectuează următorii paşi:

- Se lansează mediul de dezvoltare în execuţie;
- În fereastra de selecţie a spaţiului de lucru se poate da calea directă sau folosind butonul **Browse** putem obţine calea către spaţiul de lucru dorit. Apoi se face click pe butonul **Ok**;
- Se deschide spaţiul de lucru.

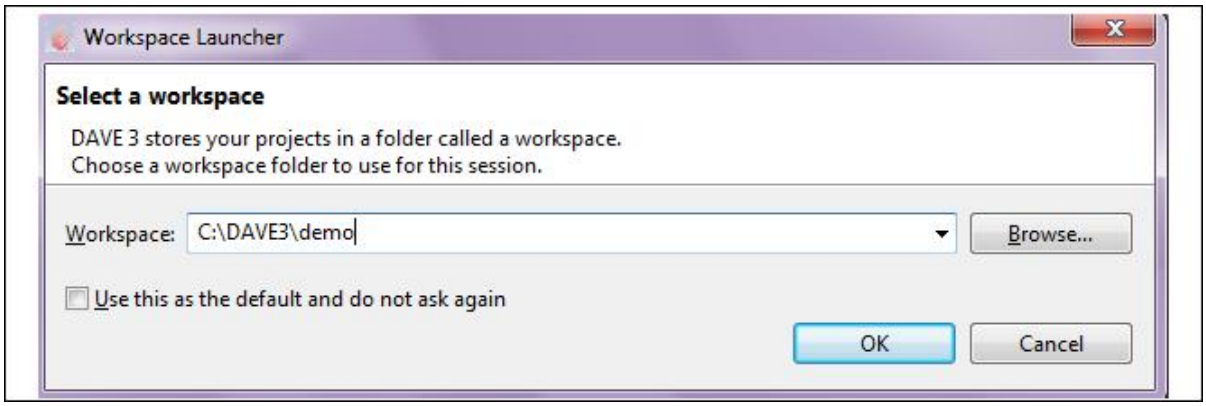

Figura 1. Fereastra de selecție a spațiului de lucru

- Urmează instalarea DAVE AppLibrary, apăsând pe butonul **DAVE AppLibrary Menu**;

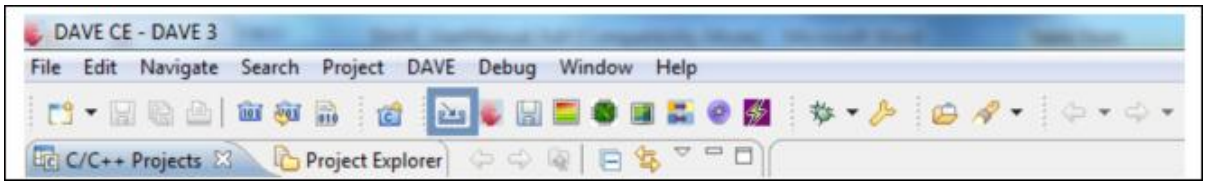

Figura 2 Butonul DAVE AppLibrary Menu

- Se face click pe **Install Apps** și apare fereastra **Download Libraries Page**;

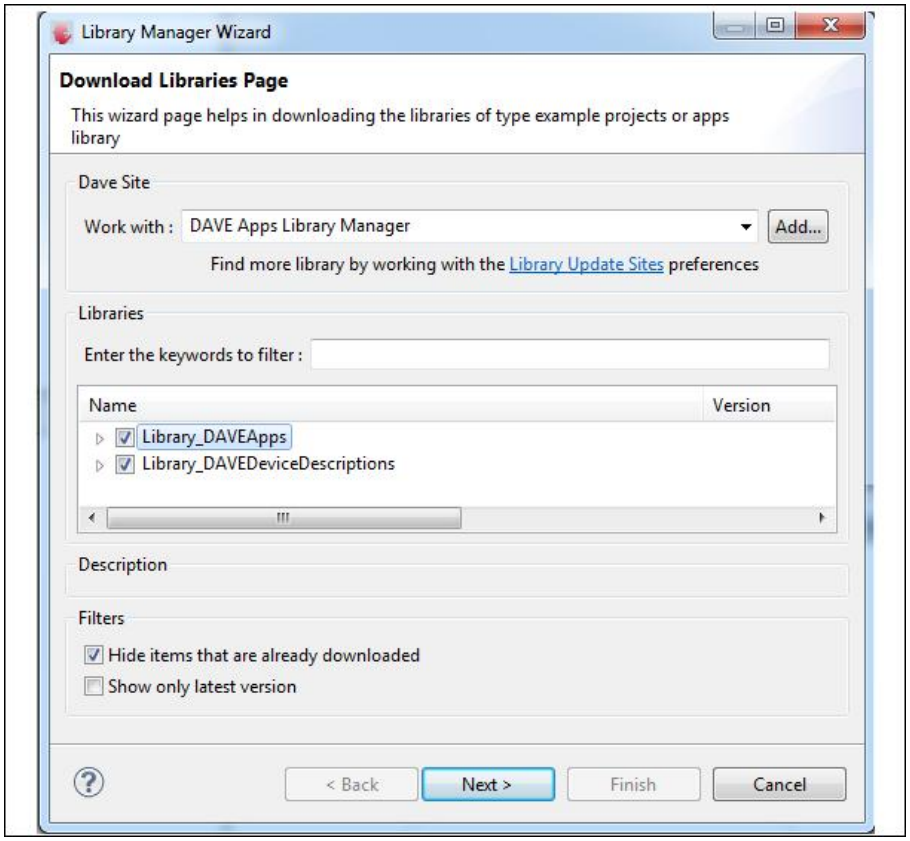

Figura 3. Fereastra Download Libraries Page

- Se face click pe **Add** și se introduce calea;
- Se bifează bibliotecile;
- Se face click pe **Finish**.

În urma acestor pași DAVE va cere un restart și este recomandabil să fie lăsat să treacă prin acest pas prin apăsare butonului **Ok** pentru siguranța unei instalări corecte.

3 Crearea unui proiect nou

Pentru a crea un proiect nou în mediul DAVE se execută următorii pași:

- Se face click pe **File** > **New** > **DAVE Project**;
- Se introduce un nume pentru proiect, spre exemplu **Demo**;
- Se selectează

Infineon XMC > ARM-GCC Application for XMC Project > DAVE CE Project;

- Se face click pe **Next**;

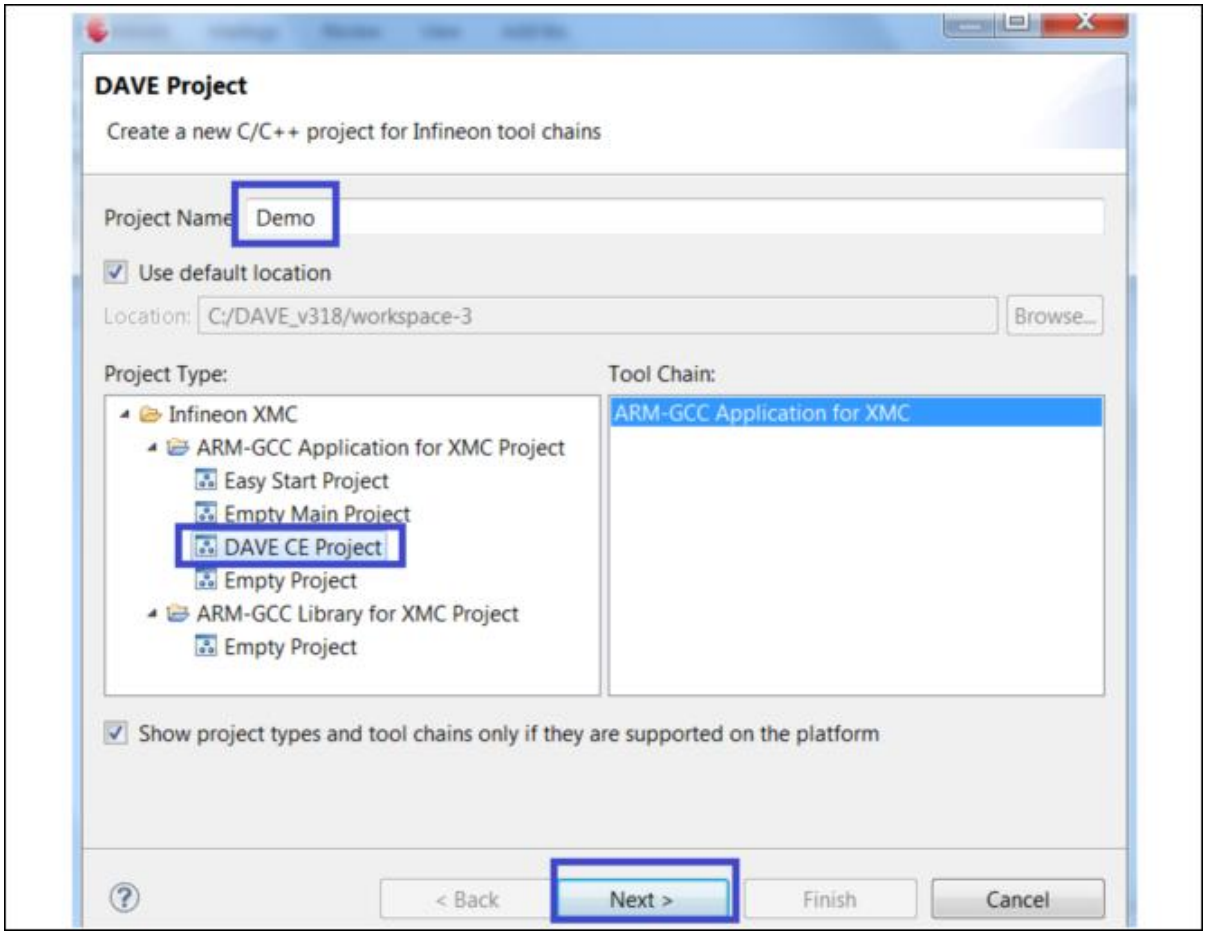

Figura 4.4 Fereastra DAVE Project; Sursa: [32]

- Se selectează **XMC 4500 series - F100x1024** > **Stepping Code AA** din fereastra **Target Selection Page** și se finalizează prin click pe **Finish**;

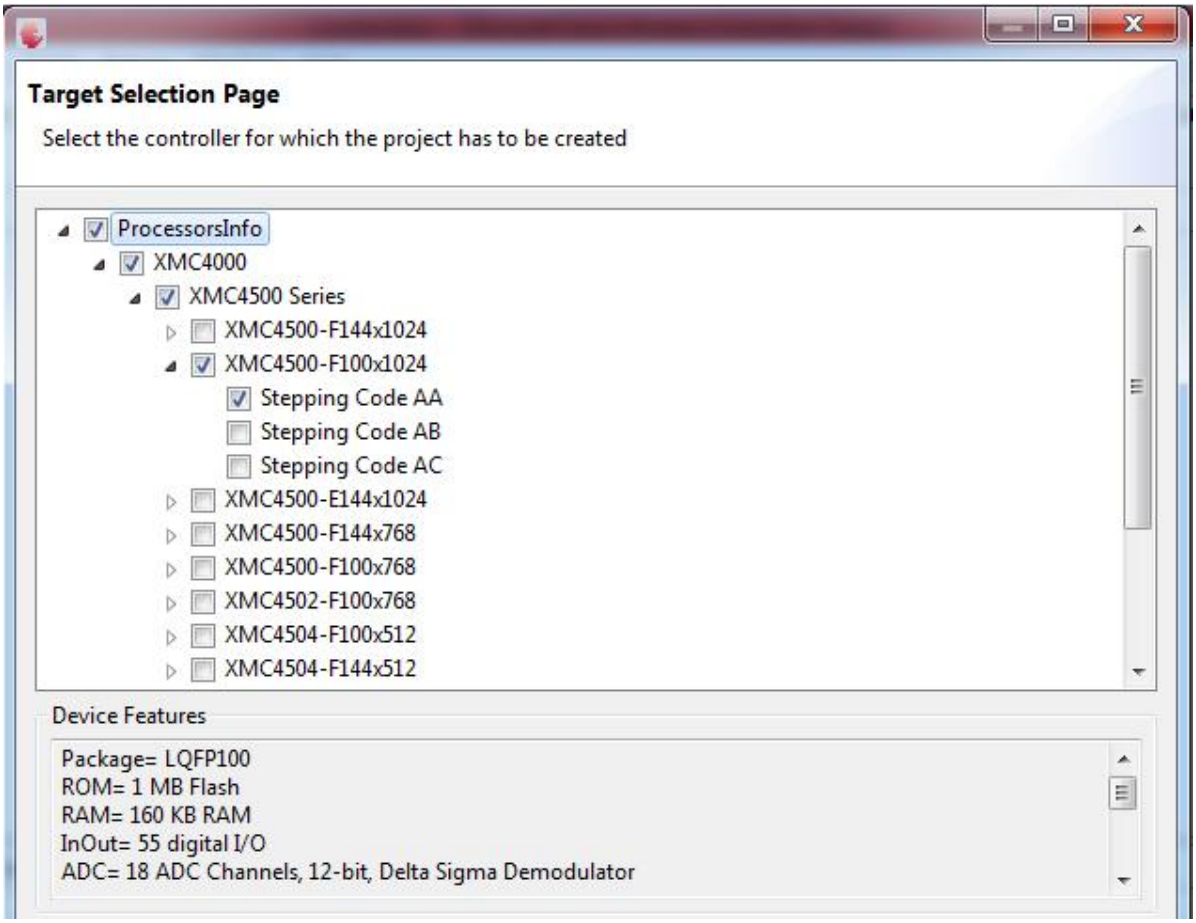

Figura 5. Fereastra Target Selection Page

- În meniul **App Selection View** se caută aplicația dorită și se face dublu-click;
- Se face click pe **DAVE** > **Generate Code**.

În urma acestor pași avem un program bazat pe un cod generat de mediul de dezvoltare DAVE IDE care folosește aplicația dorită din cadrul **DAVE Apps**.

4 Depanarea programului

Pentru a depana programul se parcurg următorii pași:

- Se face click pe meniul **Debug**;
- Click pe **TASKING C/C++ Debug** pentru a crea o instanță de depanare;
- Din tab-ul Debugger se selectează **Infineon XMC4500 Relax Kit**;
- Se selectează conexiunea **J-Link over USB**;
- Se face click pe **Debug**;

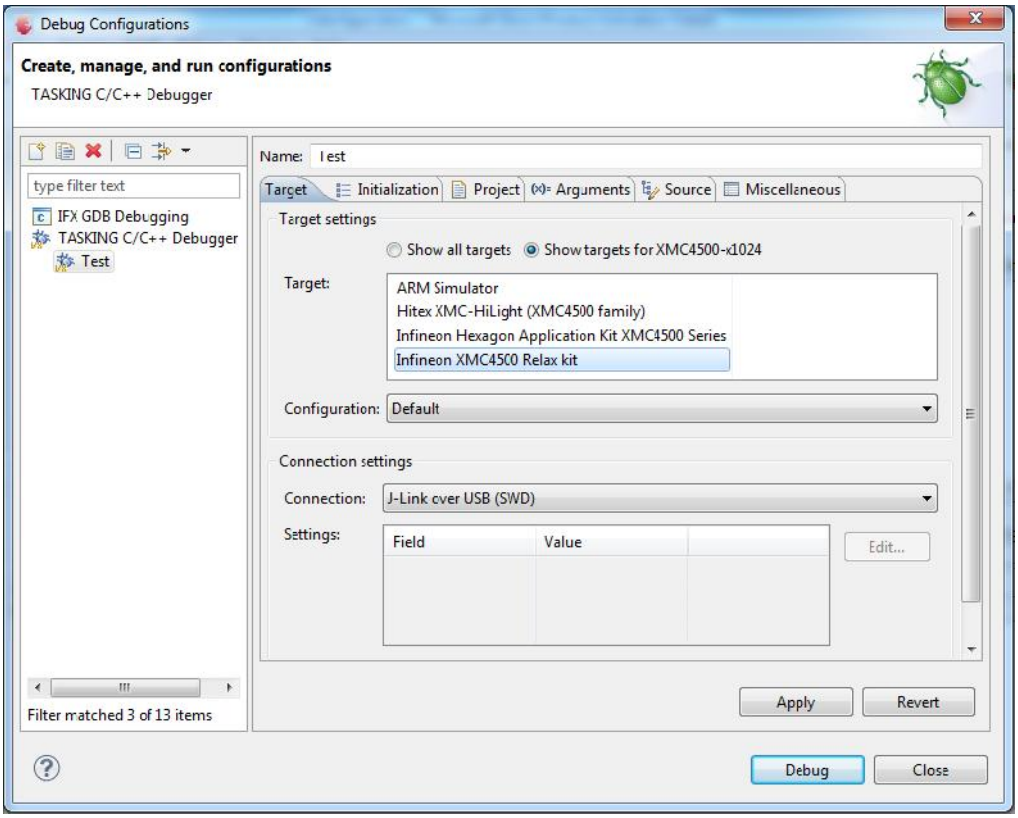

Figura 6. Fereastra de configurare a depanatorului

- Se lansează sesiunea de depanare prin apăsarea tastei **F11**.

Se rulează programul pas cu pas şi se vizualizează resursele.

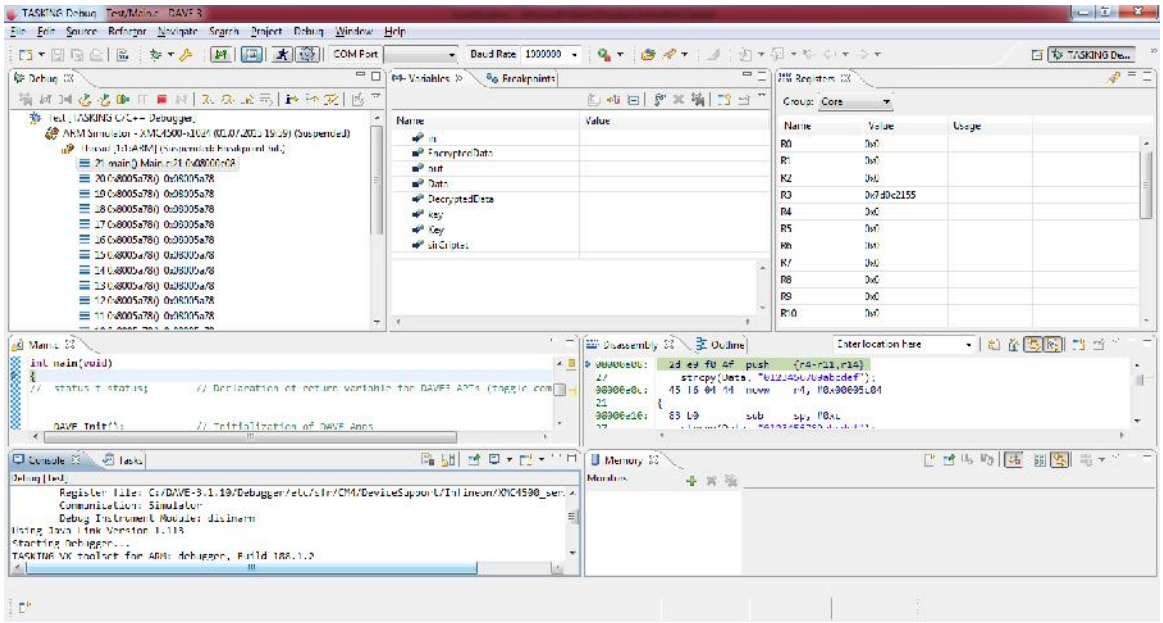

Figura 7. Interfața de depanare