4.5. Sisteme de operare specifice

4.5.1 Standarde pentru sisteme de operare real-time

Sistemele de operare in timp real si embedded nu reprezinta produse utile in sine, ci servesc ca platforma pe care se construiesc aplicatiile. Ca pentru orice platforma, existenta standardelor faciliteaza enorm munca programatorilor, pentru ca face scrierea si portarea aplicatiilor pe platforme hardware noi mult mai simpla, ieftina si rapida. In lumea embedded si real-time standardizarea nu este aspectul critic, pentru ca majoritatea proiectelor au cerinte specifice, necesita extensii unice la produsele deja existente, nu necesita modificari soft frecvente si sint rareori vizibile utilizatorului final. Toate aceste "caracteristici" nu ajuta cu adevarat in fortarea dezvoltatorilor de a utiliza standardele; ei prefera totusi sa lucreze cu unelte standard, astfel explicindu-se popularitatea programelor free software GNU.

POSIX

POSIX("Portable Operating System Interface", un nume date de parintele miscarii free software, Richard Stallman) este un standard pentru apeluri de functii (interfata de programare a aplicatiilor API) a sistemelor de operare compatibile UNIX. POSIX are citeva specificatii referitoare la primitive real-time. Definitia sa despre timp real este destul de slaba – Abilitatea unui sistem de operare de a oferi un anumit nivel al serviciilor intr-un timp de raspuns determinat.

Standardul POSIX este gestionat de comitetul Portable Application Standards Committee(PASC) al IEEE.

Componentele POSIX relevante pentru real-time sint: 1003.1b (real-time), 1003.1d(extensii aditionale real-time), 1003.1j (extensii real-time avansate).

Specificatiile POSIX definesc de asemenea 4 asa numite profile pentru sistemele in timp real:

- PSE51 (Minimal Realtime System Profile). Acest profil ofera un set de functionalitati de baza pentru un sistem embedded pur, monoproces, de exemplu pentru controlul autonom al dispozitivelor I/O speciale. Nu este necesara nici interactiunea cu utilizatorul nici un sistem de fisiere (stocare de masa). Sistemul ruleaza un singur proces POSIX care poate avea mai multe fire de executie POSIX. Aceste fire de executie pot utiliza transferul de mesaje POSIX. Procesul poate utiliza transferul de mesaje pentru a comunica cu alte sisteme conforme PSE5X (de exemplu, mai multe procesoare pe un backplane comun, fiecare rulind un sistem PSE51 independent). Modelul hardware al acestui profil presupune un singur procesor impreuna cu memoria sa, dar nu sint necesare unitate de management a memoriei sau dispozitive periferice comune – linii seriale, placa de retea ethernet.
- PSE52(Realtime Controller System Profile). Acest profil este profilul PSE51 plus suportul pentru sistem de fisiere (posibil implementat ca un disc RAM) si operatiuni I/O asincrone.

- PSE53(Dedicated Realtime System Profile) este profilul PSE51 plus suportul pentru procese multiple, dar fara suportul sistemului de fisiere de la PSE52. Platforma hardware poate avea o unitate de management al memoriei.
- PSE54(Multi-Purpose Realtime System Profile) este un superset al celorlalte profile si consta in esenta in toate specificatiile standardelor POSIX.1, POSIX.1b, POSIX.1c si POSIX.5b. Nu toate procesele sau firele de executie trebuie sa fie in timp real. Procese ineractive cu utilizatorul sint permise pe un sistem PSE54, astfel ca toate specificatiile POSIX.2 si POSIX.2a sint de asemenea incluse. Modelul hardware al acestui profil presupune unul sau mai multe procesoare cu unitati de management al memoriei, dispozitive de stocare rapide, interfete speciale, suport de retea si dispozitive de afisare.

Obiectivul Linux este respectarea standardelor POSIX, dar nu oricum si cu orice cost. Fisierul header /usr/include/unistd.h da informatii despre ce parti ale standardului sint deja implementate. De exemplu, implementarea firelor de executie si a modurilor planificatorului

Specificatii real-time pentru Java

Specificatiile real-time pentru limbajul Java nu reprezinta de fapt un sistem de operare, ci un runtime pentru acest limbaj de programare. Distinctia nu este neaparat fundamentala pentru utilizatorul desktop normal, dar devine extrem de importanta la aplicatii real-time, pentru ca biblioteca runtime trebuie sa faca utilizabile serviciile sistemului de operare pe care lucreaza. Acest lucru inseamna ca o biblioteca runtime cu facilitati de timp real este inutila pe un sistem de operare non-real-time.

Specificatiile realtime pentru Java au fost publicate in 2001 si, similar specificatiilor POSIX, nu reprezinta o implementare; exista deja citeva implementari comerciale. Aspectele de baza ale acestor specificatii sint:

- implementarilor specificatiilor le sint permise sa introduca optimizarile si extensiile lor proprii, ca de exemplu algoritmi de planificare sau colectarea memoriei eliberate
- gestiunea minimala a proceselor include planificare preemtiva bazata pe prioritati statice cu cel putin 28 nivele de prioritate
- prevenirea inversiunilor de prioritate este obligatorie
- exceptiilor trebuie sa le fie permis sa schimbe contextul altui fir de executie
- trebuie sa existe clase ce ofera acces direct la memoria fizica

4.5.2 Linux pentru aplicatii real-time si embedded

Linux este un sistem de operare de uz general cu un nucleu non-preemtiv: incearca sa dea tuturor proceselor un procent corect din resursele sistemului(procesor, memorie, dispozitive periferice...) si nu intrerupe activitatile nucleului. Planificatorul spatiului utilizator este de tip time-slicing (cuante de timp): el acorda mai multe sau la fel de multe cuante de timp proceselor. Este posibila schimbarea intr-o anumita masura a prioritatilor proceselor utilizator(prin comanda nice) dar nu se poate face suficient pentru a avea o planificare determinista. Alte motive pentru care Linux este un sistem in timp real slab sint date de intirzierile nepredictibile cauzate de operatiile non-preemtive rulind in spatiul nucleu si de marimea mare a nucleului. Nimeni nu poate intelege nucleul suficient de bine pentru a fi in stare sa prezica timpul de executie al unei anumite operatii.

Structuri hardware si algoritmi specifici microsistemelor EMBEDDED

Considerentele de mai sus se aplica de asemenea pentru toate sistemele de operare de uz general, Windows, AIX, IRIX, HP-UX, Solaris. Pare paradoxal dar batrinul DOS era mai aproape de un sistem in timp real decit Linux,pentru ca planificatorul sau era mai putin “corect” si avansat, si avea mai putine servicii sistem de care trebuia sa se tina cont. Oricum, DOS era in avantaj numai daca era un singur proces in timp real!.

Pentru ca nici unul din sistemele de operare desktop sau server nu este un candidat bun pentru aplicatii embedded sau in timp real, diverse companii au pornit dezvoltarea de sisteme de operare speciale, mai des pentru piete de dimensiuni reduse. Majoritatea sint similare UNIX, dar nu sint mutual compatibile. Piata este extrem de fragmentata, cu zeci de astfel de sisteme, si nici unul nu detine supremația absoluta. Aceasta era, cel putin, situatia inaintea aparitiei Linux pe aceasta piata. Dupa anul 2000 au avut loc o serie de fuziuni si achizitii si eforturi substantiale din partea companiilor pentru a oferi produse „compatibile Linux“. Faptul ca Microsoft a patruns de asemenea pe aceasta piata cu linia de produse PocketPC/ Windows CE nu a facut decit sa accelereze aceasta evolutie.

In lumea dezvoltatorilor nucleului Linux nu se fac eforturi pentru transformarea acestui sitem de operare in unul in timp real, dar nucleul evolueaza in directia unei mai mari preemtivitati, datorita, in primul rind, nevoii de a ataca lumea platformelor multiprocesor. Creatorul Linux, finlandezul Linus Torvalds a mentionat de ce Linux nu va deveni un sistem de operare in timp real:

- calculatoarele devin tot mai rapide astfel ca un sistem de operare de uz general va satisface din ce in ce mai mult cerintele utilizatorilor “real-time”.(aceasta inseamna totusi cei care au nevoie de un sistem rapid, ceea ce nu este acelasi lucru cu sistem determinist)
- oferta de facilitati real-time intr-un sistem de uz general va conduce la comportamente gresite ale programatorilor de aplicatii – fiecare va dori ca aplicatia sa sa ruleze cel mai rapid, si va fi programata cu prioritate maxima. Experienta arata ca de cele mai multe ori acest fapt conduce la constringeri de timp incompatibile intre diferite aplicatii

Oricum, nu exista motive tehnice pentru care Linux sa nu devina intr-o masura mai mare real-time, iar tehnologia pentru imbunatatirea sa in zona serverelor high-end este utila si scopurilor real-time si embedded: multi-threading real in nucleu, blocari si puncte de planificare mai fine cerute de multiprocesarea simetrica, migrarea proceselor intre procesoare, dispozitive “hot-swap” etc.

Exista oricum o serie de eforturi in zona free software referitoare la aplicatiile real-time si embedded. Acestea vizeaza mai multe directii:

- Eliminarea functionalitatilor din nucleul Linux standard.

Aceasta abordare doreste reducerea necesarului de memorie al sistemului de operare si este deci centrata pe sistemele embedded. uCLinux este un exemplu in acest sens. Alte proiecte dezvolta biblioteci C simple si de dimensiuni reduse, pentru ca versiunile GNU normale au devenit foarte mari; de exemplu, BusyBox este un inlocuitor pentru majoritatea programelor utilitare intilnite intr-o distributie Linux, uCLib este o versiune simplificata a bibliotecii standard C.

- anexe la nucleul Linux standard

Aceasta abordare inlocuieste planificatorul standard cu unul bazat pe un algoritm determinist, si adauga puncte de planificare arborelui sursa Linux, totul pentru a face nucleul mai rapid in raspuns.

- anexe real-time la nucleu

In aceasta abordare Linux ruleaza ca un proces de prioritate scazuta intr-un nucleu real-time de dimesiuni reduse. Acest nucleu ia controlul asupra resurselor hard si le prezinta nucleului Linux sub forma simularii software. Cele doua exemple majore care urmeaza aceasta cale sint RTLinux si RTAI.

- sisteme de operare independente de Linux

Aceste proiecte au fost dezvoltate complet independent de Linux, iar unele dintre ele chiar inaintea acestuia. De exemplu – RT_EMS si eCos.

Nucleul Linux al unui calculator desktop tipic ocupa citeva sute de KB fara a pune la socoteala memoria ocupata de programele utilitare si de aplicatie. Din acest motiv dimesiunea sa este mult prea mare pentru multe sisteme embedded; de asemenea, Linux necesita un minut sau mai mult pentru pornire pe un calculator PC uzual, timp mult prea lung pentru majoritatea aplicatiilor embedded., si asteapta sa ruleze pe un sistem cu harddisk, cu o sursa de minim 100W ce alimenteaza un procesor modern si placa video. Totusi, unul din avantajele majore ale Linux este configurabilitatea, pe baza careia se poate obtine propriul nucleu prin compilare. Nucleul poate fi construit din diferite module mai mult sau mai putin independente, astfel ca modulele inutile aplicatiei pot fi lasate de-o parte la compilare. In acest fel se pot elimina suportul pentru retea, placa video si alte asemenea periferice, obtinind in final un sistem suficient de mic incit incape pe un singur disc floppy.

Aspectele analizate anterior pot sugera ca sistemul de operare Linux nu are nici o sansa pentru a fi utilizat ca sistem embedded. Cu toate acestea, el dispune de anumite avantaje ce se pot dovedi decisive curind (in special pentru ca pretul procesoarelor si al memoriei este in continua scadere): configurabilitatea, abilitatea de a fi administrat de la distanta sau multele posibilitati de a adauga facilitati de securitate.

Bibliografie:

1. Istvan Sztojanov, Sever Paşca, Elisabeta Buzoianu, Aplicaţii hardware şi software cu microcontrolerul PIC12F675, Editura Cavallioti, ISBN 978-973-7622-54-9, Bucuresti 2008
2. Istvan Sztojanov, Alexandru Vasile, Elisabeta Buzoianu, Sever Paşca, *Programarea microcontrolerelor din familia Intel, Aplicaţii practice hardware cu 80C552*, Editura Man-Dely, ISBN 973-85681-5-3, Bucureşti 2004.
3. http://vega.unitbv.ro/~romanca/EmbSys/
4. http://facultate.regielive.ro/cursuri/electronica/
5. www.microcip.com
6. Andrei Drumea, Teza de doctorat, UPB 2009