

2.2. Tehnici de gestionare a perifericelor: pipelining, DMA, PEC

Comunicația unui sistem EMBEDDED cu perifericele este o acțiune foarte importantă pentru acesta mai ales în castigarea tipului de execuție. O îmbunătățire importantă este dată de perfecționarea structurii de tip pipeline. La Sistemele EMBEDDED unitatea de prelucrare este împărțită în unitatea de interfață cu magistrala (BIU), care se ocupă de aducerea în avans a instrucțiunilor din memorie și depunerea lor într-o coadă, și unitatea de execuție (EU), care preia instrucțiunile din coadă și le execută. Această împărțire permite **lucrul în paralel** al celor două unități, ceea ce se traduce printr-o funcționare mai rapidă. Fiecare instrucțiune (care constă din 4 faze - adresare, citire, decodificare, execuție) are faze mai complexe sau mai simple. Aceste faze (mai ales cea de execuție, care este cea mai complexă) poate consta la rândul său din mai multe operații mai simple. Ideea este că fiecare din aceste operații lucrează în principiu cu alte resurse, deci toate operațiile se pot executa în paralel. Astfel, execuția unei instrucțiuni de comunicație cu un periferic poate fi împărțită într-un număr mare de acțiuni elementare, numite stagii ale pipeline-ului. Deci, la un moment dat se pot afla în execuție în microcontroler mai multe instrucțiuni, în diferite faze; în cazul cel mai fericit există câte o instrucțiune tratată în fiecare stadiu al pipeline-ului. Deși execuția unei instrucțiuni de la început până la sfârșit necesită un număr mare de acțiuni, o instrucțiune poate începe să fie executată imediat ce instrucțiunea anterioară a trecut de primul stadiu.

De ce este atât de eficientă această structură? Activitatea microcontroler este coordonată cu ajutorul semnalului de ceas al sistemului. Trecerea execuției unei instrucțiuni de la un stadiu la altul se poate face numai atunci când "bate" ceasul, deci la intervale regulate de timp. Pe de altă parte, fiecare acțiune elementară (stadiu) se execută într-o anumită durată finită de timp. Dacă semnalul de ceas este prea rapid, acțiunile nu se mai pot realiza pe durata dintre două "bătăi" ale ceasului, ceea ce ar duce la pierderea controlului asupra execuției instrucțiunilor. Ca urmare, frecvența ceasului nu poate fi crescută oricât de mult, ci este limitată de duratele de execuție ale stagiilor. Dacă acțiunile elementare sunt mai simple (ceea ce implică o descompunere mai fină a instrucțiunilor și deci un număr de stagii mai mare), ele vor consuma mai puțin timp; implicit, frecvența ceasului va putea fi crescută. Dacă analizăm funcționarea unui pipeline, observăm că, în cazul cel mai fericit, la fiecare "bătaie" a ceasului se poate termina de executat câte o instrucțiune, deci performanța procesorului depinde direct de creșterea frecvenței semnalului de ceas.

Sistemele EMBEDDED au evoluat în sensul creșterii continue a numărului de stagii a pipeline-ului. La ultimele microcontroler s-a ajuns la un pipeline cu 8 stagii, ceea ce este mult mai mult decât oricare variantă anterioară. Cu alte cuvinte, execuția unei instrucțiuni a microcontrolerului este împărțită în 8 operații elementare. Pentru sistemele EMBEDDED cu număr foarte mare de stagii se folosește și denumirea de unități superpipeline.

Totuși, structura de tip pipeline are și dezavantaje. Ideea sa de pornire este că fiecare stadiu lucrează cu alte resurse ale microcontrolerului sau sistemului EMBEDDED decât restul stagiilor. Această cerință nu poate fi niciodată satisfăcută în totalitate. Ca un exemplu simplu, o operație de adunare necesită folosirea unității aritmetico-logice (ALU) pentru efectuarea



Structuri hardware si algoritmi specifici microsistemelor EMBEDDED

calculului propriu-zis. În același timp, în faza de adresare a unei instrucțiuni, valoarea registrului indicator de instrucțiuni este incrementată, pentru a putea aduce codul următoarei instrucțiuni. Deoarece incrementarea este tot o operație de adunare, va fi nevoie tot de ALU. Astfel, o instrucțiune de adunare aflată în faza de execuție și o altă instrucțiune aflată în faza de adresare vor concura pentru aceeași resursă (ALU). Asemenea situații apar de fapt mult mai des, deoarece între instrucțiuni există relații de dependență rezultate din însăși logica programului. Se întâmplă foarte des ca o instrucțiune să aibă nevoie de rezultatul unei instrucțiuni anterioare, care încă nu l-a calculat. Din acest motiv, de multe ori o instrucțiune (și implicit cele care urmează după ea) trebuie să aștepte până când devine disponibilă o resursă de care are nevoie, dar care este momentan folosită de altă instrucțiune. După cum am văzut, o asemenea resursă poate fi fie o componentă hardware a procesorului, fie rezultatul altei instrucțiuni. Ca urmare, în practică se întâmplă rareori ca microcontrolerul să termine de executat câte o instrucțiune la fiecare "bătăie" a ceasului, deci câștigul de performanță nu este atât de mare cât sperăm.

O concluzie importantă care se desprinde de aici este că simpla lungime a pipeline-ului (adică numărul de stagii) nu este singurul factor care influențează performanța procesorului. Împărțirea corectă a instrucțiunilor în operații elementare poate fi la fel de importantă. Din păcate, această alegere nu poate fi făcută ca urmare a unor considerații teoretice sau a unor criterii clare, fiind în mare măsură o problemă de inspirație. În practică se poate vedea cum microprocesoarele AMD, care sunt compatibile cu cele Intel la nivel de limbaj, dar au o implementare diferită pentru pipeline (care este mult mai scurt), reușesc să obțină performanțe asemănătoare, deși lucrează la frecvențe mult mai mici. O soluție deja folosită de microcontrolerele actuale este existența a două sau mai multe pipeline-uri; astfel se pot executa mai multe instrucțiuni în paralel, atunci când dependențele dintre instrucțiuni nu introduc perioade de așteptare.

Bibliografie:

1. Istvan Sztojanov, Sever Pașca, Elisabeta Buzoianu, Aplicații hardware și software cu microcontrolerul PIC12F675, Editura Cavallioti, ISBN 978-973-7622-54-9, Bucuresti 2008
2. Istvan Sztojanov, Alexandru Vasile, Elisabeta Buzoianu, Sever Pașca, *Programarea microcontrolerelor din familia Intel, Aplicații practice hardware cu 80C552*, Editura Man-Dely, ISBN 973-85681-5-3, București 2004.
3. <http://vega.unitbv.ro/~romanca/EmbSys/>
4. <http://facultate.regielive.ro/cursuri/electronica/>
5. www.microcip.com
6. Vlad Rădulescu, Henri Luchian, Adrian Buburuzan Arhitectura calculatoarelor și sisteme de operare, Departamentul de Învățământ la Distanță



UNIUNEA EUROPEANĂ

MINISTERUL MUNCII, FAMILIEI ȘI
PROTECȚIEI SOCIALE
AMFOSDRUFONDUL SOCIAL EUROPEAN
POSDRU
2007-2013INSTRUMENTE STRUCTURALE
2007-2013

ELAN

**Promovarea Culturii Antreprenoriale: Adaptabilitate, Dinamism, Inițiativă în Industria Electronică
Investește în oameni !**

Proiect cofinanțat din Fondul Social European prin

Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013"

Structuri hardware si algoritmi specifici microsystemelor EMBEDDED



UNIUNEA EUROPEANĂ



MINISTERUL MUNCII, FAMILIEI ȘI
PROTECȚIEI SOCIALE
AMFOSDRU



FONDUL SOCIAL EUROPEAN
POSDRU
2007-2013



INSTRUMENTE STRUCTURALE
2007-2013