

## Program által meghatározott processzorok

A rendszerlogika menet közben megvalósított újraépítésével ez a módszer képessé tehet egyetlen FPGA-t tíz vagy akár több száz közönséges processzor feladatainak ellátására.

**A**z alkalmazás által meghatározott processzorok az *átszerkeszthető számítási elv (reconfigurable computing, RC)* elvén alapulnak. Az RC egy olyan számítási eljárás, amely elmosza a hardver és szoftver közti határvonalat és megteremti az alapjait annak, hogy újabb nagy lépést tegyünk a számítási hatékonyság növelése terén, csökkentett teljesítmény- és tárterület-igény mellett. Az RC gyakorlati megvalósítása az *átszerkeszthető (reconfigurable)* hardvereszközök alkalmazásával történik. Egy RC-rendszerben lévő processzorok olyan hardvereszközök, amelyek a rajtuk futtatott programra lettek optimalizálva.

Cikkemben elmagyarázom az RC eljárás elvét, megvizsgálók néhány SRC-rendszert, amelyek az RC gyakorlati megvalósítását jelentik, és megmutatom, hogy milyen teljesítménybeli előnyökkel bír az RC a hagyományos mikroprocesszorokhoz képest. Bemutatom emellett az RC programozási modelljét és az RC-ben rejlő lehetőségeket az *Open Hardware* támogatására.

### Mit jelent az átszerkeszthető számítási elv és miért fontos ez a számunkra?

Az RC egy a hardveren alapuló számítási módszer, amely minden egyes futtatandó alkalmazás számára dinamikusan hozható létre. Az RC olyan hardverelemekből épül fel, amelyek dinamikusan megadható logikájú áramkörtartalmaznak, vagyis az alkalmazott számítási módszer nem a gyártáskor kerül rögzítésre. Az RC már sok éve létezik, és számos hardverösszetevőben került már megvalósításra. Ilyenek az FPGA-k (*Field Programmable Gate Array, általános célú programozható áramkör*), az FPOA-k (*Field Programmable Object Array*, az FPGA-hoz hasonló jellegű, de számos fontos tulajdonságában eltérő programozható áramkör - a ford.) és az *összetett programozható logikájú eszközök (complex programmable logic devices, CPLD)*.

Az alkalmazásfejlesztők számára fontos tényező, hogy a modern újraszerkeszthető áramkörtartalmaz olyan órajellel és teljesítménnyel rendelkeznek, amelyek lehetővé teszik, hogy RC-hardverekben nagyteljesítményű számításokra is alkalmazzák azokat.

Az RC megvalósítására használt legelterjedtebb lapkátípus az FPGA. Az FPGA SRAM memóriacellákból felépülő lapka,

amelyben ezek a memóriacellák tárolják a lapka beállításait. Az FPGA-k logikai kapukat, flip-flopokat, RAM-ot, aritmetikai magot, órajel-generátort és az ezek összekötésére szolgáló beállítható huzalozást tartalmaznak. Az FPGA-k tetszőleges logikai funkció megvalósítására beállíthatók, így olyan egyedi processzorok hozhatók létre, melyek egy adott alkalmazásra optimalizálhatók.

Egy FPGA-készlet így alkothat akár MIPS, SPARC PowerPC vagy Xeon processzort, esetleg egy teljesen egyedi felépítéssel rendelkezőt is. Valójában az sem szükséges, hogy utasításfeldolgozó egységről legyen szó, *közvetlen futtató logika (DEL, direct execution logic)* is lehet, amely csak számítási logikát tartalmaz, és nincs szüksége az algoritmust meghatározó utasításokra.

A közvetlen futtató logikát tartalmazó (DEL) processzorok igen nagy teljesítmények elérését teszik lehetővé. Ezek a processzorok pontosan az adott algoritmus végrehajtásához szükséges erőforrásokkal hozhatók létre. A hagyományos utasításfeldolgozó egységek rögzített erőforrásokkal rendelkeznek, összedókkal, szorzókkal, regiszterekkel és átmeneti tárrakkal, és jelentős lapkafelület és teljesítmény szükséges az olyan többletmunka végrehajtásához, mint az utasítások visszafejtése, a végrehajtási sorrend megállapítása és az átmeneti tár kezelése.

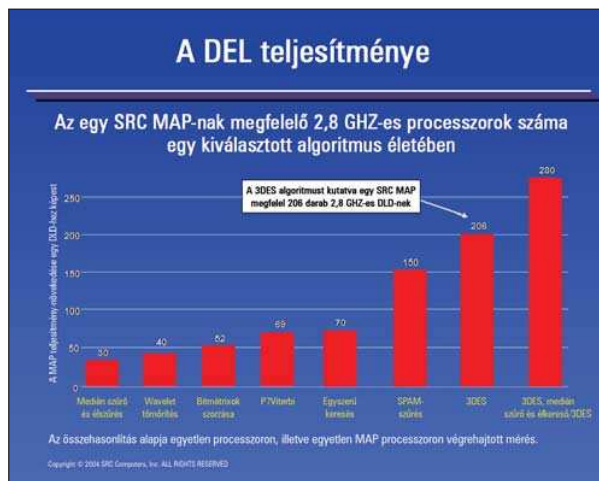
A DEL-processzorok olyan átrendezhető számítógépek, amelyekben minden alkalmazáshoz más-más felépítés tartozik szemben a rögzített felépítésű processzorokkal, amelyeknél mindent ugyanazzal az egységgel kell megoldani. A DEL-processzorok a leghatékonyabb áramkörtartalmaz felépítést szolgáltatják egy adott alkalmazáshoz az algoritmusban található párhuzamosságok kezelésének és a funkcionális egységek pontosságának tekintetében. Az átépíthetőségéből adódóan minden program számára egyedi DEL-processzor hozható létre a másodperc tört-része alatt.

De miért fontos számunkra, hogy a DEL-processzorok dinamikusan hozhatók létre egy alkalmazás számára és hogy hatékonyabban használják ki a rendelkezésükre álló áramkörtartalmaz lapkát, mint a hagyományos mikroprocesszor?

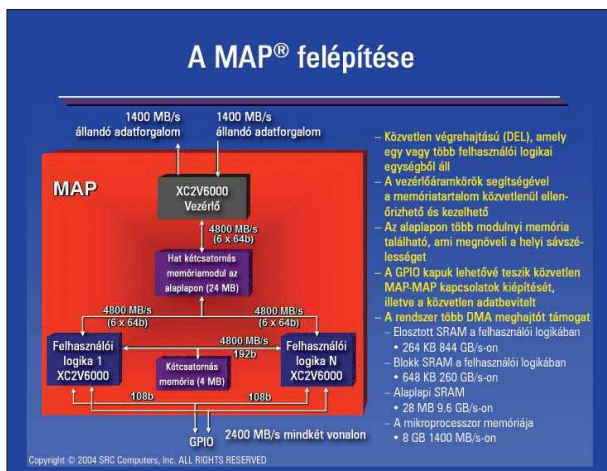
A válasz egyszerű: hatékonyabb teljesítmény- és energiafelhasználás. Egy DEL RC processzor úgy építhető fel, hogy tartalmaz minden párhuzamosságot, ami az adott



1. ábra A közvetlen futtatású logika (DEL) minden logikai kaput a valódi probléma megoldására mozgósít



2. ábra Számos 2,8 GHz-es processzor szükséges a MAP közvetlenül futtató processzor teljesítményének az eléréséhez



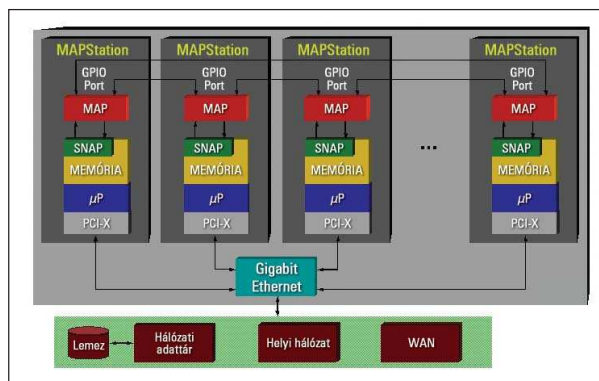
3. ábra A MAP szerkezeti felépítése

algoritmusban előfordul, a mikroprocesszorban lévő felesleges többletmunka nélkül. A cikk további részében a téma részletesebb tárgyalásának érdekében az RC processzorokat FPGA elemekkel megvalósított egységeknek feltételezzük.

### Hogyan érhető el az a bizonyos nagy teljesítmény?

Az RC processzorok teljesítménye a logika párhuzamos futtatási képességéből adódik. Az RC processzorok teljes mértékben párhuzamos működésűek. Valójában egy adott algoritmusnak megfelelő logika létrehozása nem áll másból, mint a párhuzamos futások összehangolásából, vagyis hogy a részeredmények a megfelelő pillanatban jöjjenek létre, kerüljenek megosztásra illetve visszatartásra.

A DEL processzor az adatutak és vezérlőjelek által összekapcsolt funkcionális egységek hálózata. A hálózatban lévő minden számítási egység minden egyes órajelre aktív válik. Az 1. ábra egy logikai részletet mutat egy kifejezés kiszámítására és a lapka kihasználtságára egy olyan Neumann-féle utasítás-processzorban, mint amilyen az Intel Pentium 4 mikroprocesszor is.



4. ábra A fűrtözött SRC-6 rendszer felépítése

Habár egy mikroprocesszor 3GHz körüli órajelen képes működni szemben az FPGA lapkák 100-300 MHz-es frekvenciájával, a párhuzamosságnak és a belső sávszélességnek köszönhetően egy DEL-processzor az összteljesítményt tekintve nagyságrendekkel múlhatja felül a mikroprocesszort. A 2. ábrán néhány összehasonlító teljesítményteszt látható, amelyben az SRC DEL-processzora a MAP és egy jellegzetes Neumann-féle utasításvégrehajtó processzor, az Intel Xeon 2,8 GHz mikroprocesszor méri össze erejét. A párhuzamos futás, a pontosan a szükséges számú funkcionális egység alkalmazása, a nagy belső sávszélesség, az utasítás feldolgozásából, betöltéséből és tárolásából adódo többletmunka kiküszöbölése együttesen vezetnek a MAP és Intel processzorok közti 30-szoros órajelkülönbséghez.

### Képes a DEL processzor a Linux futtatására?

Elvileg a DEL alapú processzorok képesek lennének a Linux futtatására, de szükség van-e erre egyáltalán? A Linux rendszermagjának kódszövegei minden bizonnyal nagyobb teljesítménnyel futnának egy DEL processzoron, és a Linux rendszercsomag programjai is éreznék ennek az előnyét. Mégis, egy operációs rendszernek, és különösen a rendszermagynak az a szerepe, hogy kezelje a hardvert, és elérhetővé tegye a programok számára a megfelelő teljesítményszintet. Más szóval



az operációs rendszernek félre kell állnia az útból és hagyni, hogy az alkalmazások kihasználhassák a hardver nyújtotta szolgáltatásokat.

A programok nem csak elmélyült számítási műveletekkel foglalkoznak, hanem emellett kapcsolatot tartanak a felhasználókkal, fájlokat olvasnak és írnak, megjelenítik az eredményeket és a Világhálón keresztül információt cserélnek a világgal. Az alkalmazásoknak tehát egyaránt szükségük van számítási erőforrásokra és egy operációs rendszer szolgáltatásaira. A nehéz számítási műveletek és a magas fokú párhuzamosság ki tudja használni a *DEL* processzorok előnyeit. Bár a soros kódok is futhatnak közvetlen futtató logikán, ezeket mégis a hagyományos processzorok tudják a legjobban kiszolgálni.

A legtöbb alkalmazás számára az optimális megoldást a hagyományos és *DEL* processzorok keveréke jelenti. Ez a kombináció lehetővé teszi az alkalmazások számára, hogy nagyságrendekkel nagyobb teljesítményt érjenek el, miközben a hagyományos *Linux* környezetben futnak és rendelkezésükre áll az operációs rendszer összes szolgáltatása és megszokott eszköze. Az alkalmazásnak azok a részei, amelyben túlsúlyban vannak a soros kódok, vagy amelyek az operációs rendszer szolgáltatásait igénylik, a rendszer hagyományos processzort tartalmazó részén futtat, míg azokat az alkalmazásokat, sőt az operációs rendszer bizonyos részeit, amelyek számára előnyös a *DEL* párhuzamosága, a szorosan a rendszerhez kapcsolt *DEL* processzorok szolgálják ki.

### Az SRC Computers RC-rendszere

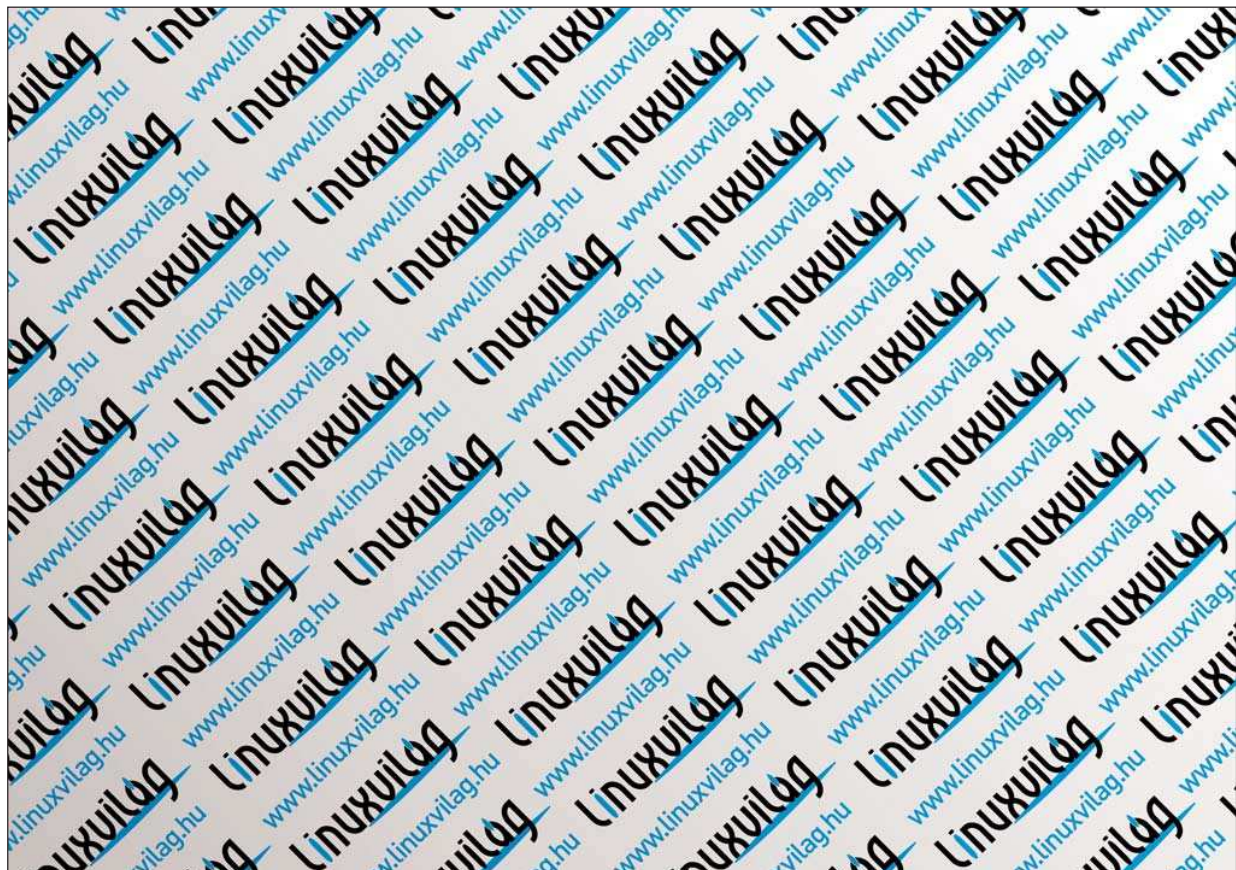
Az *SRC* létrehozott *DEL*-processzorokat és mikroprocesszorokat egyaránt tartalmazó rendszereket. Ezek a rendszerek operációs rendszerként a *Linuxot* futtatják, emellett biztosítanak egy *Carte* nevű programozói környezetet, amely alkalmas a mikroprocesszoros utasításokat a *DEL*-lel ötvöző programok írására, és egyetlen rendszeren belül támogatja a mikroprocesszorra és *DEL*-processzorra épülő hardverhátteret.

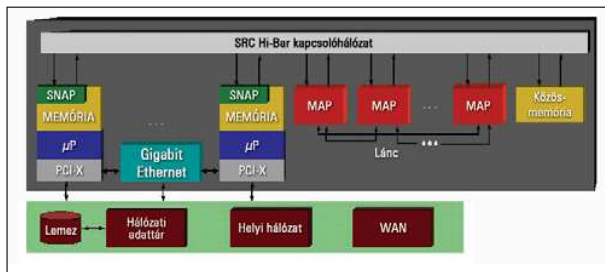
### A DEL-processzor: MAP

A *MAP* az *SRC* nagyteljesítményű, szabadalmazott *DEL*-processzora. A *MAP* átszerkeszthető összetevőket tartalmaz a vezérlés és felhasználó által meghatározott számítások megvalósítására, az előzetes utasításkód-lehívásra és az adat-elérésre. Ez a számítási képesség igen nagy belső és külső sávszélességgel párosul. A *MAP* kétkapus alaplapra integrált memóriái 11,2 GB/sec helyi átviteli sebességet biztosítanak. A *MAP* külön bemeneti és kimeneti kapukkal van felszerelve, amelyek 1,4 GB/sec adatforgalom kezelésére képesek. Ezen felül minden *MAP* rendelkezik két általános I/O (*GPIO*) kapuval további 4,8 GB/sec sávszélességet biztosítva a közvetlen *MAP*-*MAP* kapcsolat vagy a forrásadatok számára. A *MAP*-processzor szerkezeti felépítését a 3. ábrán láthatjuk.

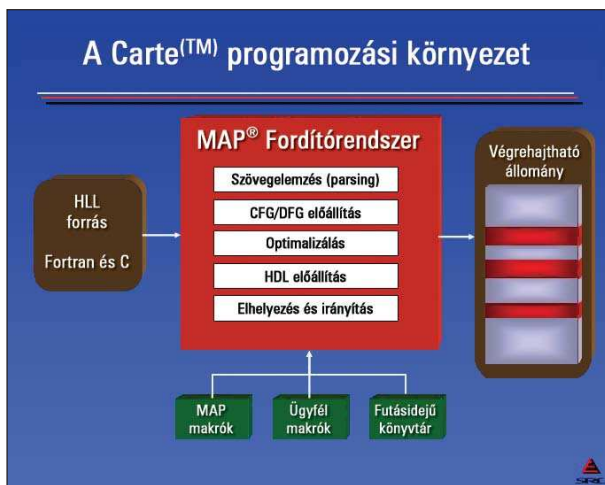
### A mikroprocesszorok kapcsolódása a SNAP segítségével

Az ezekben az eszközökben használt *DLD*-k (*Dense Logic Device, hagyományos fix logikájú eszközök*) kétprocesszoros *Intel IA-32* egységek. Ezek a külső

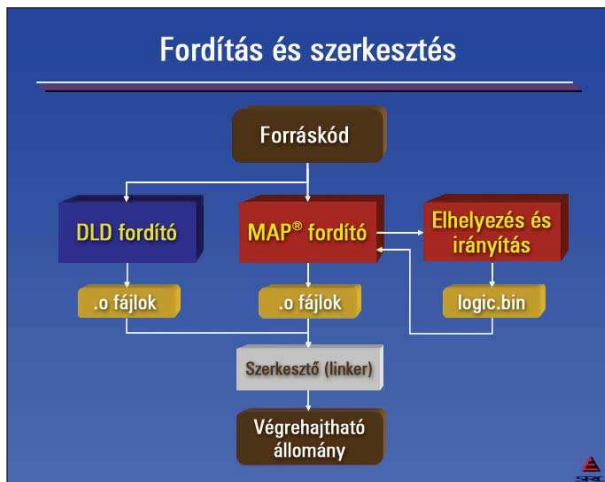




5. ábra A Hi-Bar kapcsolóval felépített SRC-6 elrendezése



6. ábra A Carte programozói környezet



7. ábra A Carte fordításának folyamata

gyártótól származó áramkörök az SRC által kifejlesztett SNAP-felülethez csatlakoznak. A SNAP lehetővé teszi a hagyományos alaplapok csatlakozását és a memóriamegosztást a MAP processzorokkal és a közös memóriacsomópontokkal, amelyek az SRC-rendszer további észét képezik.

A SNAP-felületet úgy tervezték, hogy az ne a mikroprocesszor ki/bemeneti alrendszerére, hanem közvetlenül a memória-alrendszerhez csatlakozzon, ezáltal lényegesen nagyobb csatlakozási sávszélességet biztosítva.

A SNAP külön kimeneti és bemeneti kapukat használ, amelyek jelenleg 1,4 GB/sec átviteli sebességet tesznek lehetővé.

A SNAP intelligens DMA-vezérlője olyan összetett utasításkód-előreolvasásra és adatelérési műveletekre képes, mint az adattömörítés, lépdelő (strided) adatelérés és a szórás/összegyűjtés, amelyek mindegyike a rendszerkapcsolatok rendelkezésére álló átviteli sávszélességének minél jobb kihasználását célozza. A kapcsolódás hatásfoka több mint tízszer jobb egy átmeneti tárat használó mikroprocesszorénál, amely az ilyen műveletekhez általánosan használt kapcsolatokat alkalmazza.

A SNAP csatlakozhat közvetlenül egyetlen MAP-hez, vagy az SRC Hi-Bar elosztójához több MAP, mikroprocesszor, vagy a közös memória eléréséhez.

### Az SRC-6 rendszer szintű felépítésének megvalósítása

A rendszer-szintű beállítások vagy egy MAP-állomásokból álló fűrtöt, vagy egy kapcsolóval megoldott kereszt-elrendezést valósítanak meg. A 4. ábrán is látható fűrt-alapú rendszerek a mikroprocesszort és a korábban ismertetett DEL-processzort közvetlen kapcsolatban használják. Habár ez az elrendezés a hagyományos és DEL-processzor szoros kapcsolatára épül, mégis ki tudja használni a szabványokra épülő fűrtözési technikát a nagyon gyors rendszerek megvalósításához.

Amikor ennél nagyobb rugalmasságra van szükség, alkalmazhatók a Hi-Bar kapcsolókra épülő rendszerek. A Hi-Bar az SRC saját fejlesztésű skalázható, nagy sávszélességű és kis válaszüjű kapcsolója. Minden Hi-Bar támogatja a 64 bites címzést és 16 bemeneti valamint 16 kimeneti kapuval rendelkezik a 16 csomópont-hoz való csatlakoztatáshoz. A Hi-Bar-hoz csatlakozhatnak mikroprocesszorok, MAP-ek és közös memóriák bármilyen, a 4. ábrán is látható elrendezésben. Minden be- és kimeneti kapu 1,4 GB/sec átviteli sebességgel rendelkezik, amely így egy kettéosztott 22,4 GB/sec sávszélességgé adódik össze a 16 kapun. A kapu-kapu késleltetés 180 ns a kapunkét megvalósított egyszerű hibajavítással és kétszeres hibaezérléssel (SECCED, Single Error Correction Double Error Detection).

A Hi-Bar kapcsolók többsoros elrendezésben is összekapcsolhatók, lehetővé téve, hogy két sor 256 csomópontot kezeljen. Minden Hi-Bar kapcsoló egy 2U magas, 19 hüvelyk széles keretszerelésű készülékben foglal helyet a tápegységével és a hűtőrendszerével együtt, így könnyen beépíthető a kiszolgálókeretekbe.

A Hi-Bar kapcsolókkal megvalósított összekapcsolást használó SRC-kiszolgálók egyesítik a közös memória csomópontokat a mikroprocesszorokkal és a MAP-ekkel. Minden egyes közös memória csomópont saját intelligens DMA-vezérlővel és akár 8 GB DDR SDRAM-mal rendelkezik. Az SRC-6 MAP-, SNAP- és közös memória csomópontjai (CM) 64 bites virtuális memóriacímzést használnak a rendszerben lévő összes memória címzéséhez, ami lehetővé teszi az alkalmazások számára, hogy egyetlen egybefüggő memóriaként kezeljék a teljes rendelkezésre álló tárterületet. Minden csomópont 1,4 GB/s elsőbbségi írási és olvasási átviteli sebességgel rendelkezik. A CM intelligens DMA-vezérlője olyan összetett DMA-



1. táblázat *Karakter-összehasonlítási eredmények*

Megvalósítás	Szövegméret	Minta	Keresési idő	Sebesség
Brute Force (Xeon)	20MB	6	0.827 sec	1.00×
Boyer-Moore (Xeon)	20MB	6	0.597 sec	1.38×
Brute Force (MAP)	20MB	6	0.0143 sec	57.75×
Brute Force (Xeon)	20MB	10	1.398 sec	1.00×
Boyer-Moore (Xeon)	20MB	10	1.051 sec	1.33×
Brute Force (MAP)	20MB	10	0.0141 sec	98.81×

2. táblázat *Egy titkosított szövegben való keresés eredményei*

Megvalósítás	Szövegméret	Minta	Keresési idő	Sebesség
DES-Brute Force (Xeon)	20MB	6	2.77 sec	1.00×
DES-Boyer-Moor (Xeon)	20MB	6	2.63 sec	1.05×
DES- Brute Force (MAP)	20MB	6	0.0143 sec	193.09×
DES-Brute Force (Xeon)	20MB	10	3.31 sec	1.00×
DES-Boyer-Moor (Xeon)	20MB	10	3.11 sec	1.06×
DES- Brute Force (MAP)	20MB	10	0.0143 sec	231.76×

műveletek végrehajtására is képes, mint az adattömörítés, lépdelő adatelérés és a szórás/összegyűjtés a rendszer belső sávszélességének minél hatékonyabb kihasználására. A belső kapcsolatok hatékonysága tízszer akkora, mint az átmeneti tárhelyekre épülő mikroprocesszoré, amely ugyanezt az ilyen műveletekhez általánosan használt kapcsolatot alkalmazza.

Ráadásul az SRC közös memória csomópontjai kizárólagos szemafor-kapcsolással rendelkeznek, amely szintén elérhető Az összes MAP-processzor és mikroprocesszor számára a szinkronizáláshoz.

### Az átszerkeszthető számítási mód programozási modellje

Az RC programozási modellje a hagyományos elgondolás szerint egy hardvertervezési szempont. Mivel az RC alapját képező FPGA-technológia által megkövetelt eszközök az elektronikai tervezési iparból származnak, egy szoftverfejlesztő tényleg nem sok eszközt találhatott ismerősnek ezen a területen. Az eszközök az olyan *hardverleíró nyelveket (HDL)* támogatták, mint a *Verilog*, *VHDL* és a *Schematic Capture*.

A SOC-rendszerek (*system-on-a-chip*, *egylapkás rendszer*) technológiájának megjelenésével és az egyre összetettebbé váló hardverleíró meghatározásokkal elérhető közelségbe kezdtek kerülni a magasszintű programozási nyelvek. A *Java* és *C*-szerű nyelvek használata egyre általánosabb az RC-lapkák programozása terén. Ez egy nagymértékű előrelépést jelent, ami azonban az alkalmazásprogramozóktól is jelentős mértékű váltást követel.

Az SRC-programozási modell egy hagyományos programfejlesztő modell, amelyben a MAP-processzor programozására a *C* és a *Fortran* használatos, és bármely nyelv,

amely összeköthető a futásidejű (C-ben írt) programkönyvtárakkal, lefordítható és futtatható a rendszer mikroprocesszoros részén.

Az SRC Carte programozói környezet az a tervezői feltételezéssel készült, hogy az alkalmazásfejlesztők az RC platformra fogják fejleszteni és átültetni a programjait. Emiatt az SRC-6 rendszerre való fejlesztés során a hagyományos fejlesztési és tervezési elvek érvényesülnek, a *magasszintű programnyelven (HLL)* való kódolás, a programfordítás, a szabványos hibakeresővel való ellenőrzés, a kód javítása. Csak amikor már hiba nélkül fut az alkalmazás egy mikroprocesszoros környezetben, akkor kerül sor a program DEL-processzorra, a MAP-re történő átfordítására.

Egy RC-rendszerre történő programfordítás két olyan lépésből áll, amely meglehetősen idegen az utasításvégrehajtott processzorra történő programozáshoz képest. A HLL-fordítóprogram kimenetének egy hardverdefiniációs nyelvnek kell lennie. A Carte-ben ez vagy a *Verilog* vagy az *EDIF (Electronic Design Interchange Format, elektronikus tervezés adatsere-formátuma)*.

Az EDIF-fájlok azok a hardverleíró objektumfájlok, amelyek az RC-lapokban megvalósított áramköröket írják le. Ha a kimenet *Verilog*, akkor ezt a HDL-t EDIF formátumúra kell hozni egy olyan *Verilog*-fordítóval mint a *Synplicity Synplify* nevű programja. Egy utolsó lépés, az elhelyezés és útvonalkijelölés, veszi az EDIF-fájlokból álló készletet és létrehozza az RC lapkán az áramkörök fizikai elrendezését. A folyamat bemeneti fájllai olyan konfigurációs bitfolyamok, amelyek az FPGA-ba tölthetők az RC-processzorba programozandó algoritmus hardveres kialakításának létrehozása céljából.

A C vagy Fortran nyelvből az FPGA által használható bitfolyammá történő fordítást a Carte programozói környezet végzi el anélkül, hogy a programozónak a folyamatba bele kellene avatkoznia. A program a mikroprocesszorokba szánt kódokat tovább fordítja objektummodulokká.

A Carte számára az utolsó lépés annak az egységesített futtatható fájlak a létrehozása, amely egyetlen linuxos futtatható fájlá olvassa össze a mikroprocesszor objektummoduljait, a MAP-bitfolyamokat, és az összes szükséges futtatási programkönyvtárat. A 6. és 7. ábrán a Carte fordító folyamatát láthatjuk.

### A nyílt forráskód hardver-lehetőségei

A Linux élen járt és jó hasznot húzott a nyílt forráskód mozgalmából, amelynek során a programfejlesztők egy elkötelezett csoportja létrehozta, és továbbfejlesztette a Linux rendszermagját, mégpedig az újítások és minőség olyan szintjén, ami nem mérhető egyetlen kereskedelmi programokkal foglalkozó vállalathoz sem. Az átszerkeszthető számítási módban megvan annak a lehetősége, hogy a hardver tervezési szintjén használja ki ezeket az újításokat és technikai előnyöket. Ennek a cikknek a jelentős

része azzal foglalkozik, hogy bemutassa azt az elvet, ahogyan az alkalmazások programozói a szabványos programozói eljárásokat használva hozzák létre az alkalmazástól függő hardvert, anélkül, hogy ismerniük kellene a hardver felépítését. Az RC-ben ennek ellenére az alkalmazásprogramozók által létrehozott építőelemek a funkcionális egységek. Ezek az egységek olyan számítási alapműveleteket végeznek, mint az összeadás, lebegőpontos szorzás vagy a trigonometrikus függvények. A funkcionális egységek ezen felül lehetnek olyan speciális nagyteljesítményű egységek is, mint a háromszoros titkosítást (DES) megvalósító függvény vagy egy olyan, nem szabványos pontosságú aritmetikai egység, mint a 24 bites IEEE lebegőpontos operátorok.

A funkcionális egységeket a logikai tervezők hozzák létre. Az RC-fordítóprogramok, mint amilyen az SRC Carte MAP fordítója, képesek arra, hogy lehetőséget biztosítsanak a fordítóprogram által támogatott szabványos operátorok mellett felhasználó által létrehozott funkcionális egységek hozzáadására. Ha ilyen újszerű és eredeti funkcionális egységeket teszünk elérhetővé az alkalmazásprogramozók számára, még nagyobb teljesítmények válhatnak elérhetővé.

A funkcionális egységek újszerű hardverfelépítésének létrehozása az a terület, ahol a nyílt hardver mozgalom jelentős előrelépést hozhat a számítástudományba. Az újítás és termékenység, amit a nyílt forráskód területén tapasztalhatunk, most a nyílt hardver köntösében jelenhet meg újra. Az RC még nagyon sok kreatív tervezőnek kínál eszközt arra, hogy új és eredeti hardvert hozzon létre, amit azután az alkalmazásfejlesztők hasznosíthatnak. Az Opensources.org-hoz hasonló csoportokon keresztül pedig megoszthatók és továbbfejleszthetők ezek a funkcionális egységek. Az a jelentős előny, ami a nyílt forrású programoknak köszönhetően a számítástudományban megmutatkozott, könnyen jelentkezhet egy nyílt hardverre összpontosító mozgalomban is.

### Kódolási példa

A DEL-processzorban rejlő teljesítményelőnyt egy karakterlánc-összehasonlító példán keresztül fogom megvilágítani. A példához tartozó forráskód letölthető a *Linux Journal FTP*-oldaláról (lásd a kapcsolódó címekeket). A példa *Christian Charras* és *Thierry Lecroq* honlapjáról származik és a *NIST Dictionary of Algorithms and Data Structures* (az algoritmusok és adatszerkezetek NIST szótára) hivatkozik rá. Összehasonlításképpen a „nyers erő” és a *Boyer-Moore* féle karakterlánc-összehasonlító algoritmust valósítjuk meg egy 2,8 GHz-es *Intel Xeon* processzoron az *Intel C++ 8.0 Linuxos* fordítóprogramjával. A „nyers erő” algoritmust az SRC-rendszerhez a *Carte 1.8* verziójú programozói környezettel állítjuk elő. A „nyers erő” algoritmus egy egyszerű karakterenkénti összehasonlítást végez a karakterlánc és a minta között. A *Boyer-Moore* algoritmust tekintik a leghatékonyabb karakterláncösszehasonlító eljárásnak. A példában egy 20 MB-os véletlenszerűen előállított szövegben keresünk hat illetve tíz mintát. A fordításokat -O3 optimalizáló beállítással végeztük. Az összehasonlító eredményeket az 1. táblázat tartalmazza. További keresési mintákat adva a feladathoz

a mikroprocesszor végrehajtási ideje megnövekszik, azonban a MAP futási idejére nincs hatással, köszönhetően a párhuzamos végrehajtásnak. Bár a *Xeon 2,8 GHz*-en fut, a MAP pedig 100 MHz-en, a DEL párhuzamossága mégis 99-szeres teljesítményelőnyt biztosít a MAP számára. A példa a MAP egyetlen FPGA-jának 60%-át vette igénybe. Egy kétlapkás összeállítás több mint 200-szoros teljesítményt eredményezne.

Annak a bemutatására, hogy egy további számítás hozzáadása a csővezetékkel ellátott ciklushoz miként befolyásolja a teljesítményt és hogy szemléltetni tudjam az egyedi funkcionális egységek képességeit, egy másik összehasonlítást is elvégeztem, amelyben a keresési eljárás egy DES-titkosítással kódolt szöveggel teszteltem. A szöveget a keresés előtt dekódolni kell. A MAP megvalósításban egy DES csővezetékes egységet alkalmaztam. A Verilog nyelvű leírás az *Opencores.org* oldalról származik, ezt építettem be a keresési ciklusba. Mivel a ciklus csővezetéket alkalmaz, órajelenként egy egész eredményhalmazt szolgáltat. Emiatt a 20 MB-os szövegben való keresés és a DES-dekódolás becsült ideje nem változik a keresések számának növelésével. Ez vezet a megdöbbentő 232-szeres sebességnövekedéshez a mikroprocesszoros megoldással szemben. A 10 mintás keresés MAP megvalósítása csak az FPGA teljesítményének 70%-át vette igénybe, így egy kétlapkás felépítés 460-szoros többletet eredményezne.

A Xeon processzoron megvalósított dekódolás *Stuart Levy* (*Minnesota Supercomputer Center*) optimalizált kódját alkalmazta.

### Összegzés

Cikkemben elmagyaráztam az átszerkeszthető számítási mód elvét, valamint példákat mutattam a módszerekre és az elérhető eredményekre. Látható, hogy nagyon jelentős teljesítménytöbblet érhető el ilyen módon. Jelenleg az RC nagymértékben hozzájárulhat a számítástudomány további fejlődéséhez és a jövő is sokkal többet tartogat a számunkra, mint az a mikroprocesszorok fejlődésének Moore-törvénye alapján várható lenne. Az RC már most elérhető a programozók számára, akik használhatják a megszokott fejlesztői modellt, és egy olyan keretrendszer is rendelkezésre áll, amellyel a hardvertervezők szélesebb köre is bekapcsolódhat a nagyteljesítményű számításokba a nyílt forrás biztosította kreativitás és alkotóerő kihasználásával. Az RC már hosszú ideje jelen van, a jelenlegi szoftver- és hardvertechnológia megteremtette annak a lehetőségét, hogy minden számítógépnek a részévé váljon a beépített processzoroktól a Peta-Scale szuperszámítógépekig.

*Linux Journal* 2005. január, 129. szám

Kapcsolódó címek: ➔ [www.linuxjournal.com/article/7867](http://www.linuxjournal.com/article/7867)



**Dan Poznanovic** (poz@srccomp.com) a szoftverfejlesztés vezetője az SRC Computers-nél. A nagyteljesítményű számítások kutatásában 1987 óta vesz részt, amikor a Cray Research céghez csatlakozott.