2.7 KOMPLEKSNA PROGRAMIRLJIVA POLJA

2.7.1 DIGITALNA PROGRAMIRLJIVA LOGIČNA POLJA

Splošne značilnosti

Digitalno elektronsko vezje je pogostokrat izvedeno s standardnimi digitalnimi integriranimi vezji, vendar ga lahko realiziramo še na več drugih, sodobnejših načinov. Odvisno od zahtevnosti in števila logičnih funkcij, prostorskih omejitev, področja vgradnje ali drugih specifičnih zahtev izberemo primerno izvedbo (prostoprogramirljivi krmilnik, namensko integrirano vezje, mikrokontroler,…). Za zahtevnejše logične in aritmetične operacije ali procesiranje podatkov oz. signalov, pa so primerna le kompleksna programirljiva logična polja.

ASIC-vezje lahko načrtamo in izdelamo za točno določen namen (**ASIC**- ***A**pplication **S**pecific **I**ntegrated **C**ircuit*), kjer se vse operacije izvedejo v enem integriranem vezju. Takšno vezje vsebuje polje omejenega števila osnovnih digitalnih gradnikov (vrata , FF,…) poleg tega pa tudi določeno število transistorjev, diod, nizkoohmskih in visokoohmskih uporov, kondenzatorjev (z lebdečim potencialom), kateri so razmeščeni na obrobju substrata. V sredinskem polju je množica prekrižanih metaliziranih povezav, ki omogočajo kakršnokoli medsebojno povezavo vgrajenih komponent. Pri programiranju se enostavno spojijo s programom predvideni prekrižani spoji. Na ta način dosežemo najboljše delovanje vezja, vendar je takšen pristop časovno in finančno zahteven, saj sta za izvedbo potrebni veliko znanje in vrhunska (draga) polprevodniška tehnologija. Poleg tega je takšen ASIC neprilagodljiv za naknadne spremembe v vezju, kar vodi k ponovitvi postopka načrtovanja in izdelave. Vendar so se v novejšem času pojavile specializirane firme, ki imajo potrebno opremo in lahko naredijo že za relativno nizko ceno tudi pri malih serijah želena vezja v ASIC izvedbi.

MIKROKONTROLERJI, katerih delovanje temelji na izvajanju napisanega programa, so v primerjavi z ASIC vezji izredno prilagodljivi na spremembe, ki jih lahko enostavno vnesemo s spremembo programa. Vendar je zaradi prilagodljivosti bistveno slabša zmogljivost digitalnega vezja, ker mora procesor vsak ukaz najprej prebrati iz pomnilnika, nato določiti njegov pomen in šele nato se lahko izvede. Slabost je tudi v omejenem naboru ukazov. V primeru, da za neko operacijo ne obstaja v naprej določen ukaz, moramo takšno operacijo izvesti z zaporedjem obstoječih ukazov, kar dodatno zmanjšuje zmogljivost.

CPLD, FPGA in **TRAC** spadajo med programirljiva vezja, ki združujejo prednosti obeh načinov in jih je možno programirati preko PC računalnika ob uporabi ustrezne programske opreme in posebej zato pripravljenega jezika za načrtovanje sodobnih digitalnih vezij (**VHDL**- *Very High Digital Language*). Programirljiva vezja se danes v industriji veliko uporabljajo za izdelavo digitalnih vezij - FPGA, pa tudi analognih – TRAC, predvsem tam, kjer za določeno funkcijo ne obstaja že narejeno integrirano vezje, mikrokontroler pa ni dovolj zmogljiv ali primeren za opravljanje takšnih funkcij. Najbolj zmogljiva programirljiva vezja so sicer predraga, da bi jih vgrajevali v velikoserijske produkte, se pa njihova cena stalno znižuje. Trenutno velja pravilo, da je uporaba programirljivih vezij upravičena za izdelavo digitalnih vezij velikosti do 50.000 vrat, ki delujejo s frekvenco do 50MHz in za serije, ki ne presegajo 50.000 kosov. Kadar so zahteve večje velja razmisliti o izdelavi ASIC vezja.

2.7.1.1 Vrste in značilnosti programirljivih vezij

Programirljiva logična vezja delimo na **enostavna**, **kompleksna** in **električno programirljiva** polja logičnih vrat.

PAL, **PLA** in **GAL** spadajo med enostavna programirljiva logična vezja. Ta vezja temeljijo na izvedbi logičnih funkcij v dvonivojski obliki in vsebujejo programirljivo IN-ALI matriko. Zmogljivejša vezja (GAL) vsebujejo tudi flip-flope, povratne zanke in tristanjske izhode. Enostavna programirljiva vezja se uporabljajo kot univerzalni nadomestek za več standardnih TTL ali CMOS vezij nizke ali srednje stopnje integracije.

CPLD- (*Complex Programable Logic Device*) so kompleksna programirljiva vezja in so zmogljivejša od enostavnih in sestavljena iz več PLD stuktur, katere so zelo podobne enostavnim programirljivim vezjem, ter programirljivega povezovalnega polja. Na povezovalnem polju je na voljo omejeno število medsebojnih povezav med PLD strukturami ter povezav z zunanjimi priključki. Pri načrtovanju programirljivih vezij potrebujemo programsko opremo za razdelitev vezja, za dodelitev posameznih funkcij notranjim strukturam in za ustvarjanje povezav v povezovalnem polju. Njihova struktura je, podobno kot pri enostavnih programirljivih vezjih, optimizirana za izvedbo funkcij v dvonivojski obliki . Zaradi takšne strukture imajo CPLD vezja predvidljive zakasnitve in so zelo hitra, vendar manj zmogljiva pri izvedbi kompleksnih večnivojskih funkcij kot električno programirljiva polja logičnih vrat.

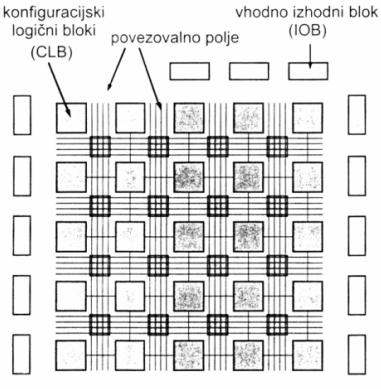
FPGA-(Field Programable Logic Array) spadajo med električno programirljiva polja logičnih vrat in vsebujejo matrike programirljivih logičnih blokov, povezovalnih virov in vhodno izhodnih blokov. Značilnejši proizvajalci FPGA vezij so Xilinx, Altera, Atmel in Actel. Primer načrtovanja bo predstavljen na osnovi vezij in programske opreme Xilinx.

2.7.1.2 FPGA- struktura vezij in programiranje (Xilinx)

Ta programirljiva vezja so sestavljena iz matrike konfiguracijskih logičnih blokov, ki so med seboj povezani s programirljivimi povezavami (slika 1). Z logičnimi bloki je možno realizirati logične funkcije, ki vključujejo tudi flip-flope.

FPGA vezje je sestavljaeno iz sledečih blokov:

- **konfiguracijski logični bloki** (CLB), ki so razporejeni v matriko,
- **programirljiva mreža** povezav med bloki,
- **vhodno/izhodni bloki** na zunanjem robu, na katere so vezani zunanji priključki.



Slika 2.32: Blokovna struktura FPGA

Vsak konfiguracijski blok ima po dva 4-vhodna in en 3-vhodni funkcijski generator, s katerimi je možno narediti poljubno logično funkcijo štirih oz. treh vhodov (slika 2). Na vhodu sta dva generatorja prenosa, ki omogočata hitrejšo izvedbo seštevalnikov, števcev, ipd..

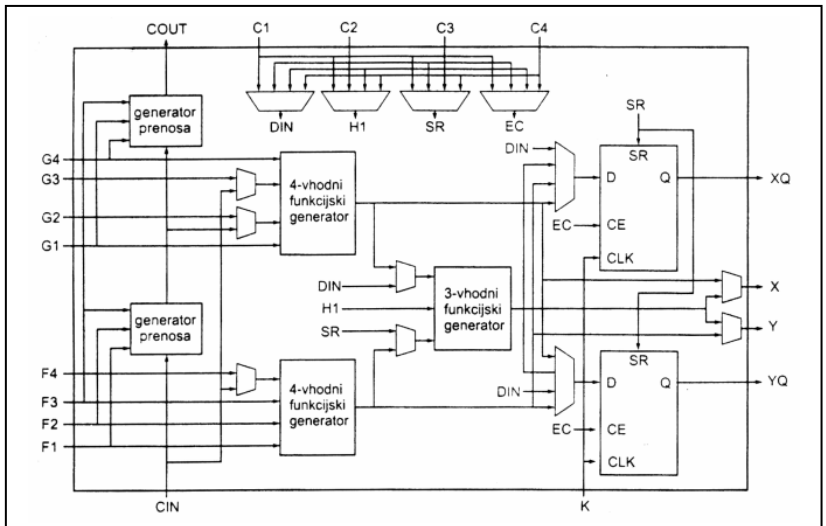
Vsak 4-vhodni funkcijski generator je možno uporabiti tudi kot mali pomnilnik (Select RAM) za 32x1 ali 16x2 bita.

Vsak konfiguracijski logični blok ima še dva flip-flopa s kontrolnimi signali za omogočanje takta in asinhrono nastavljanje izhoda. Vezja imajo glede na izvedbo od 20.000 do 1.000.000 logičnih vrat.

Postopek načrtovanja vezja v FPGA

Klasični način načrtovanja (sinteze) digitalnih vezij poteka z risanjem sheme vezja na nivoju vrat, kompleksnejših komponent in celotnega sistema. Shema vezja zelo pregledno prikazuje relacije med posameznimi gradniki, vendar pa shematsko načrtovanje postane neprimerno pri zelo kompleksnih digitalnih vezjih, ki so sestavljena iz velikega števila gradnikov.

V tem primeru je primerna uporaba visokonivojskih jezikov, kot je npr. ***VHDL***, s katerim opišemo delovanje vezja.



Slika 2.33: Zgradba konfiguracijskega logičnega bloka

Z VHDL jezikom lahko na hiter in pregleden način vnesemo vezje (zakonitost) in opravimo simulacijo delovanja. Računalniški program za sintezo vezja, nam iz VHDL opisa vezja, generira vezje na nivoju logičnih vrat. Datoteka na nivoju logičnih vrat je osnova za izdelavo vezja v izbrani tehnologiji (npr. monolitna, programirljiva vezja,…)
V jeziku VHDL je npr. dvovhodni multiplekser opisan s sledečim stavkom:

Izhod <= vhod0 ***when*** *izbira=`0` else vhod1*;

Stavek pove, da bo izhodni signal enak signalu vhod0, kadar bo imel signal izbira vrednost 0, sicer pa bo na izhodu signal iz vhoda 1. Signali so najosnovnejši gradniki jezika VHDL in predstavljajo vhodne in izhodne priključke ter notranje povezave v vezju. Vodila so opisana z vektorskimi signali, ki jim je pri deklaraciji določena velikost. Zapis dvovhodnega multipleksorja je enak za enobitno ali pa npr. za 8- bitno konfiguracijo, če so signali vhod0, vhod1 in izhod definirani kot 8-bitni vektorji.
To ponazarja moč VHDL jezika, s čimer lahko na precej enostaven način zapišemo dokaj kompleksne strukture. Načrtovanje kompleksnih vezij je zato precej hitrejše in učinkovitejše v primerjavi z risanjem sheme.

Procedura programiranja FPGA

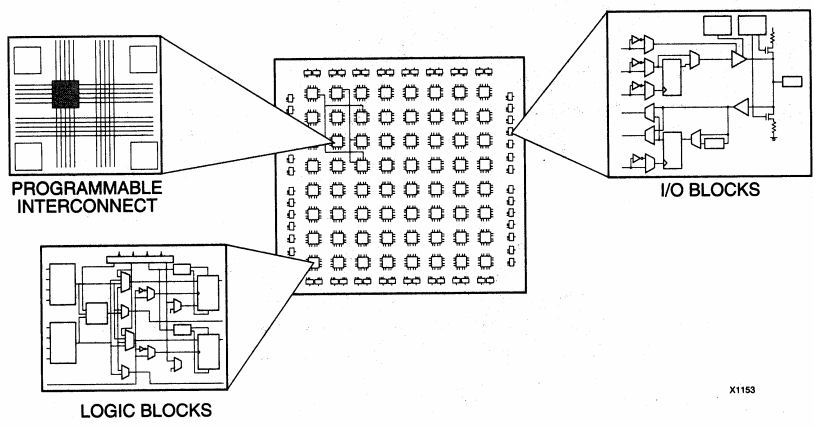
Opis in **simulacijo** delovanja vezja v VHDL jeziku omogoča drugo programsko orodje *Active HDL*, ki ima zelo zmogljiv urejevalnik VHDL kode (z barvanjem sintakse, avtomatskim zamikanjem, dopolnjevanjem ključnih besed, …). Simulacijo vezja izvedemo tako, da VHDL kodo prevedemo, nastavimo vhodne signale in poženemo simulator. Vhodni signali so lahko vnaprej definirane oblike signalov (npr. takt, števci,…), konstantne vrednosti, ki jih lahko med simulacijo spreminjamo ali pa njihov časovni potek zapišemo s formulo.

Za **Sintezo vezja** iz VHDL opisa uporabimo program *Synopsis FPGA Express*, ki naredi iz VHDL datoteke novo datoteko (netlisto) z vezjem na nivoju logičnih vrat, flip-flopov in ostalih struktur, ki so na voljo v izbrani tehnologiji.

Programiranje FPGA omogoča program *Xilinx Design Manager*, kamor uvozimo datoteko z vezjem in datoteko z uporabniškimi zahtevami (User Constraint File). V datoteki z uporabniškimi zahtevami določimo npr. tudi maksimalno frekvenco in razporeditev vhodnih oz. izhodnih signalov na želene priključke FPGA. Postopek prevajanja je relativno obširen, saj poteka v šestih korakih.

V prvem koraku prevajalnik zbere vse datoteke in opravi optimizacijo vezja. Nato sledi tehnološka preslikava v kateri se logična vrata preslikajo v funkcijske generatorje, ki bodo opravljali želeno funkcijo. V nadaljevanju program razmesti in poveže konfiguracijske logične bloke in opravi časovno analizo vezja, kjer se izračunajo zakasnitve vseh signalov znotraj FPGA. Zadnji korak je izdelava konfiguracijske datoteke s katero programiramo.

Med postopkom dobimo tudi informacijo o zasedenosti vezja in kot rezultat časovne analize še VHDL datoteko z modelom prevedenega vezja in datoteko z izračunanimi zakasnitvami. Obe datoteki lahko prenesemo nazaj v začetni programski paket *Active HDL* in opravimo simulacijo vezja z realnimi zakasnitvami.



Slika 2.34: Notranja struktura FPGA vezja (Xilinx)

2.7.2 ANALOGNA PROGRAMIRLJIVA POLJA

Splošne značilnosti

TRAC programirljiva vezja predstavljajo del družine analognih programirljivih vezij (**FPAD**-*Field Programable Analog Device*), ki omogočajo enostavno pot reševanja problemov od načrtovanja vezij do same izdelave analognih integriranih vezij za specifične potrebe. Podobno kot pri FPGA vezjih so tudi na tem področju na voljo ustrezna programska orodja, ki omogočajo hitro načrtovanje, simulacijo delovanja in programiranje vezja. Ko so rezultati simulacij vezja takšni kot smo si jih zamislili oz., ko je odziv vezja res takšen kot ga potrebujemo, naložimo kodo načrta v čip in s tem je delo končano. Vezje omogoča vnašanje zakonitosti oz. postopka obdelave analognih vhodnih signalov po matematični poti. V primerjavi s klasičnimi analognimi integriranimi vezji, kjer obdelava vhodnih signalov temelji na karakteristikah posameznih vgrajenih komponent, je pri TRAC vezjih to izvedeno preko osnovnih matematičnih operatorjev. V bistvu lahko isto prenosno funkcijo izvedemo na najvišji ravni abstrakcije, torej na nivoju matematike. Matematično pot načrtovanja omogoča hardver, ki je sposoben izvajanja določenega števila matematičnih operacij. S pomočjo smiselnih povezav operacij oz. operatorjev je možna realizacija katerekoli analogne funkcije. Osnovnih operacij, ki jih lahko preko kodiranja vnesemo v posamezne operatorje je 8: ADD, NEGATE, LOG, ANTI-LOG, AUX (ojačevalnik, diferenciator, integrator) in REC (usmernik)

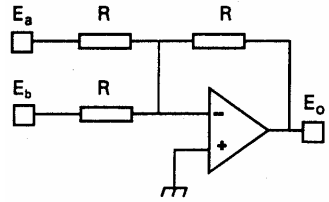
Med najbolj pogostimi matematičnimi operacijami pri obdelavi signalov je zagotovo množenje. Množenje je v digitalni tehniki algoritemsko relativno enostavno izvesti, vendar je zato potrebno signale pretvoriti iz analognega v digitalni prostor, kar ni vedno zaželjeno. TRAC vezja isto nalogo uspešno izvedejo kar v analognem okolju. Potrebno je le zagotoviti pomnilnik za trajno ohranjanje programa, kot je EEPROM ali baterijsko podprt RAM. Seveda v primerih povezave z mikrokontrolerjem pomnilnik že obstaja in ga lahko uporabimo tudi za TRAC.

Opis posameznih osnovnih funkcij

ADD (koda 011)

Funkcija seštevanje je izvedena z operacijskim ojačevalnikom in tremi enakimi upori. Na podlagi navidezne mase na invertiranem vhodu, lahko zapišemo sledeč izraz:

$$E_o = -R(E_a /R + E_b /R) = -(E_a + E_b)$$



Slika 2.35: Struktura analognega seštevalnika

NEGATE (koda 010)

Funkcijo invertiranja dobimo prav tako iz seštevalnika le, da imamo uporabljen samo en vhod kot invertirajoč ojačevalnik: $E_o = - E_b$

NON INVERTING PASS (koda 100)

Funkcija je uporabjena zgolj zaradi topologije in zagotavlja le prevodno pot skozi celico brez kakršnegakoli modificiranja signala (ojačevalnik z ojačanjem 1)

$$E_o = E_b$$

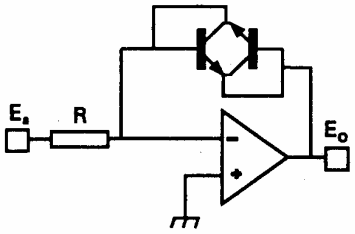
LOG (koda 110)

Funkcija logaritmiranja je izvedena s pomočjo operacijskega ojačevalnika in dveh antiparalelnih diod v povratni vezavi. Navidezna masa na invertirajočem vhodu omogoča sledeči zapis:

$$E_o = -kT/q \log (E_a/R\, I_o + 1)$$

kjer so:

k: *Boltzmanova konstanta;* **T**: *absolutna temperatura,*
q: *naboj elektrona;* **I_o** *: tok zasičenja PN spoja*



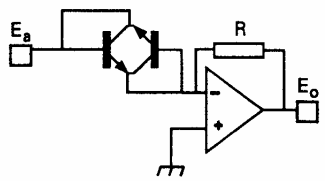
Slika 2.36: Struktura vezja za logaritmiranje

ANTI-LOG (koda101)

Funkcija antilogaritmiranja je izvedena podobno kot logaritemska le, da sta upor in antiparalelni diodi na zamenjanih mestih. Navidezna masa na invertirajočem vhodu omogoča sledeči zapis:

$$E_o = -R\, I_o + (\exp q\, E_a/kT-1)$$

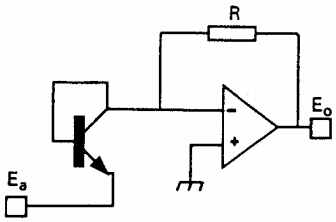
V primeru, da signal spustimo skozi logaritemsko in antilogaritemsko celico, se tok zasičenja in absolutna temperatura izničita.



Slika 2.37: Struktura vezja za antilogaritmiranje

RECTIFIER (koda 111)

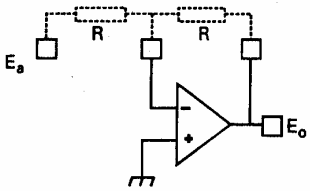
Dioda na vhodu operacijskega ojačevalnika omogoča realizacijo usmernika.



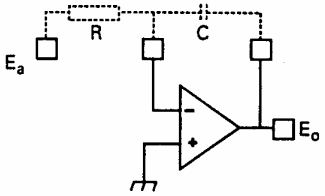
Slika 2.38: Struktura usmernika

AUX (koda001)

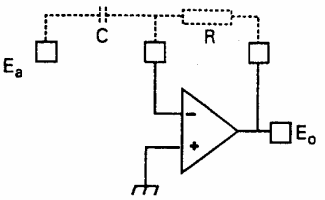
Je pomožna funkcija in omogoča izbiro želene funkcije z nastavitvijo zunanjih elementov. Izbiramo lahko med ojačevalnikom, diferenciatorjem ali integratorjem.



Slika 2.39: Struktura ojačevalnika



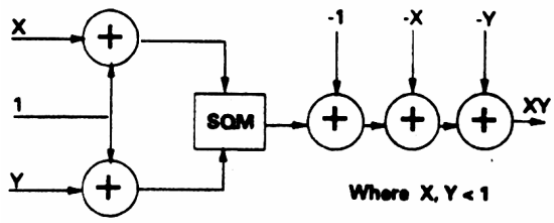
Slika 2.40: Struktura integratorja



Slika 2.41:Struktura diferenciatorja

OFF (koda 000)

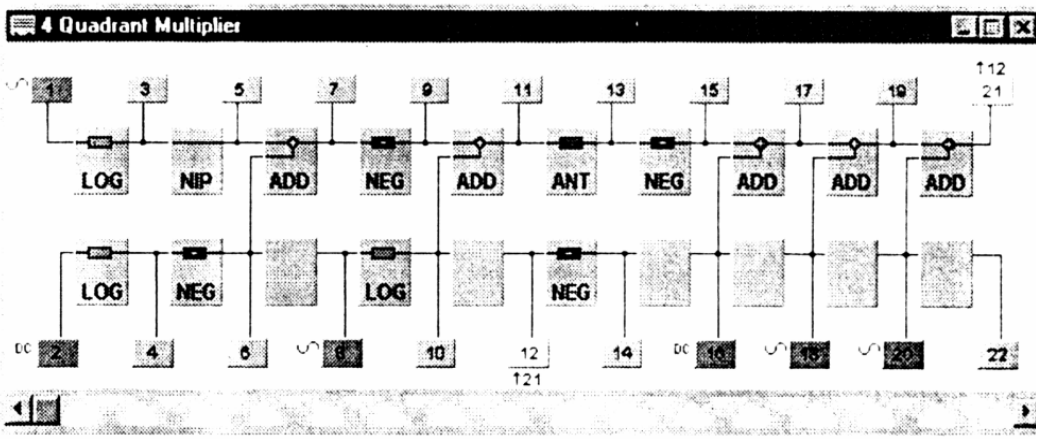
Ta koda preprečuje pot signalu skozi celico.



Slika 2.42: Algoritem 4-kvadrantnega množilnika

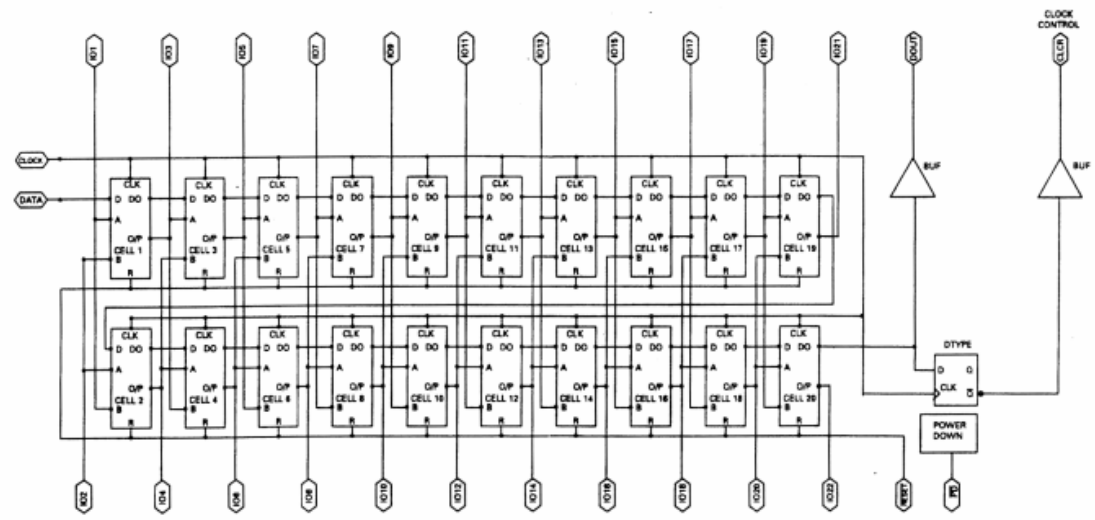
Primer:

Program, ki omogoča 4-kvadrantno množenje analognih napetosti

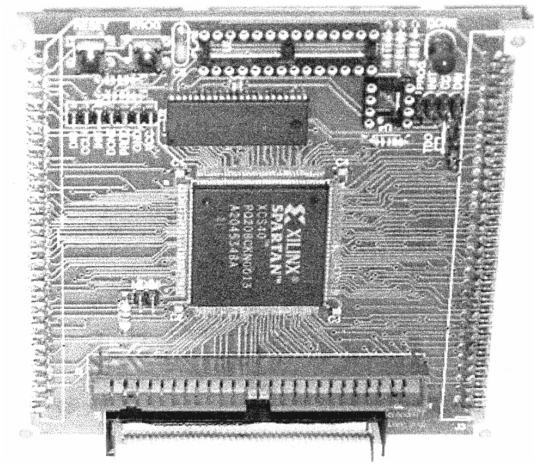


Slika 2.43: Primer shematskega pogleda na programirane celice za funkcijo 4-kvadrantnega množilnika (ZETEX)

Notranja struktura TRAC vezja



Slika 2.44: Blokovna shema TRAC vezja (ZETEX)



Slika 2.45: FPGA na razvojnem modulu

Za bolj poglobljeno delo s temi vezji je na spletnih straneh mnogo instrukcijskega gradiva s primeri programiranja in podrobnejšimi navodili. Proizvajalci teh vezij nudijo različno podporo, nekaj od tega je mogoče najti na sledečih spletnih straneh:

http://www.fpga-advantage.com
http://www.altera.com/support/examples/vhdl
http://www.xilinx.com