

REALISATIE-TECHNIEKEN VOOR DIGITAAL ONTWERP

Verschillende mogelijkheden om een digitaal ontwerp praktisch te realiseren:

- **VASTE LOGICA**
 - gebruik maken van bestaande TTL en/of CMOS IC's
- **CUSTOM DESIGN IC'S**
 - een IC speciaal voor dit ontwerp laten fabriceren
 - drie mogelijkheden:
 - » STANDARD CELLS
 - » GATE ARRAYS
 - » FULL CUSTOM
- **PROGRAMMABLE LOGIC**
 - = 'best of both worlds' ?

Realisatie-techniek 1 : VASTE LOGICA

VASTE LOGICA:

- conventionele TTL/CMOS logica
- populair, goed gekende bouwblokken
- lage eenheidsprijs
- **bouwblok = beperkte functie (SSI/MSI)**
 - » veel IC's nodig
 - » grote PCB's
 - » veel powerdissipatie
 - » veel stock
 - » verhoogde prijs eindproduct
- **(detail)wijziging in ontwerp**
 - » leidt meestal tot een nieuwe lay-out van de PCB

Realisatie-techniek 2 : CUSTOM DESIGN IC

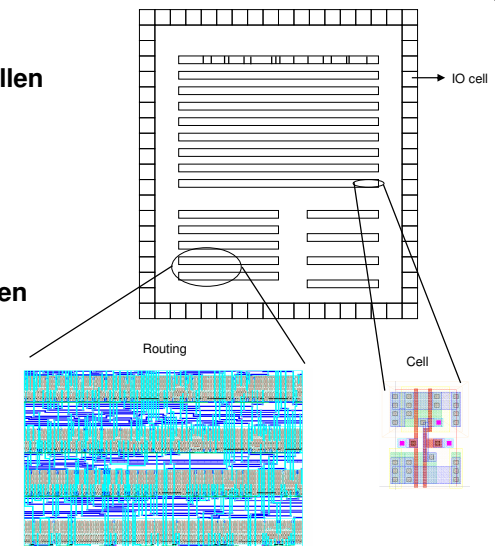
CUSTOM DESIGN IC : ALGEMEEN

- volledige digitale functie in 1 IC onderbrengen = optimale Si-benutting
- flexibel : wat nodig is (en juist dat) kan geïntegreerd worden
- lange ontwikkeltijd en dus hoge NRE kost = Non-Recurring Engineering
- hoge productiekosten
- lage eenheidsprijs bij (zeer) grote reeksen
- fout of wijziging in design → volledig opnieuw !
- ASIC = Application Specific IC

Realisatie-techniek 2 : CUSTOM DESIGN IC

• STANDARD CELLS

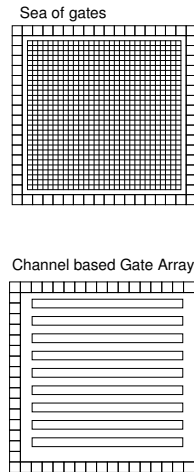
- bibliotheek van functionele cellen
 - » bouwblokken
 - » reeds ontworpen
 - » gegarandeerd foutvrij
 - » vergelijkbaar met TTL-databoek
- alleen in design gebruikte cellen worden geïntegreerd ('place & route')
- geen Si verkwisting
- prijs IC ~ mm² Si
- zeer lange productietijd :
 - » IC's moeten volledig geprocessed worden



Realisatie-techniek 2 : CUSTOM DESIGN IC

• GATE ARRAYS

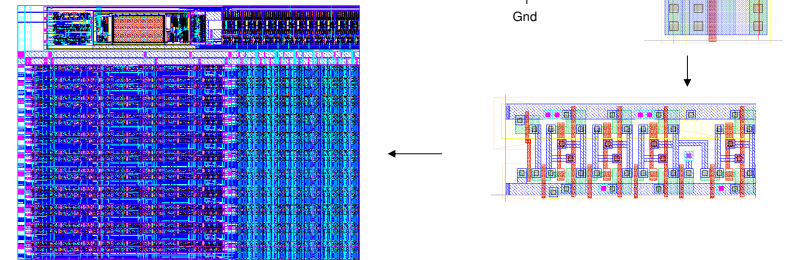
- matrix van identieke cellen, reeds afgewerkt op 'wafer'
- grootte-orde : 500 - 200.000 gates
- klant-specifieke functie wordt gerealiseerd door het patroon van de nog aan te brengen verbindingen tussen deze cellen
- kan resulteren in (relatief) veel onbenut Si
- typische max. bezetting : 75% à 90%
- kortere 'turn-around time' dan bij standard cell
- lagere kostprijs omdat zelfde basispatroon voor meerdere gebruikers geschikt is



Realisatie-techniek 2 : CUSTOM DESIGN IC

• FULL CUSTOM

- alles volledig zelf ontwerpen, tot op transistorniveau
- zeer lange ontwikkelingstijd
- leidt tot optimale Si-benutting
- bij grote reeksen : laagste prijs
- optimale prestatie
- slechts in uitzonderlijke situaties



Realisatie-techniek 3 : Programmeerbare logica

• PLD = Programmable Logic Devices

- = 'best of both worlds' ? (discrete logica & gate-array)
- IC's reeds bij de klant beschikbaar
- gebruiker kan IC zelf 'configureren' m.b.v. ontwerpsoftware (PC) + (evtl.) programmeertoestel
- groot scala aan fabrikanten en componenten
 - » Simple PLD (SPLD)
 - PROM, PLA, PAL
 - GAL
 - » Complex PLD (CPLD) of Erasable PLD (EPLD)
 - » Field-Programmable Gate-Array (FPGA)

Voordelen van Programmeerbare logica

• eenvoud in ontwerp

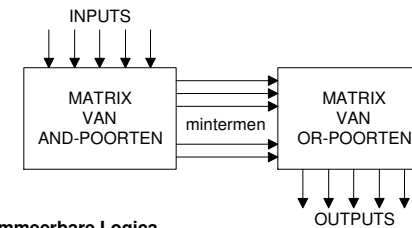
- » computer aided design
- » schematic entry
- » Hardware Description Language (HDL)
 - hoog niveau beschrijving
 - automatische synthese (omzetten naar 'gates')
 - vereenvoudiging
 - ERC & DRC (Electrical/Design Rule Check)
- » simulatie
- » PLD-implementatie
- » onmiddellijk uit te testen

Voordelen van Programmeerbare logica

- grote keuze fabrikanten & componenten
 - » architectuur, density
 - » technologie, snelheid, vermogen
 - » prijs
- eenvoudiger PCB (Printed Circuit Board)
 - » één PLD = tientallen 74xx
 - » grotere flexibiliteit bij keuze van I/O-pinnen
 - » functionele wijziging in PLD = zelfde PCB
 - » hogere betrouwbaarheid (minder componenten, minder verbindingen, minder solderingen)
 - » beter testbaar
 - » lagere totale kostprijs
- design security

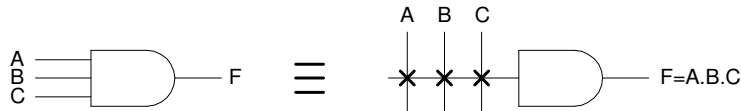
Simple PLD's (SPLD)

- Elke logische functie kan gerealiseerd worden in een "Sum Of Products" (SOP) vorm
- Een SOP vorm kan gerealiseerd worden door INV/AND/OR (of INV/NAND/NAND)
- De bouwblokken van AND/OR gates zijn vooraf gemaakt
- Programmeren = het maken of verbreken van connecties tussen de gates

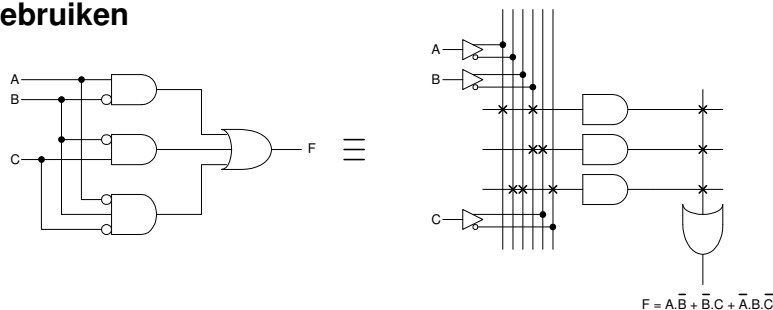


Conventies

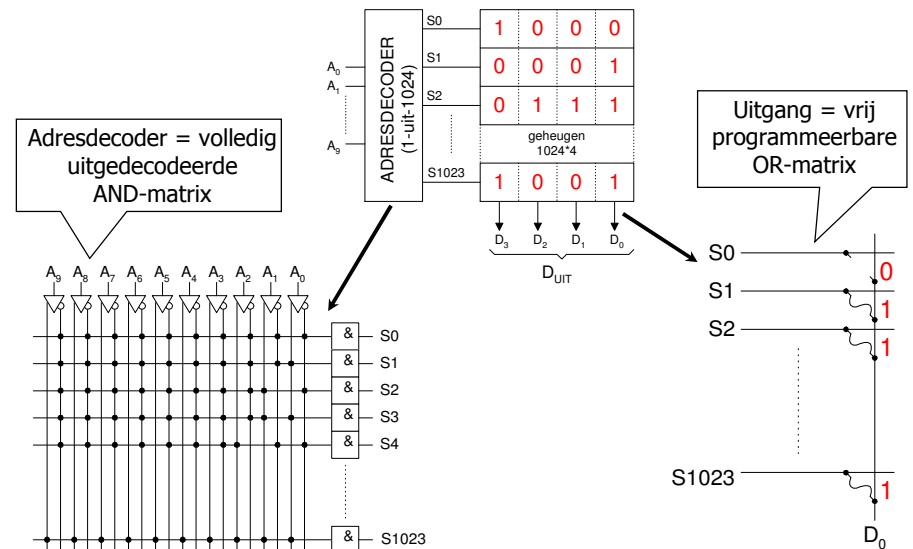
Om het aantal lijnen in het schema te verminderen tekenen we de verschillende ingangen op eenzelfde lijn.



Elke input wordt geïnverteerd zodat we beide kunnen gebruiken



ROM als PLD



ROM als SPLD

- PROM kan gebruikt worden om een aantal logische functies in te programmeren
- zeer geschikt voor bv. decoders
- nadeel: volledig uitgedecodeerde adresdecoder
 - werkt goed als ook alle ingangscombinaties kunnen voorkomen
 - niet efficiënt om meerdere functies te maken die *niet* afhankelijk zijn van *dezelfde* ingangsvariabelen

Voorbeeld: $X=f_1(A,B,S) \rightarrow 8$ mogelijke combinaties

$Y=f_2(C,D,E,S) \rightarrow 16$ mogelijke combinaties

Vereiste PROM: 6 ingangsvariabelen

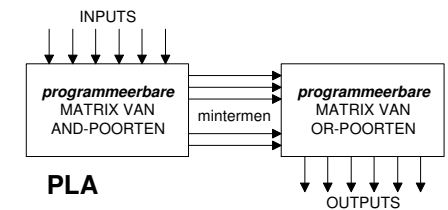
2^6 ingangscombinaties = 64 te programmeren adressen
alhoewel slechts 24 combinaties zinvol zijn

PLA = Programmable Logic Array

- Probleem met PROM: volledig uitgedecodeerde adresdecoder leidt tot veel *redundantie*
 - niet-relevante ingangscombinaties moeten ook geprogrammeerd worden

- Oplossing in PLA:

- AND-matrix is ook vrij programmeerbaar gemaakt
- gebruiker kan mintermen samenstellen met alleen maar de relevante ingangsvariabelen

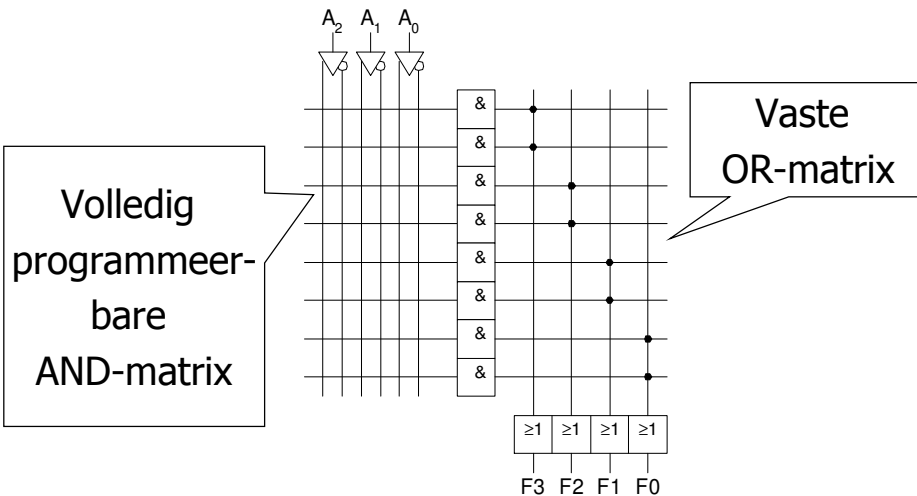


- Goed idee, maar ...

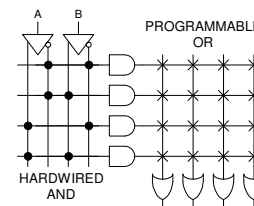
- meer interne logica nodig om beide array's programmeerbaar te maken
- duurder, trager
- nooit echt doorgebroken in de praktijk

PAL = Programmable Array Logic

- PAL = omgekeerd principe van PROM

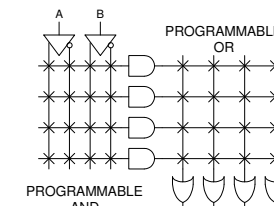


PROM



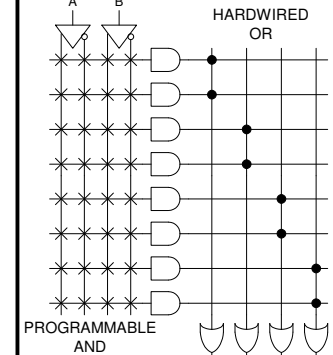
PROM:
vaste AND,
programmeerbare OR

PLA



PLA:
programmeerbare AND,
programmeerbare OR

PAL



PAL:
programmeerbare AND,
vaste OR

PAL NAAMGEVING

PAL 16 L 8 B -4 C N

Number of
array INPUTS

Number of
OUTPUTS

**SPEED
POWER
OPERATING
CONDITIONS
PACKAGE**

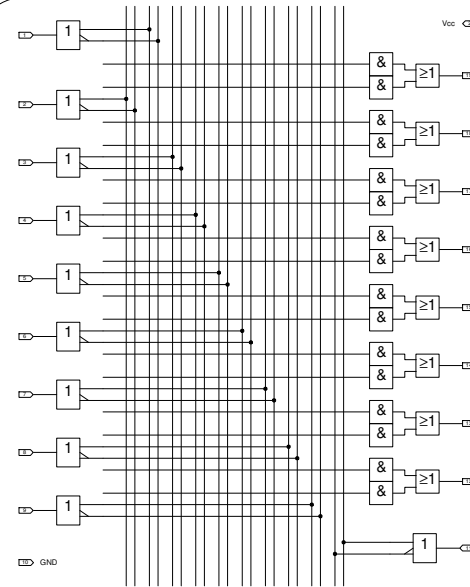
OUTPUT CELL

L=active LOW
H=active HIGH
C=complementary
P=programmable polarity
R=registered
X=exclusive OR
A=arithmetic
...

Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

PAL 10H8



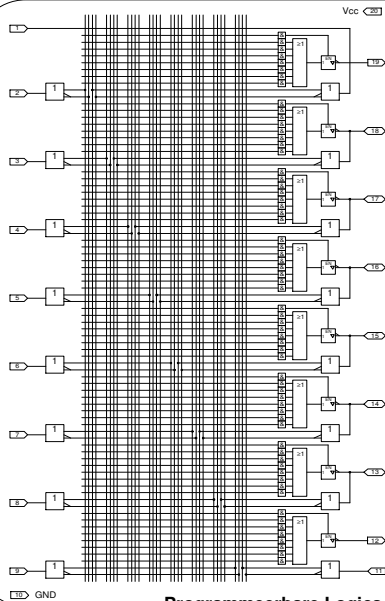
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

Eenvoudigste PAL:

- 10 ingangen
- 8 (actief hoge) uitgangen
- elke uitgang wordt gevormd als de logische som van (slechts) twee mintermen, die kunnen samengesteld worden uit de tien ingangen (en hun inverse)
- 20-pins IC (10 in + 8 uit + Ucc + GND)
- Varianten:
PAL12H6, 14H4, 16H2
PAL10L8, 12L6, 14L4, 16L2

PAL 16L8



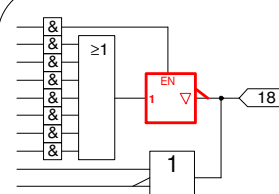
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

Uitgebreidere PAL:

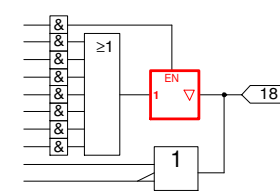
- maximaal 16 ingangen
- maximaal 8 (actief lage) uitgangen
- ook slechts 20-pins IC (16+8=24?)
 - » 10 vaste ingangen
 - » 2 vaste uitgangen
 - » 6 pennen in- of uitgang
 - » bidirectionele pennen worden intern teruggekoppeld naar de AND-matrix
- elke uitgang wordt gevormd als de logische som van zeven mintermen
- uitgang = tri-state inverter
 - » kan in Hi-Z toestand gebracht worden door afzonderlijke minterm
 - » op dat ogenblik kan pen als ingang gebruikt worden
- Uitbreiding: PAL20L8

PAL 16L8



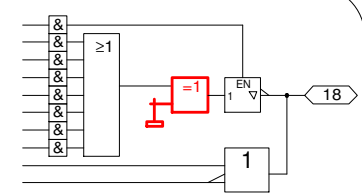
PAL16L8:
actief **LAGE**
uitgangen

PAL 16H8



PAL16H8:
actief **HOGE**
uitgangen

PAL 16P8

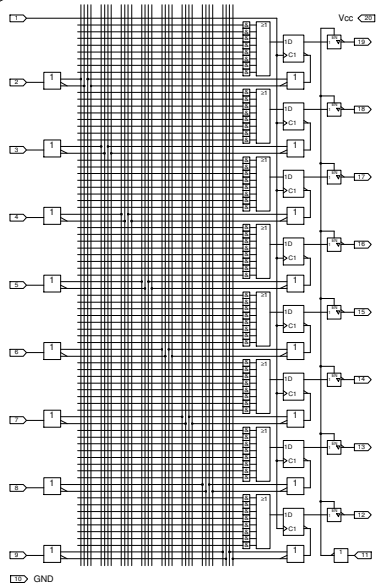


PAL16P8:
Programmeerbare
polariteit d.m.v.
EXOR-poort

Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

PAL 16R8



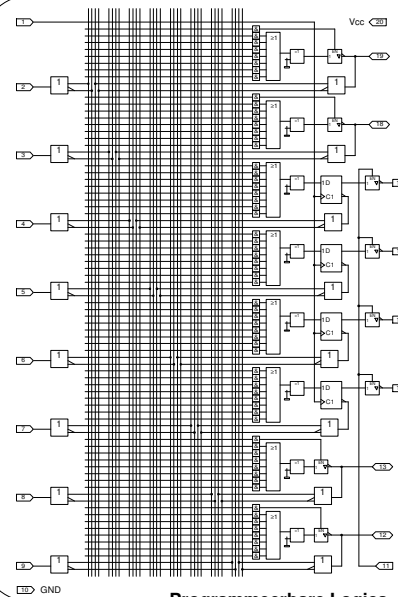
Sequentiële PAL:

- 8 ingangen
- 8 uitgangen, via een 3-state inverter verbonden met de Q-uitgangen van 8 D-flipflops; vandaar de **R** in de naam
- FF-uitgangen worden eveneens teruggekoppeld naar de AND-matrix; daarom blijft de benaming PAL**16**R8 (en niet PAL8R8)
- pen 1 = gemeenschappelijke CLOCK-pen (*synchrone* PAL); flipflops reageren op de stijgende klokflank
- pen 11 = gemeenschappelijke OUTPUT ENABLE-pen
- Varianten: 16R6 en 16R4

Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

PAL 16RP4



'Gemengde' PAL:

- 8 'zuivere' ingangen
- 4 combinatorische uitgangen
 - » logische som van zeven ingangen
 - » programmeerbare polariteit (P)
 - » individuele *output enable*
 - » terugkoppeling naar AND-array
 - » ook bruikbaar als ingang
- 4 sequentiële uitgangen (R)
 - » logische som van acht ingangen
 - » programmeerbare polariteit
 - » gemeenschappelijke *output enable* via één ingangspen
 - » D-flipflop in uitgangslijn met gemeenschappelijke klok via één ingangspen
 - » flipflop-uitgang wordt teruggekoppeld naar AND-array

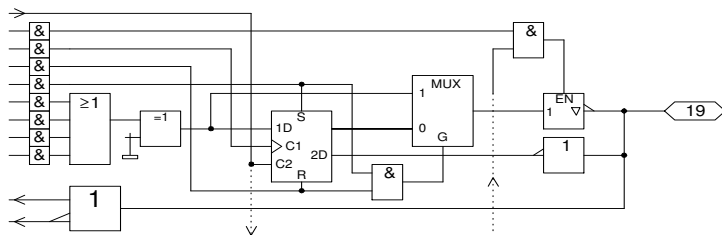
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

PAL 16RA8

Asynchrone PAL:

(1 van de 8 macrocellen)



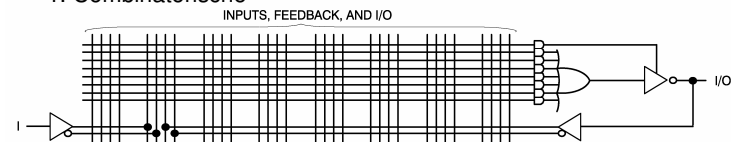
- 8 'zuivere' ingangen
- 8 uitgangen: combinatorisch of sequentieel
 - » logische som van vier ingangen
 - » programmeerbare polariteit
 - » D-flipflop met asynchrone klok (individuele minterm per macrocel)
 - » asynchrone PRESET- en CLEAR ingangen
- keuze seq/comb gebeurt via multiplexer aan uitgang
- individuele of gemeenschappelijke *output enable*
- terugkoppeling naar AND-array, uitgangspen ook bruikbaar als ingang
- 'vrij configureerbaar': principe wordt uitgebreid in GAL en CPLD

Programmeerbare Logica

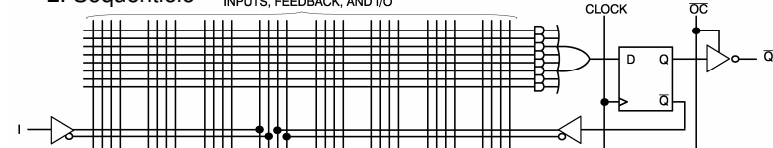
DIGITALE ELEKTRONICA 2EL

Combinatorische en sequentiële PAL

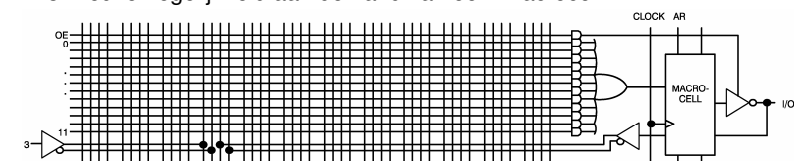
1. Combinatorische



2. Sequentiële



3. Keuzemogelijkheid aan de hand van een 'macrocell'



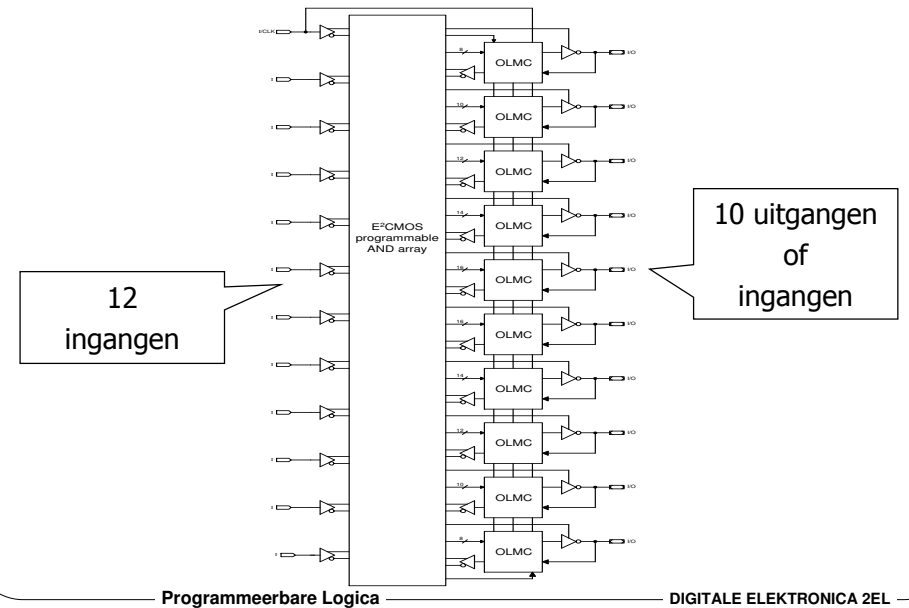
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

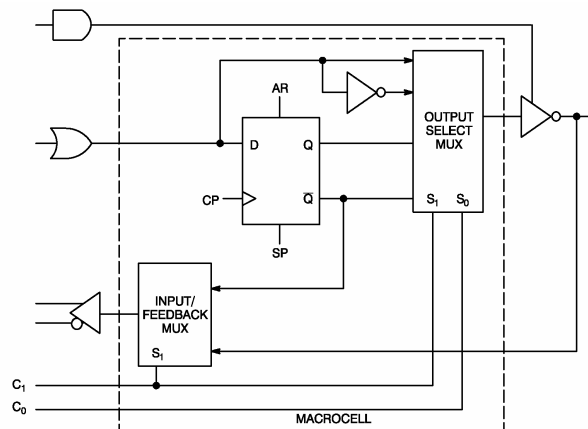
GAL = Generic Array Logic

- GAL = uitgebreide PAL met zeer flexibele uitgangsblok
- gebaseerd op E²CMOS-technologie
 - E² = Electrically Erasable, dus herprogrammeerbaar
 - in tegenstelling tot de éénmalig programmeerbare PAL
- slechts enkele, maar zeer universele types, die elk een ganse reeks van PAL's kunnen vervangen
- GAL16V8 vervangt bijna alle 20-pen PAL's
- GAL20V8 vervangt bijna alle 24-pen PAL's
- GAL22V10
 - is uitgegroeid tot een praktische basiscomponent

GAL 22V10 : structuur



GAL 22V10 : Output Logic MacroCell (OLMC)



Registered/Combinatorial		
C ₁	C ₀	Configuration
0	0	Registered/Active LOW
0	1	Registered/Active HIGH
1	0	Combinatorial/Active LOW
1	1	Combinatorial/Active HIGH

ispGAL : In-System Programmable GAL

- 'normale' GAL: programmeren (en evtl. herprogrammeren) gebeurt met programmeertoestel
 - bijkomende productiestap
 - problemen bij componenten met 'high pin count' en 'small lead pitch'
- ispGAL: kan op de PCB geprogrammeerd worden
 - vereist 4 bijkomende pinnen + extra interne logica
 - kan nu op verschillende manieren geprogrammeerd worden:
 - » met hardware programmer, voorafgaand aan de PCB assemblage
 - » met 'Automatic Test Equipment' (ATE) na de assemblage
 - » vanuit een computer na assemblage via 'download connection'
 - » vanuit een in-system microprocessor na assemblage

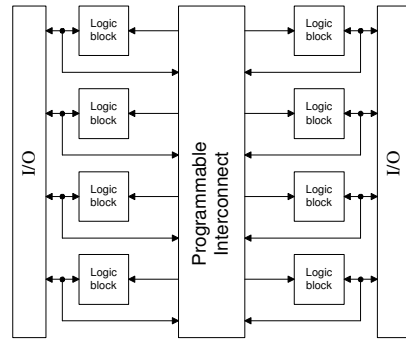
CPLD = Complex PLD

- Bestaat uit verschillende logische blokken die elk equivalent zijn met een (eenvoudige) GAL zoals de 22V10

- Deze logische blokken worden met elkaar verbonden via een programmeerbare interconnect matrix structuur

- Geïntroduceerd door Altera onder de benaming EPLD = Erasable PLD

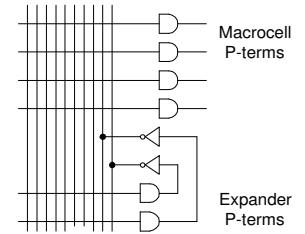
- gebaseerd op UV-EPROM technologie



CPLD : nieuwe kenmerken

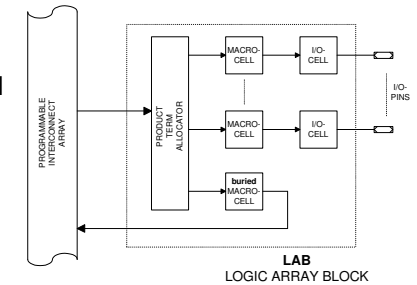
- Product-term sharing

- het delen (gemeenschappelijk stellen) van producttermen door verschillende OR-poorten (i.p.v. telkens opnieuw te moeten maken)
- door deze 'Expander Product Terms' kan een functie veel meer product-termen bevatten



- Buried macrocell

- vergelijkbaar met gewone macrocell
- uitgang komt echter niet terecht op een uitgangspen, maar wordt teruggevoerd naar de AND-array
- 'buried' = begraven



EPLD = Erasable PLD

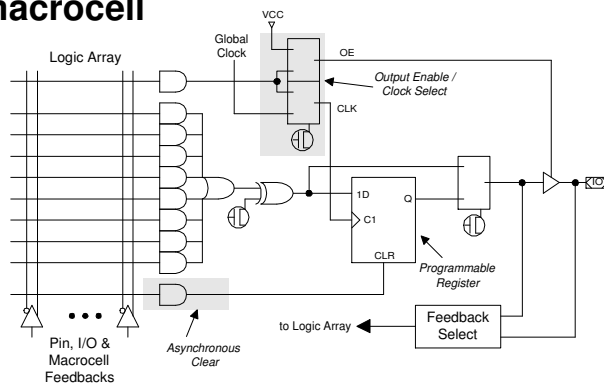
- Geïntroduceerd door Altera in 1983

- gebaseerd op EPROM-technologie

- » dus: UV-wisbaar

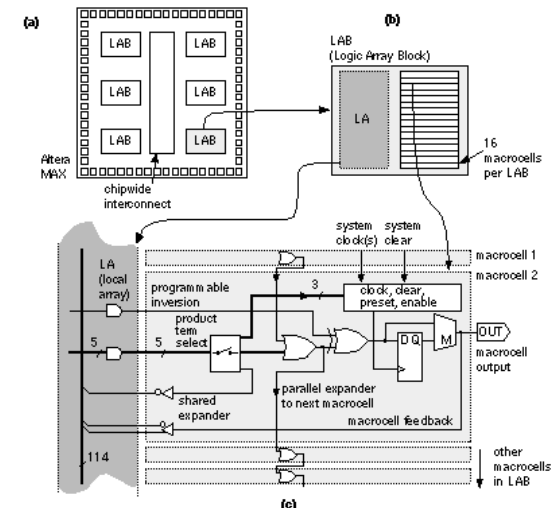
- » programmeren met programmeertoestel

- Basis-macrocell



Altera EPLD

- MAX5000 reeks



FPGA = Field Programmable Gate Array

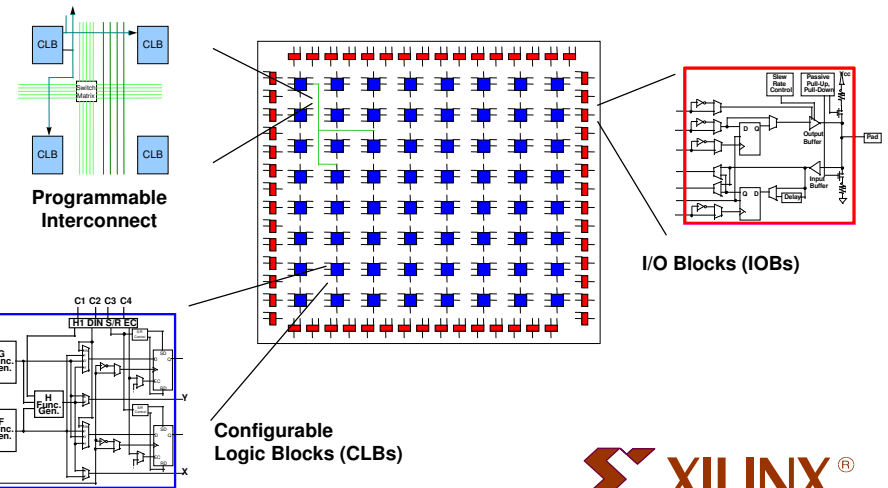
• Geïntroduceerd door Xilinx in 1985

- gebaseerd op SRAM-technologie
 - » dus: spanning weg, informatie weg (*volatile*)!
- in-system (re)programmable
 - » niet met programmeertoestel
 - » wel vanuit PC of vanuit een vast geheugen (bv. seriële EPROM)
 - » reprogrammable 'in-the-flight'

• Structuur

- matrix van vrij configureerbare, complexe logische blokken
 - » CLB = Configurable Logic Block
- omgeven door een ring van input/output blokken
 - » IOB = I/O Interface Block
- programmeerbare verbindingen tussen CLB's onderling en tussen CLB en IOB

XILINX Basic FPGA Architecture

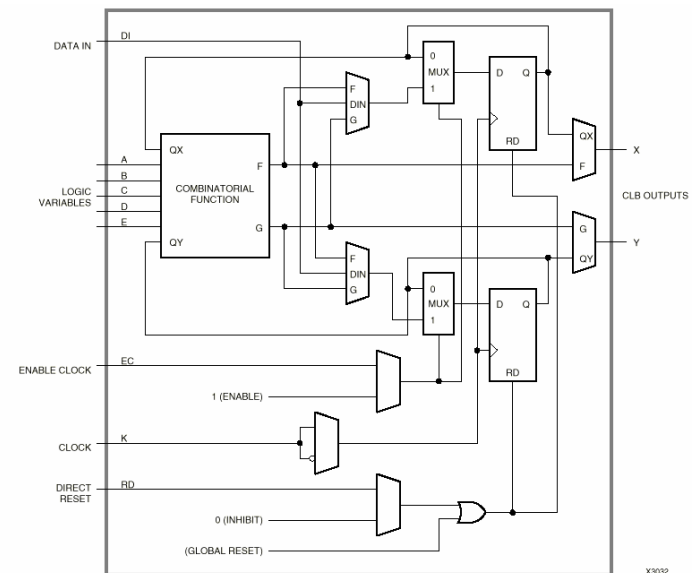


XILINX FPGA families

	CLB's	matrix	FF's	I/O
XC2000				
XC2064	64	8x8	64	58
XC2018	100	10x10	100	74
XC3000				
XC3020	64	8x8	64	64
XC3030	100	10x10	100	80
XC3042	144	12x12	144	96
XC3064	224	16x14	224	120
XC3090	320	16x20	320	144
XC3195	484	22x22	484	176
XC4000				
XC4003	100	10x10	360	80
XC4005	196	14x14	616	112
XC4006	256	16x16	768	128
XC4008	324	18x18	936	144
XC4010	400	20x20	1120	160
XC4013	576	24x24	1536	192
XC4020	784	28x28	2016	224
XC4025	1024	32x32	2560	256

Discontinued!

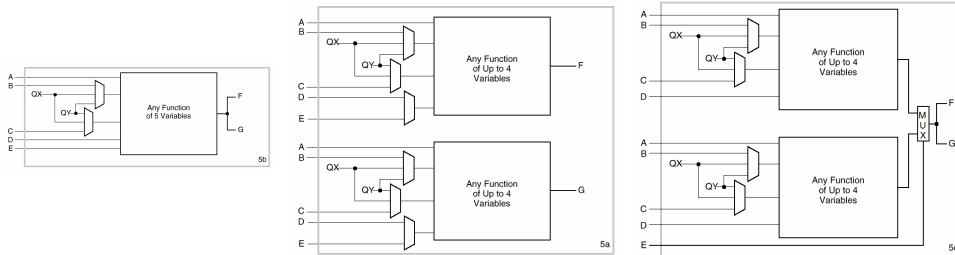
XILINX XC3000 FPGA : CLB



XILINX XC3000 FPGA : CLB

• Combinatorisch gedeelte in CLB:

- LUT (*look up table*) met 5 logische ingangen
- hiermee kunnen volgende functies gerealiseerd worden:
 - » ofwel één willekeurige functie van 5 variabelen
 - » ofwel twee onafhankelijke functies van elk 4 variabelen
 - » ofwel een dynamische selectie tussen 2 functies van 4 variabelen



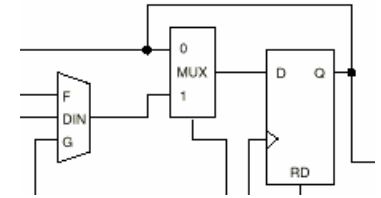
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : CLB

• Sequentieel gedeelte in CLB:

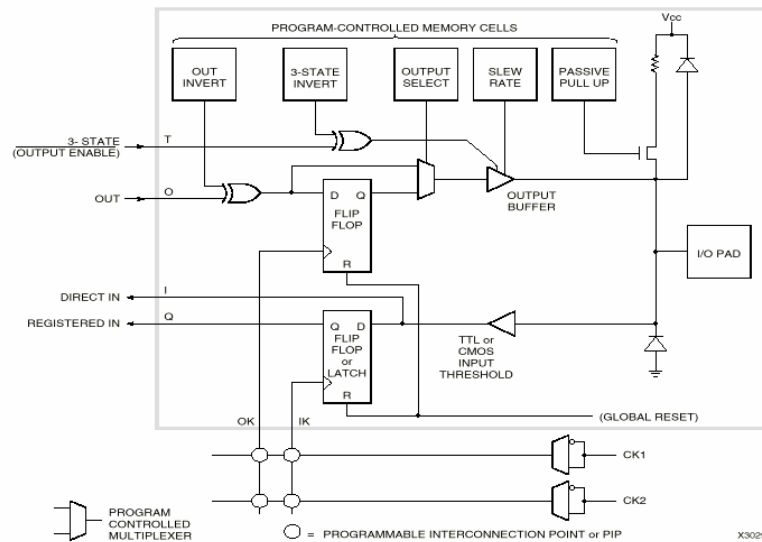
- twee D-flipflops
- D-ingang komt van
 - » combinatorisch blok (LUT)
 - » afzonderlijke D-ingang
- CLOCK-ingang
 - » gemeenschappelijk voor beide D-flipflops
 - » al dan niet geïnverteerd (dus: actief op stijgende of dalende flank)
- CLOCK ENABLE
 - » gemeenschappelijk voor beide D-flipflops
- asynchrone CLEAR
 - » gemeenschappelijk voor beide D-flipflops
 - » CLEAR gebeurt door:
 - individuele ingang
 - global reset (gestuurd door externe RESET-pin en bij POWER ON)



Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : IOB



Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : IOB

• externe pin wordt gebruikt als ingang of als uitgang

- zowel bij ingang als bij uitgang kan men kiezen voor een *direct path* (zonder flipflop) of een *registered path* (met flipflop)
- flipflops hebben twee mogelijke clock-signalen (+ inverteerbaar)
- ingangsgedeelte
 - *threshold* instelbaar: TTL- of CMOS-compatibel
 - *direct* of *registered* (via latch of via flipflop)
- uitgangsgedeelte: extra mogelijkheden
 - data wordt al dan niet geïnverteerd
 - data passeert al dan niet via D-flipflop
 - 3-state output buffer (met programmeerbare *enable*)
 - instelbare *slew-rate*
 - » snel, maar meer vermogen
 - » traag met minder vermogen
 - al dan niet passieve *pull-up*

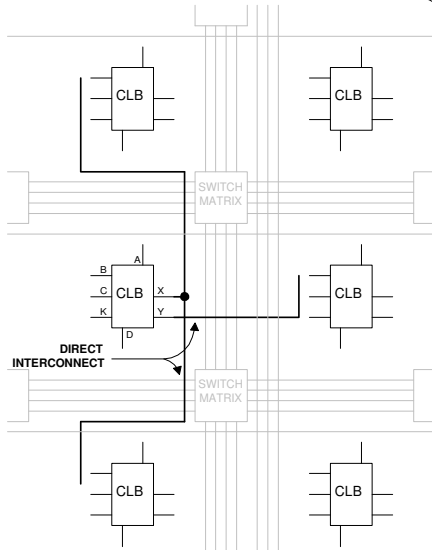
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : INTERCONNECT

• Interconnect op 3 niveaus:

- direct interconnect of local interconnection
 - » korte, rechtstreekse verbindingen tussen aangrenzende blokken
 - » beperkt aantal mogelijkheden
 - » snelste verbinding



Programmeerbare Logica

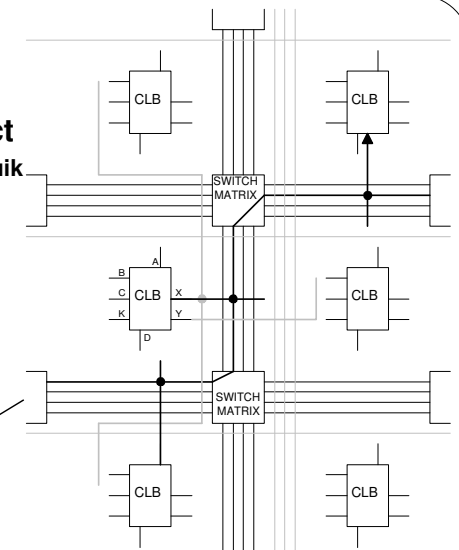
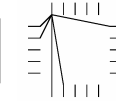
DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : INTERCONNECT

• Interconnect op 3 niveaus:

- direct (local) interconnect
- general purpose interconnect
 - » universele verbindingen die gebruik maken van de schakelmatrices (switch matrix) tussen de blokken
 - » haast alles is mogelijk
 - » vrij grote vertraging omdat het signaal via veel schakelaars en tussenbuffers moet passeren

Voorbeeld van mogelijke verbindingen in de switch matrix



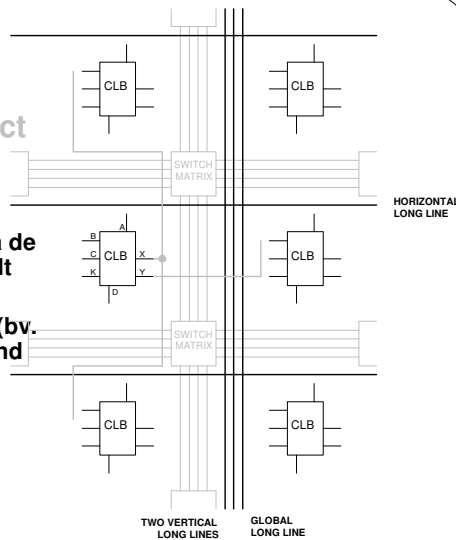
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC3000 FPGA : INTERCONNECT

• Interconnect op 3 niveaus:

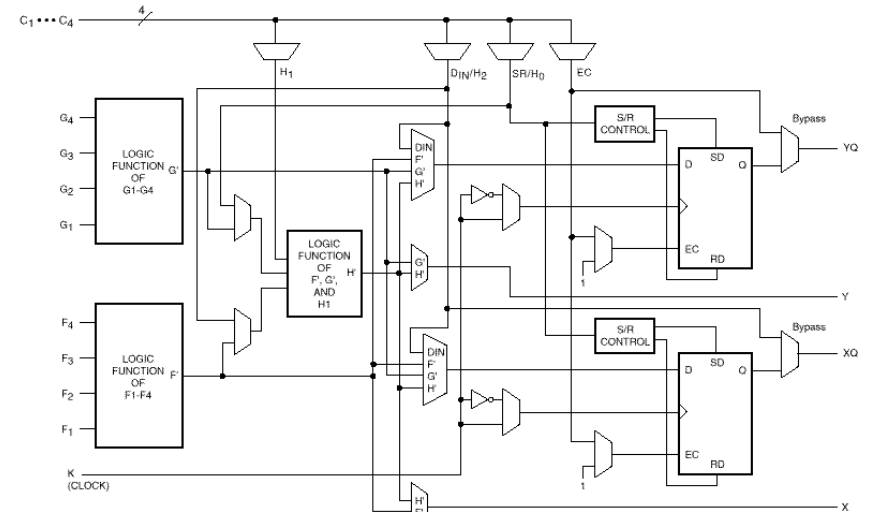
- direct (local) interconnect
- general purpose interconnect
- longlines of global interconnect
 - » kleinere vertraging omdat niet via de switch matrices gepasseerd wordt
 - » voorbehouden voor signalen die minimale skew moeten vertonen (bv. clock) of die over een grote afstand moeten getransporteerd worden



Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC4000 FPGA : CLB



Multiplexer Controlled by Configuration Program

X6692

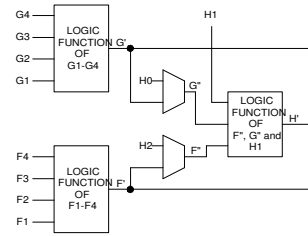
Programmeerbare Logica

DIGITALE ELEKTRONICA 2EL

XILINX XC4000 FPGA : CLB

• Combinatorisch gedeelte in CLB:

- twee LUTs (*look up tables*) met elk 4 logische ingangen
 - » genereren de functies $F' = f(F_1, F_2, F_3, F_4)$ en $G' = f(G_1, G_2, G_3, G_4)$
- daarnaast een derde LUT
 - » genereert een functie H' van drie variabelen:
 - F' of de aparte ingang H_2
 - G' of de aparte ingang H_0
 - de aparte ingang H_1



– Hiermee kunnen volgende functies gerealiseerd:

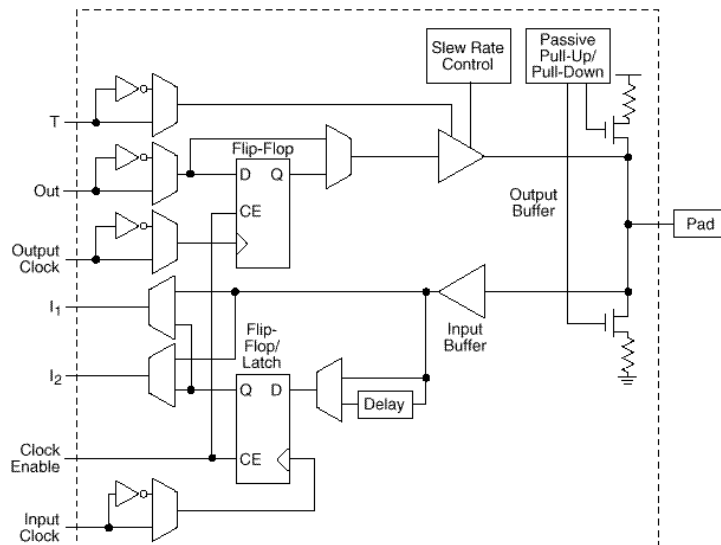
- » twee functies van 4 onafhankelijke variabelen plus een derde functie van nog eens maximaal drie onafhankelijke variabelen
- » een willekeurige functie van 5 variabelen
- » elke functie van 4 variabelen + sommige functies van 6 variabelen
- » sommige functies van max. 9 variabelen

XILINX XC4000 FPGA : CLB

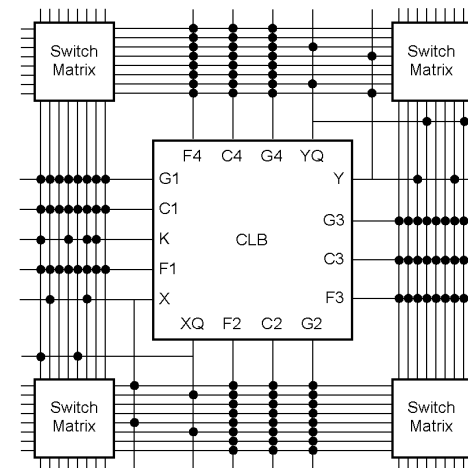
• Bijkomende functionaliteit van CLB:

- LUT (*look up table*) kan ook gebruikt worden als snelle RAM
 - » verschillende modes:
 - level-sensitive
 - edge-triggered (= synchronous)
 - dual port edge-triggered
 - » werkt vlugger dan externe RAM
- Fast Carry Logic
 - » elke LUT voor 4 ingangen in voorzien om gebruikt te worden als sommatoren voor twee woorden van elk 2 bits
 - snelle generatie van CARRY of BORROW
 - wordt rechtstreeks doorgegeven naar nabijgelegen CLB
 - passeert niet via switch matrix, dus zeer snel
 - » betere performance voor
 - adders, subtractors, accumulators, comparators, counters

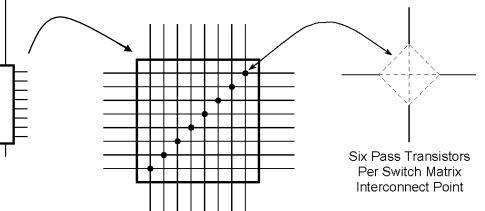
XILINX XC4000 FPGA : IOB



XILINX XC4000 FPGA : INTERCONNECT



General Purpose Interconnect via switch matrix



CONFIGUREREN VAN DE FPGA

- **FPGA configureren = opladen van de SRAM-cellen met een 'bitpatroon', zodat de gewenste schakeling gerealiseerd wordt**
- **moet steeds gebeuren met de FPGA *in circuit***
- **kan op verschillende manieren:**
 - vanuit de PC met een speciale *download cable*
 - vanuit een externe seriële PROM
 - » vereist slechts één kleine chip en weinig verbindingen
 - » seriële PROMs bestaan van 18K tot 256K
 - » Interne CCLK klok spreekt een teller aan in de seriële PROM
 - vanuit een externe gewone (parallele) PROM
 - » Interne CCLK klok spreekt een teller in de FPGA aan
 - » Data wordt intern toch nog steeds serieel verwerkt, zodat de configuratie op dezelfde snelheid gebeurt
 - vanuit en onder de controle van een externe microprocessor of -controller

ONTWERP-SOFTWARE

- **Programmeerbare bouwstenen hebben een steeds toenemende complexiteit**
 - SPLD = Simple PLD (PAL en GAL)
 - CPLD = Complex PLD (EPLD)
 - FPGA = Field Programmable Gate Array
- en bevatten duizenden te programmeren punten**
- **Hulp van de computer is onmisbaar om een bepaalde functie onder te brengen in een PLD**
 - hulp bij het ingeven en simuleren van het ontwerp
 - hulp bij het 'vertalen' van dit ontwerp naar een PLD-schakeling
 - hulp bij het configureren van de PLD
- **EDA = Electronic Design Automation**
CAEE = Computer Aided Electronic Engineering

ONTWERP-SOFTWARE voor SPLD : PALASM

- **SPLD = Simple PLD (PAL + GAL)**
 - eerst op de markt gebracht door de firma MMI (Monolithic Memories Inc), die later werd opgekocht door AMD
- **PALASM (=PAL-assembler)**
 - door AMD/MMI (uitvinders v/d PAL) geleverde ontwikkelingsomgeving
 - staat heel kort bij de component (databoek is nodig!)
 - is een programmeertaal met een vrij eenvoudige syntax
 - schakeling kan op verschillende manieren worden ingegeven:
 - » logische vergelijkingen
 - » waarheidstabellen
 - » toestandmachines (FSM = Finite State Machine)
 - zorgt zelf voor minimalisatie
 - biedt de mogelijkheid tot simulatie
 - genereert de JEDEC-file (= fuse-map) die het programmeertoestel nodig heeft om de PLD te programmeren

ONTWERP-SOFTWARE voor SPLD : ABEL

- **ABEL = Advanced Boolean Expression Language**
 - op de markt gebracht door DATA-I/O, een fabrikant van programmeer-toestellen (en dus niet van PLD's zelf)
 - staat op een hoger niveau (*compiler*) dan PALASM (*assembler*)
 - » heeft complexere syntax-constructies
 - kent bv. de instructie `COUNT := COUNT + 1` om een teller te beschrijven
 - bij PALASM moet je zelf de vergelijkingen voor de verschillende flipflops afleiden
 - » kan meer onafhankelijk van de component gebruikt worden
 - moet uiteraard hetzelfde doel vervullen
 - » *design entry* (met *equations*, *truth tables* en/of *state diagrams*)
 - » *design simulation*
 - » *logic synthesis* (= vertalen naar JEDEC-file)
 - ook nu nog vaak onderdeel van grotere ontwerp-omgeving
- **Andere PLD-compiler : CUPL (van Logical Devices)**

ONTWERP-SOFTWARE voor CPLD en FPGA

- **Complexere componenten vragen ook een ontwerp-omgeving met een hogere complexiteit**
 - we gaan een FPGA met 10000 gates niet beschrijven met alleen maar Booleaanse vergelijkingen
- **Ontwikkelomgeving =**
 - FRONT-END**
 - staat het kortst bij de gebruiker
 - bevat DESIGN ENTRY en SIMULATION
 - vaak van een onafhankelijke software-fabrikant
 - BACK-END**
 - staat het dichtst bij de component
 - bevat DESIGN IMPLEMENTATION
 - altijd van de fabrikant van de gekozen PLD

DESIGN ENTRY

Twee mogelijkheden voor DESIGN ENTRY :

- **Schematic Capture**
 - tekenpakket om op grafische wijze het schema van de schakeling in te voeren
 - wordt onoverzichtelijk voor grote ontwerpen
- **HDL = Hardware Description Language**
 - de schakeling wordt 'beschreven' met een soort programmeertaal
 - » beschrijving van *de structuur* en/of *het gedrag* van de schakeling
 - » vergelijk met ABEL, maar op een hoger niveau
 - » oorspronkelijk bedoeld voor specificatie en documentatie
 - » geschikt voor simulatie en voor automatische synthese
 - in Europa : VHDL = VHSIC Hardware Description Language
(VHSIC = Very High Speed Integrated Circuit)
 - in Amerika : VERILOG

SIMULATIE

Twee mogelijkheden voor SIMULATIE :

- **functionele simulatie**
 - gebeurt onmiddellijk na het ingeven van het schema (of na synthese van de HDL-code)
 - controleert de logische werking van de schakeling
 - houdt geen rekening met vertragingen of hanteert 'unit-delay'
- **timing simulatie**
 - kan pas gebeuren nadat de schakeling volledig is geïmplementeerd in de gekozen component
 - op dat ogenblik kent de software de juiste vertragingen
 - er gebeurt nu een 'back-annotation', waarna de (zelfde) functionele simulatie opnieuw kan gebeuren, maar nu wel rekening houdende met de exacte te verwachten vertragingen