# DISSENY I IMPLEMENTACIÓ DELS PROCESSOS EMOCIONALS D'UN AGENT SOFTWARE SOBRE HARDWARE RECONFIGURABLE

## ÍNDEX

# INTRODUCCIÓ

## DESCRIPCIÓ

Aquest projecte tracta de dissenyar els processos o funcions per a que un agent emocional puga prendre les decisions necessàries per a dur a terme un objectiu. Amés es realitzarà una implementació sobre una FPGA.

## CONTEXT

Este projecte s'emmarca dins un projecte més gran dut a terme en el DISCA, en el qual s'està desenvolupant un robot emocional, és a dir, una màquina que reaccionarà de manera diferent depenent dels estímuls externs que li arriben.

Dins aquest projecte es vol avaluar i comparar la diferència de recursos emprats, tant temporals com espacials, per tal de dur a terme els càlculs necessaris. Aquestos càlculs es realitzaran mitjançant una FPGA i un processador multi-nucli per a que l'agent emocional decidisca la millor opció entre les disponibles.

## OBJECTIUS DEL PROJECTE

**Descripció general**: es pretén realitzar la implementació sobre una FPGA dels processos emocionals de l'agent.

**Descripció detallada**: les variables a avaluar una vegada obtinguda la implementació són els temps de còmput i els nombre de portes o recursos emprats. S'avaluarà els graus de llibertat disponibles i les diferents implementacions possibles depenent d'aquests: taules, funcions iteratives, funcions recursives, etc.. Es realitzarà un disseny genèric per tal de possibilitar una implementació independent del hardware i permetre diferents precisions d'entrades i eixides. Es programarà sobre un placa SmartFusion de Actel i s'avaluarà la possibilitat de programar-ho en altres tecnologies disponibles al mercat (Altera, Xillinx, etc).

# ESTRATÈGIA DE TREBALL

Analitzar les diferents funcions de càlcul.

Aprendre les diferents ferramentes de treball.

Implementar en VHDL el disseny escollit.

Programar la placa SmartFusion de Actel.

Veure altres opcions d'implementació sobre altres tecnologies disponibles al mercat i baremar les diferents opcions.

# EINES UTILITZADES

Placa SmartFusion de Actel.

Libero IDE v9.0

ModelSim (versió per a Actel, Xilinx i Altera)

Synplify

FlashPro

Identify Debugger

Llenguatge de descripció de hardwre VHDL-200X, compatible amb VHDL-93.

Xilinx ISE Webpack 12.1

XST synthesyser

Altera Quartus II 10.0

# Introducció teòrica als conceptes relacionats

## FPGAs

Abans de començar amb les nostres FPGA, revisarem els conceptes bàsics de les FPGA.

Una FPGA (Field Programmable Gate Array) és un dispositiu construït principalment amb material semiconductor que conté blocs de lògica que permeten ser programats. La lògica programable pot reproduir des de funcions tan senzilles com les que realitza una porta lògica fins a sistemes complexos en un xip o SoC (System on a Chip).

Les FPGAs s'utilitzen en aplicacions similars als ASICs, construïdes per a finalitats més específiques, encara que són més lentes, tenen un major consum de potència i no poden comprendre sistemes tan complexos. Encara així, les FPGAs tenen els avantatges de ser reprogramables (el que afegeix una gran flexibilitat al flux de disseny), els seus costos de desenvolupament i adquisició són molt menors per a petites quantitats de dispositius i el temps de desenvolupament és també menor.

Certs fabricants compten amb FPGAs que només es poden programar un cop, pel que els seus avantatges i inconvenients es troben a mig camí entre els ASICs i les FPGAs reprogramables.

Històricament les FPGAs sorgeixen com una evolució dels conceptes desenvolupats en les PLAs i els CPLDs.

Tradicionalment, els enginyers han utilitzat les FPGA amb eines de programació fetes per experts. No obstant, com que les FPGA s'han tornat més ràpides i més rentables, els enginyers i investigadors amb poca o cap experiència en disseny de hardware digital estan buscant aprofitar les FPGA per crear solucions personalitzades. Per abarcar aquest creixent interés, els proveïdors estan creant eines de més alt nivell que fan més fàcil programar FPGAs i brindar els beneficis de la tecnologia FPGA a noves aplicacions.

### FPGAs respecte als ASICs

Les FPGAs s'utilitzen en aplicacions similars als ASICs però tenen els següents inconvenients i avantatges respecte a ells.

**Inconvenients:**

- Son més lents.
- Consumeixen més potència.
- No poden realitzar sistemes tan complexos.

**Avantatges:**

- Son re-programables.
- Costos de desenvolupament i adquisició molt més baixos.
- Temps de desenvolupament menor.

## Consideracions de seguretat

Pel que fa a la seguretat, les FPGAs tenen avantatges i desavantatges en comparació amb els ASICs o microprocessadors segurs. La flexibilitat FPGAs , fa que les modificacions malicioses que pugi haver durant la fabricació, siguin de menor risc. Per molts FPGAs, el disseny carregat està exposat mentre es carrega (en general en cada encesa del dispositiu). Per abordar aquesta qüestió, alguns FPGAs suporten el xifrat 'bitstream encryption'.

## Programació i fluxe de disseny

La feina del programador es definir la funció lògica que realitzarà cada un de les CLB (cel·la del bloc lògic), seleccionar el mode de treball de cada IOB (bloc d'entrada/eixida) e interconnectar-los.

El fluxe de disseny d'un sistema podria ser:

- **Divisió del disseny principal en mòduls** separats. La modularitat es un dels conceptes principals de tot disseny. Normalment es diferència entre dos metodologies de disseny: top-down i botton-up. La metodologia top-down consisteix en que un disseny complex es divideix en dissenys més simples que es puguen dissenyar (o descriure) més fàcilment. La metodologia botton-up consisteix en construir un disseny complexe a partir de mòduls, ja dissenyats, més simples. En la pràctica, un disseny usa generalment ambdues metodologies.

- Per a **l'entrada de dissenys**, poden utilitzar-se diversos mètodes tal com VHDL, Verilog o ABEL.

- **Simulació funcional**, és a dir, comprobarem que allò escrit en el punt anterior realment funciona com volem, si no ho fa tindrem que modificar-lo. En aquest tipup de simulació es comproba que el codi VHDL o Verilog (u otro tipo de lenguaje HDL) executa correctament el que es pretén.

- **Síntesis**. En aquest pas s'adapta el diseny anterior (que sabem que funciona) a un hardware en concreto, ja siga una FPGA o un ASIC. Hi ha sentències del llenguatge que no són sintetitzables, com per exemple divisions o exponenciacions amb nombres no constants. El fet de que no totes les expressions en VHDL siguen sintetitzables és degut a que el VHDL és un llenguatge genèric per a modelat de sistemes (no sols per a disseny de circuits digitals), per això hi ha expressions que no poden ser transformades a circuits digitals. Durant la síntesis es té en compte l'estructura interna del dispositiu, i es defineixen restriccions, com l'assignació de pins. El sintetitzador optimitza las expressions lògiques amb la finalitat que ocupen menor àrea, o bé són eliminades les expressions lògiques que no són utilitzades pel circuit.

- **Simulació post-síntesis**. En aquest tipus de simulació es comprova que el sintetitzador ha realitzat correctament la síntesis del circuit, al transformar el codi HDL en blocs lògics connectats entre sí. Aquest pas és necessari ja que, a vegades, els sintetitzadors produeixen resultats de síntesis incorrectes, o bé realitzen simplificacions del circuit al optimitzar-lo.

- **Placement i routing**. El procés de placement consisteix en situar els blocs digitals obtinguts en la síntesis de forma òptima, de forma que aquells blocs que es troben molt interconnectats entre sí se situen pròximament. El procés de routing consisteix en enrutar adequadament els blocs entre sí, intentant minimitzar retards de propagació para maximitzar la freqüència màxima de funcionament del dispositiu.

- **Back-annotation**. Una vegada ha sigut completat el placement & routing, s'extrauen els retards dels blocs i les sues interconnexions, amb la finalitat de poder realitzar una simulació temporal (també anomenada simulació post-layout). Estos retards són anotats en un fitxer SDF (Standart Delay Format) que associa a cada bloc o interconnexió un retard mínim/típic/màxim.

- **Simulación temporal**. A pesar de la simulació anterior potser el disseny no funcione quan es programa. Una de les causes pot ser degut als retards interns

del xip. Amb esta simulació es pot comprovar, i si hi ha errors s'ha de tornar a un dels anteriors passos.

- **Programació en el dispositiu**. S'implementa el disseny en el dispositiu final y es comprova el resultat. La programació consisteix en carregar una memòria RAM estàtica, una EEPROM o establint els antifusibles, que són com xicotets interruptors programables per tensió.

## Aplicacions

Qualsevol circuit d'aplicació específica pot ser implementat en una FPGA, sempre i quan aquest disposi dels recursos necessaris. Les aplicacions on més comunament s'utilitzen les FPGA inclusiu els DSP (processament digital de senyals) són sistemes aeroespacials i de defensa, prototipus de ASICs, sistemes d'imatges per a medicina, sistemes de visió per a computadors, reconeixement de veu, bioinformàtica, emulació de hardware de computadora, etc. Hem de saber que el seu ús en altres àrees, es cada vegada major, sobretot en aquelles aplicacions que requereixen un alt grau de paral·lelisme.

Existeix codi font disponible (sota llicencia GNU GPL) de sistemes com microprocessadores, microcontroladores, filtres, móduls de comunicacions i memòries, entre altres. Aquests codis s'anomenen cores.

## Tecnologia de la memòria de programació

Les FPGAs també es poden diferenciar per utilitzar diferents tecnologies de memòria:

Volàtils: Basades en RAM. La seva programació es perd al treure l' alimentació. Requereixen una memòria externa no volàtil per a configurar-les al arrancar (abans o durant el reset).

No Volàtils: Basades en ROM. Hi ha dos tipus, les reprogramables, asades en EPROM o flash que es poden borrar i tornar a reprogramar, encara que amb un limit de uns 10.000 cicles, i les no reprogramables, basades en fusibles que només es poden programar una vegada, cosa que les fa poc recomanables per a treballs en laboratoris.

### Fabricants de FPGA

A principis de 2007, el mercat de les FPGA s'havia col·locat en un estat on hi havia dos grans productors de FPGA de propòsit general i un conjunt d'altres competidors els quals es diferencien per oferir dispositius de capacitats úniques. Els dos grans són Xillinx i Altera. Altres fabricants són Lattice, Actel, QuickLogic, Atmel, Achronix Semiconductor, MathStar, Inc.

# IEEE-754. Representació de nombres en coma flotant

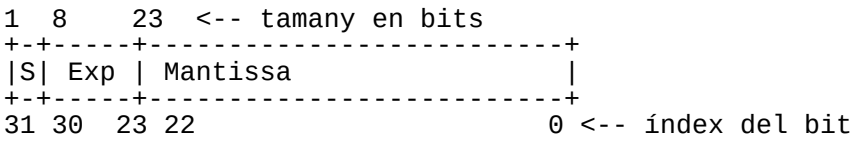
L'estàndard de la IEEE para aritmètica en coma flotant (IEEE 754) és l'estàndard més estès per a la computació en coma flotant, i és seguit per moltes de les millores de CPU i FPU (unitat de coma flotant). El estàndard defineix formats per a la representació de números en coma flotante (incloent el cero) y valores desnormalitzats, així com valors especials com infinit i NaN (*not a number*), amb un conjunt d'operacions en coma flotant que treballa sobre aquestos valors. També especifica quatre modes d'arrodoniment y cinc excepcions (incloent quan ocorren dites excepcions y què passa en l'execució).

IEEE 754 especifica quatre formats per a la representació de valores en coma flotant: precisió simple (32 bits), precisió doble (64 bits), precisió simple estesa (≥ 43 bits, no usada normalment) y precisió doble estesa(≥ 79 bits, usualment implementada amb 80 bits). Sols els valors de 32 bits són requerits per l'estàndard, els altres són opcionals. Molts llenguatges especifiquen quins formats y aritmètica de la IEEE implementen, a pesar de que a vegades són opcionals. Per exemple, el llenguatge de programació C, ara permet però no requerix la aritmètica de la IEEE (el tipus de C float és típicament usat per a la precisió simple de la IEEE i el tipus double usa la precisió doble del la IEEE).

El títol complet del estàndard és *IEEE Standard for Binary Floating-Point Arithmetic* (ANSI/IEEE Std 754-1985), i també és conegut per IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems.*

### Precisión simple 32-bits

Un numero en coma flotant de precisió simple s'emmagatzema en una paraula de 32 bits.




On S es el bit de signe i Exp és el camp exponent. (Per al signe: 0=Positiu ; 1= Negatiu).

L'exponent és desplaçat en en número en precisió simple, un exponent en el rang −126 a +127 és desplaçat mitjançant la suma de 127 per a obtindre un valor en el rang 1 a 254 (0 i 255 tenen valors especials descrits més endavant). Quan s'interpreta el valor en coma flotant, el número és desplaçat de nou però a l'inversa per tal d'obtindre l'exponent real.

El conjunt de valors possibles poden ser dividits en els següents:

- zeros
- números normalitzats
- números desnormalitzats
- Infinits
- NaN

Els casos següents poden generar l'estat de NaN en la majoria dels llenguatges de programació que accepten aquest estat com retorn d'una funció matemàtica:

- Totes les operacions matemàtiques que tinguin NaN com operant matemàtic.
- Les divisions indeterminades o per infinit (0 / 0, ∞ / ∞, ∞ / - ∞, - ∞ / ∞, ...)
- Les multiplicacions de 0 per infinit (0 × ∞, 0 × - ∞)
- Les sumes i restes de valors infinits (∞ + (- ∞), (- ∞) + ∞)
- Aplicant funcions que excedeixin el domini de la mateixa. Per exemple, l'arrel quadrada d'un nombre negatiu, logaritme de qualsevol nombre menor o igual que 0, o la inversa d'un cosinus que sigui menor que -1 o major que +1.

## Classes de nombres representats en IEEE-754

Las classes es distingeixen principalment per el valor del camp Exp, sent modificada aquesta pel camp fracció. Considera Exp i Fracció com camps de números binaris sense signe (Exp es troba en el rang 0–255):

| Classe | Exp | Fracció |
|---|---|---|
| Zeros | 0 | 0 |
| Números desnormalitzats | 0 | distint de 0 |
| Números normalitzats | 1-254 | Qualsevol |
| Infinits | 255 | 0 |
| NaN (Not a Number) | 255 | distint de 0 |

Per a números normalitzats, els més comuns, Exp és l'exponent desplaçat i Fracció és la part fraccional de la mantissa (o significant). El número té valor v:

$$v = s \times 2\text{^}e \times m$$

On,

s = +1 (numeros positius) quan S és 0

s = −1 (números negatius) quan S és 1

e = Exp − 127 (en altres paraules, a l'exponent se li suma 127 i s'emmagatzema, a aquest també se li anomena "biased with 127" en anglès)

m = 1,Fracció en binari (açò és, el significant és el número binari 1 seguit per la coma decimal, seguit pels bits de Fracció). Per tant, 1 ≤ m < 2.

Notes:

1. Els números desnormalitzats són iguals excepte que e = −126 y m = 0,Fraccio. (e NO és -127 : el significant ha de ser desplaçat a la dreta por un bit més, de forma que incloga el bit principal, que no sempre és 1 en aquest cas. Açò es balanceja incrementant l'exponent a -126 per al càlcul).
2. −126 és el menor exponent per a un número desnormalitzat
3. Hi ha dos zeros. +0 (S és 0) y −0 (S és 1)
4. Hi ha dos infinits +∞ (S és 0) y −∞ (S és 1)

5. Els NaNs poden tindre un signe i un significant, però aquestos no tenen altre significat que el que poden aportar en proves de diagnòstic; el primer bit del significant és moltes vegades utilitzat per a distingir NaNs senyalitzats de NaNs silenciosos
6. Els NaNs y els infinits tenen tots els bits a 1 en el camp Exp.

# VHDL

VHDL és l'acrònim que representa la combinació de VHSIC i HDL, on VHSIC és l'acrònim de "Very High Speed Integrated Circuit" i HDL és a la seva vegada l'acrònim de "Hardware Description Language".

És un llenguatge fet servir per enginyers definit pel IEEE (Institute of Electrical and Electronics Engineers) en ANSI/IEEE 1076-1993 que es fa servir per especificar,dissenyar i simular circuits digitals.

Altres mètodes per dissenyar circuits són la captura d'esquemes (amb eines tipus CAD) i els diagrames de blocs, però aquest no son pràctics en dissenys complexes. Altres llenguatges pel mateix propòsit poden ser Verilog i ABEL.

Tot i que es pot fer servir de forma general per descriure qualsevol circuit es fa servir principalment per programar PLD (Programable Logic Device - Dispositiu Logic Programable) i FPGAs.

# La funció tangent hiperbòlica

S'anomenen funcions hiperbòliques al cosinus hiperbòlic (denotat cosh), sinus hiperbòlic (sinh) i les funcions que s'obtenen a partir d'elles, com la tangent hiperbòlica (tanh), la cotangent hiperbòlica (coth), la secant hiperbòlica (sech ) i la cosecant hiperbòlica (cosech).

De la mateixa manera que les funcions trigonomètriques permeten localitzar sobre el cercle trigonomètric, les funcions hiperbòliques donen la posició d'un punt qualsevol de la branca positiva de la hipèrbola d'equació:

$$x^2 - y^2 = 1$$

(En un sistema de coordenades ortonormal).

Un punt A (cosh a, sinh a) pertany a aquesta hipèrbola perquè les seves coordenades verifiquen la seva equació, concretament:

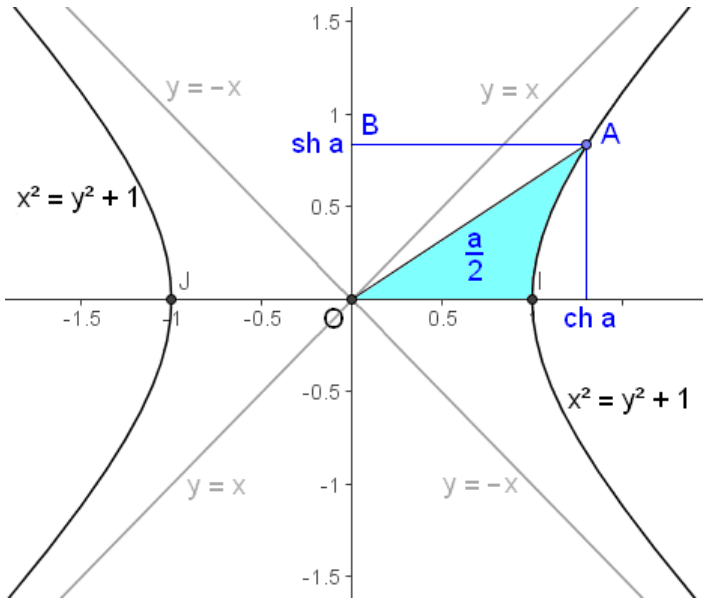
$$\cosh^2 x = \sinh^2 x + 1$$

Això equival a dir que el sistema:

$$\begin{cases} x = \cosh x \\ y = \sinh x \end{cases}$$

és una representació paramètrica (o equació paramètrica) d'aquesta branca de hipèrbola.

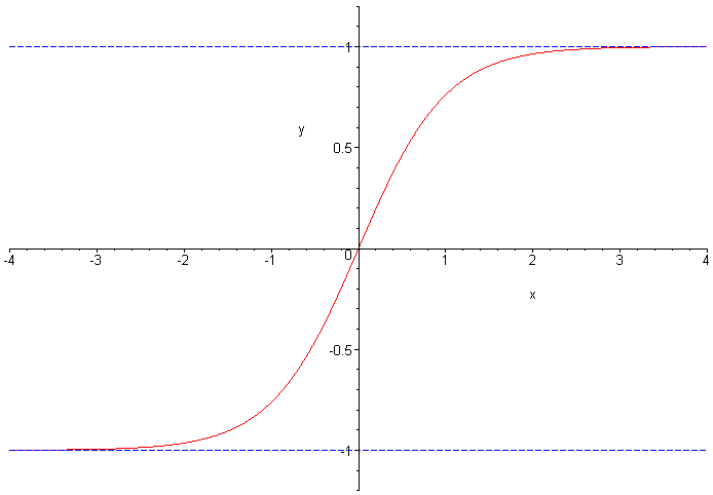
Però el més sorprenent és que el paràmetre "a" te una interpretació geomètrica senzilla: és el doble de l'àrea delimitada per l'eix d'abscisses, la recta (OA) i la hipèrbola (superfície dibuixada en blau). La semblança amb la trigonometria circular és cridanera i deixa entreveure que existeix un vincle molt profund entre les dues geometries, la circular (euclidiana) i la hiperbòlica.




Definició geomètrica de funció hiperbòlica sh=sinh i ch=cosh

La tangent hiperbòlica d'un nombre real x es designa mitjançant el quocient entre el sinus hiperbòlic i el cosinus hiperbòlic.

$$\tanh x = \frac{\sinh x}{\cosh x}$$




Representació gràfica de tanh(x)

# COS DEL PROJECTE

## Anàlisi de la funció tangent hiperbòlica

La funció tangent hiperbòlica es la funció que s'ha escollit per a provocar la resposta del robot emotiu front a les entrades que li son proporcionades, és a dir per a dissenyar el seu comportament. S'ha triat aquesta funció perquè es una funció continua en el domini [-1, 1], que seran els nostres valors límits d'entrada. Amés és una funció sense canvis bruscos, que evoluciona de forma suau.

Com se li proporciona l'entrada des del robot? Ja que no és l'objecte d'estudi d'aquest projecte, direm que, simplificant molt, el robot pren una sèrie de mesures que es poden quantificar, com podria ser la distancia des d'aquest a diferents objecte, la posició dels diferents objectes, el seu volum, el seu pes, etc. Amb tota aquesta informació, es calcula, donant a cadascuna de les diferents mesures un pes corresponent i es normalitza el resultat entre [-1, 1]. Una vegada extret aquest valor, es passarà a l'entrada de l'FPGA per tal de calcular la tanh d'aquest valor normalitzat, i així determinar quina serà la resposta del robot.

Per a calcular el valor de la tangent hiperbòlica anem a utilitzar el seu desenvolupament en sèrie de Taylor en l'interval [-1 , 1]. La sèrie de Taylor d'una funció qualsevol es defineix com la següent suma infinita:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^n(a)}{n!}(x-a)^n$$

On n! és el factorial d'n i $f^n(a)$ indica l'enèssima derivada d'f en el punt a.

Obtenim doncs que el desenvolupament es el següent:

$$\tanh x = \sum_{n=1}^{\infty} \frac{B_{2n} 4^n (4^n - 1)}{(2n)!} x^{2n-1}$$

On $B_{2n}$ són els Nombres de Bernuilli, que es defineixen mitjançant la següent fòrmula recursiva:

$$\begin{cases} B_0 = 1 \\ B_m = -\sum_{j=0}^{m-1} \binom{m}{j} \frac{B_j}{m+1-j} \end{cases}$$

Una altra forma d'obtindre un desenvolupament de Taylor, mes simple i mes eficient l'hora de calcular el seu valor, es utilitzar la Formula d'Euler del nombre $e^x$:

$$e^{ix} = \cos x + i \operatorname{sen} x$$

D'aquí obtenim:

$$\cos x = \frac{e^{ix} + e^{-ix}}{2} = Re(e^{ix})$$

$$\operatorname{sen} x = \frac{e^{ix} - e^{-ix}}{2i} = Im(e^{ix})$$

$$\cosh x = \cos ix = \frac{e^x + e^{-x}}{2}$$

$$\sinh x = -i \sin ix = \frac{e^x - e^{-x}}{2}$$

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{\frac{e^x - e^{-x}}{2}}{\frac{e^x + e^{-x}}{2}} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

El desenvolupament en sèrie de Taylor per a la funció $e^x$ és:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

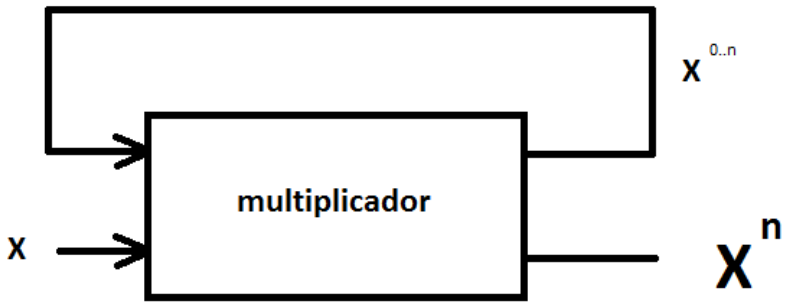
$$e^{-x} = \sum_{n=0}^{\infty} (-1)^n \frac{x^n}{n!}$$

Per tant, combinant les tres fórmules anteriors obtenim:

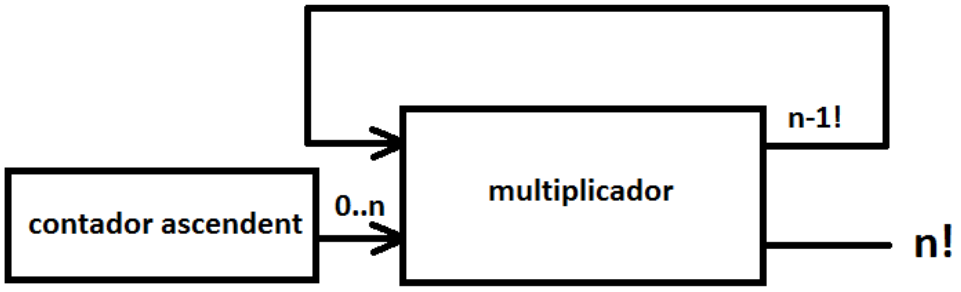
$$\mathbf{\tanh x} = \frac{\sum_{n=0}^{\infty} \frac{x^n}{n!} - \sum_{n=0}^{\infty} (-1)^n \frac{x^n}{n!}}{\sum_{n=0}^{\infty} \frac{x^n}{n!} + \sum_{n=0}^{\infty} (-1)^n \frac{x^n}{n!}} = \frac{\sum_{n=0}^{\infty} \frac{x^{2n+1}}{2n+1!}}{\sum_{n=0}^{\infty} \frac{x^{2n}}{2n!}} = \sum_{n=0}^{\infty} \frac{\frac{x^{2n+1}}{2n+1!}}{\frac{x^{2n}}{2n!}}$$

# Disseny del circuit de càlcul de la tangent hiperbòlica

Si observem el resultat del desenvolupament en sèrie, veiem que es tracta de calcular potències d'x i dividir-les pel factorial del nombre al qual hem elevat l'x. Així doncs, podem pensar en que un dels mòduls del nostre disseny consistirà en un multiplicador per una constant, que serà x, amb l'entrada realimentada obtenint així les potències. I un altre dels mòduls serà un multiplicador d'enters, una de les entrades del qual serà la realimentació del resultat i l'altra provindrà d'un comptador ascendent, així que d'aquest tindrem els factorials d'n. Veiem-ho a les següents il·lustracions:

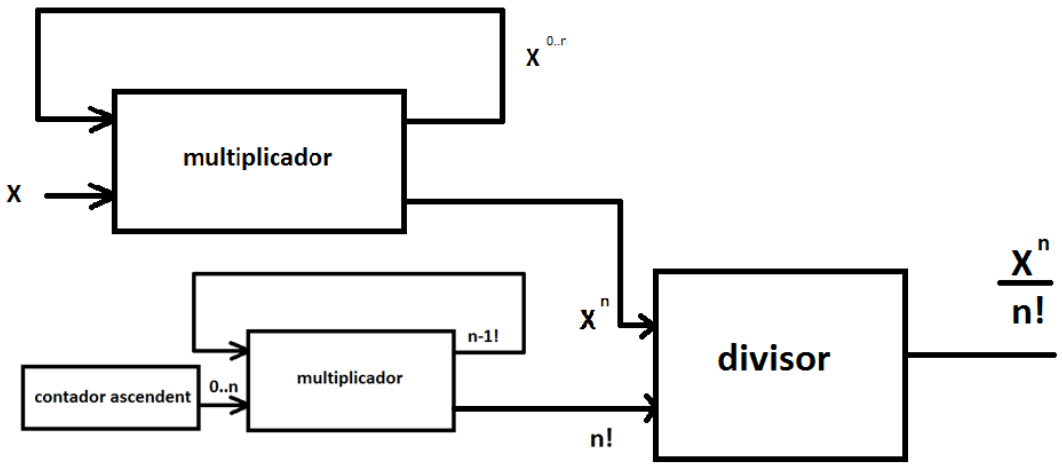



Mòdul per a calcular les potències d'x



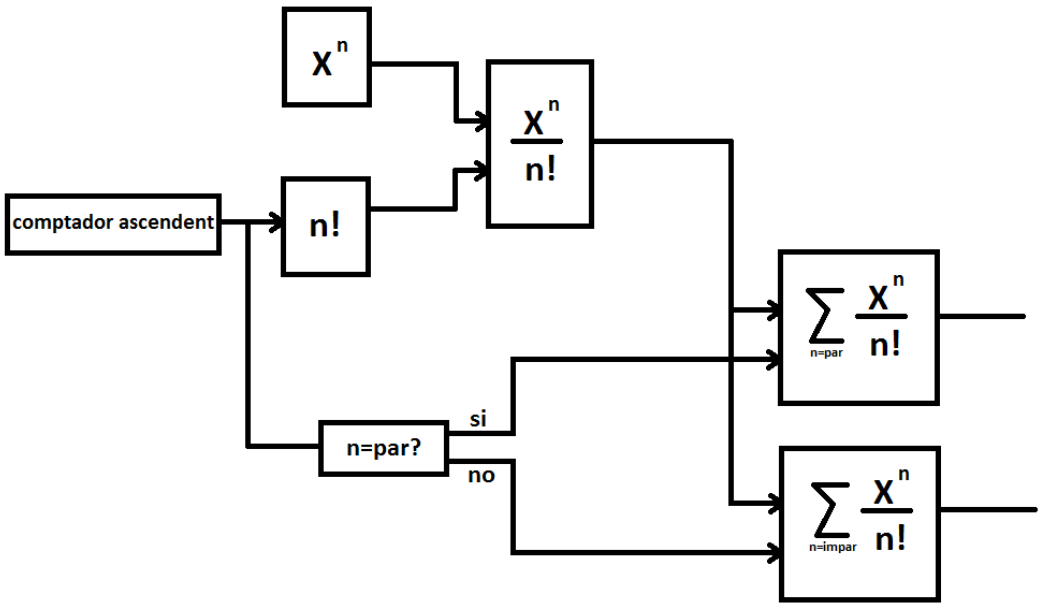

Mòdul per a calcular n!

Connectant les eixides d'ambdós mòduls a un divisor, obtindrem factors de la forma $\frac{x^n}{n!}$ i ja estarem més pròxims del disseny complet.




Mòdul per al càlcul dels factors $\frac{x^n}{n!}$

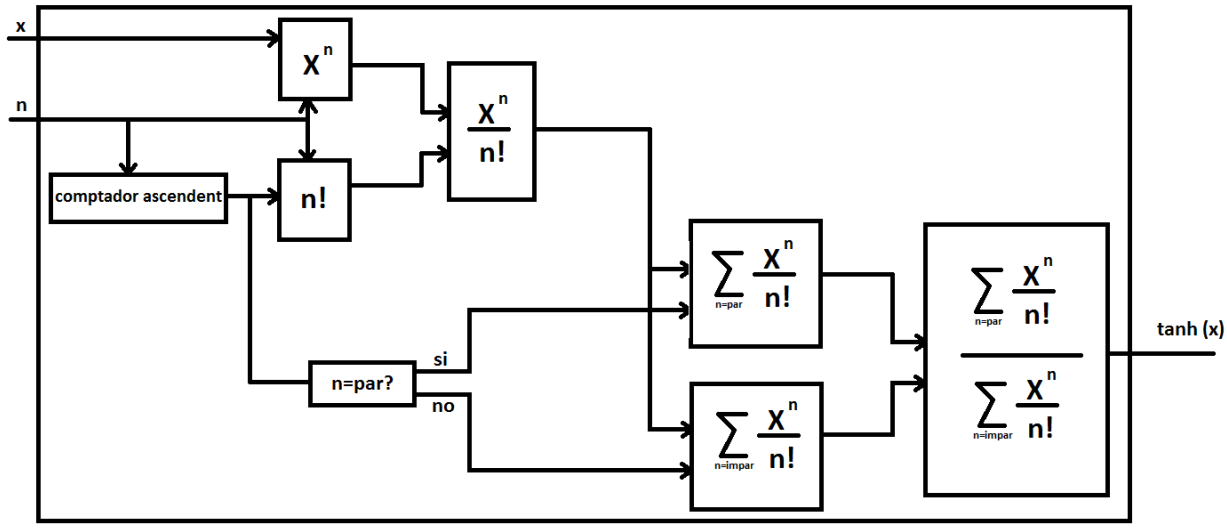
Una vegada obtinguts aquestos factors, el que farem serà dur-los a dos sumadors diferents amb acumulador, per tal d'obtindre d'una banda el sumatori dels factors parells (de potència i factorial parell) i per altra banda els factors d'index imparell. Per a separar els factors parells dels imparells, utilitzarem l'eixida del comptador ascendent utilitzat per al càlcul del factorial. Representarem els mòduls anteriors amb caixes amb menys detall que a les anteriors il·lustracions per tal de clarificar el disseny. Quedaria com segueix:




Càlcul dels factors parells i senars

Sols ens falta, doncs fer la divisió dels sumatoris dels factors parells i imparells més algunes senyals de control. La entrada n, és el nombre de factors a calcular, ja que el sumatori és infinit però amb uns quants factors obtindrem la resolució necessària per als nostres càlculs. Aquesta entrada es passa als mòduls $x^n$ i n! per a indicar, quan han de parar de calcular potències i fatorials respectivament. També es passa a l'entrada del comptador per a que quan arribe a n-1, el compte es reinicie.

El disseny complet del circuit és el següent (s'han deixat de representar algunes senyals, com per exemple, el rellotge que controla tot el circuit o el reset assíncron):




Algún dels avantatges que té aquest disseny és que el càlcul de cadascuna de les potències i els factorials, i amb això els factors de potència entre factorial, es calcula a partir dels termes anteriors, amb la qual cosa l'eficiència augmenta. Amés, l'entrada n, ens permet ajustar la resolució tant com ens siga necessari, encara que aquest ajust s'haurà de fer abans de sintetitzar el circuit (s'explicarà més endavant).

# La representació IEEE-754 elegida

Una vegada entesa la representació de nombres en coma flotant proposada per l'IEEE, veiem quina és la representació més adequada als nostres interessos. Veiem que el circuit està dissenyat de forma genèrica sense definir cada entrada quina grandària ha de tindre, així l'elecció, del tamany de l'entrada "x" ha d'estar estudiada.

Per tal d'escollir bé la nostra representació, s'han tingut en compte diversos factors. El principal factor a tindre en ment és la presició dels resultats que es volen obtindre. Per això, s'han estudiat diferents longituds per a la grandària en nombre de bits tant de la mantissa com de l'exponent. Un altre dels factors a tenir en compte és com varia la freqüència dels circuits en cada FPGA.

Tenint en compte aquestos dos factors, s'ha estimat que la representació més adequada per a nosaltres és una representació on la mantissa té un tamany de set bits i l'exponent un tamany de quatre bits, obtenint en total una representació basada en l'IEEE-754 però amb només dotze bits. Aquesta forma escollida de representació dels nombres és un balanç entre una representació amb el mínim possible nombre de bits, per tal de minimitzar el període de funcionament del circuit i així maximitzar la freqüència, però amb el nombre suficient de bits per tal de que l'error estiga en la quarta xifra decimal. A més, els DSPs que incorporen algunes de les FPGAs que utilitzem, incorporen multiplicador, sumadors i acumuladors de almenys 18 bits, així que seran suficients per a no disminuir de forma innecessària la freqüència màxima a la que pot funcionar el circuit.

# Consideracions del nostre codi VHDL

Una altra de les entrades que no està definida en quan tamany al circuit, és l'entrada "n", que com ja s'ha comentat abans servix per a indicar al circuit el nombre de termes de la sèrie de Taylor, que en principi és infinita i per tant hem de truncar. Per tant, s'ha realitzat un altre estudi, que podeu seguir al fitxer d'excel inclós en el CD del projecte, on es representa en tot el domini [-1, 1], l'error que es produix per truncament en funció del nombre de termes de la suma infinita escollits.

Aquesta vegada s'ha determinat que el nombre de termes més adequat per a que l'error no influisca en els resultats que volem obtindre és de vuit.

Tenint en compte que tant el nombre de termes com la representació per als nombres, és a dir, les entrades "x" i "n" podrien canviar en un futur, s'ha utilitzat en el codi VHDL de l'entrada de disseny el que s'anomenen els *paràmetres genèrics.* Aquests tipus de paràmetres són utilitzats en VHDL per a escriure models flexibles utilitzant una llista de constants, de manera que el codi que genèric és reutilitzable. Es pot utilitzar, per exemple, per definir un bus general d'amplada variable.

En el nostre cas, hem fet el codi en genèric, per tal de que no s'hagi de reescriure el codi en cas de que es desitge canviar la representació de l'entrada "x" o el nombre de termes a calcular "n" del desenvolupament en sèrie de la tangent hiperbòlica. Hem definit n com a una entrada de 3 entrades, la qual podrà representar valors en binari natural entre 0 i 7.

A banda d'aquestes consideracions, el nostre codi ha estat codificat com una màquina d'estats fent ús de la seua descripció comportamental. Els estats estan controlats pel rellotge, i s'han afegit alguns estats d'espera després de realitzar operacions costoses.

Una altra cosa a tindre en compte és que el llenguatge VHDL-93 no incorpora suport per aritmètica en coma flotant, així que s'ha fet ús d'unes llibreries, que poc a poc van incorporant-se als compiladors més recents de VHDL-200X, amb una versió compatible per a VHDL-93. Els codis font s'incorporaran al CD.

Per a fer ús d'aquests fitxers font, s'ha de crear una llibreria d'usuari en cada un dels entorns anomenada ieee_proposed i afegir els tres fonts del CD (fixed_float_types_c.vhd, fixed_pkg_c.vhd i float_pkg_c.vhd). La documentació està disponible al següent enllaç: http://www.vhdl.org/fphdl/

S'ha de tindre en compte també, que per a que el ModelSim puga utilitzar també aquesta llibreria, s'ha de compilar prèviament amb el comandament "vlib".

# El nostre fluxe de disseny

El fluxe de disseny, junt a cadascuna de les ferramentes que hem utilitzat és el següent:

1. Hem dissenyat el circuit electrònic capaç de calcular la tangent electrònica d'un nombre real. El procés està descrit amb més detalls al capítol "Disseny del circuit de càlcul de la tangent hiperbòlica". Bàsicament el que s'ha fet és dividir en mòduls sencills, utilitzant la realimentació d'entrades per a millorar la eficiència.

2. L'entrada del disseny ha estat duta a terme mitjançant una descripció comportamental del circuit dissenyat a l'anterior punt. S'ha utilitzat per a això el llenguatge de descripció de hardware VHDL. La ferramentes utilitzades han estat els editors de HDL integrats amb cadascun dels entorns de desenvolupament de cada fabricant, el Libero v9.0 per a les FPGA d'Actel, el Xilinx ISE WebPack 12.1 per a les FPGA de Xilinx i l'Altera Quartus II per a les de Altera.

3. La simulació funcional es va portar a terme amb el ModelSim en les diferents versions segons el fabricant de la FPGA. Amb el Libero hem utilitzat el ModelSim Actel 6.5d, amb el Xilinx ISE el ModelSim Xillinx Edition - III Starter, i el ModelSim-Altera Starter Edition amb el Quartus II. Per a comprovar el correcte funcionament del circuit, hem creat una taula excel amb els valors esperats de la

tangent hiperbòlica depenent del nombre de termes agafats en el desenvolupament de la sèrie de Taylor per a la funció. Ací tenim una captura de pantalla d'una de les simulacions.
Totes elles provocaren el mateix resultat, així que comprovarem que el comportament del circuit era l'esperat.

4. La síntesi es va dur a terme amb la ferramenta synplify, que es pot integrar en tots els tres paquets distribuits pel fabricant d'FPGAs. Depenent de la FPGA utilitzada, els resultats eren molt diferents, degut a les diferents prestacions de cadascuna d'elles. Els resultats es veuran en l'apartat següent.

5. Les simulacions post-síntesi provocaren els mateixos resultats que les simulacions pre-síntesi i es dugueren a terme amb el corresponent ModelSim segons l'FPGA. L'única cosa es va tindre en compte perque variava entre les diferents simulacions era la freqüència a la que el circuit podia funcionar segons els resultats de les síntesis. Així, tots els resultats foren igualment correctes, encara que unes FPGAs feien la tasca més ràpidament que altres.

6. A partir d'aquesta etapa del flux, només es porta a terme amb la primera FPGA, ja que ja tenim una idea aproximada de quins serien els costos temporals i espacials del disseny amb cadascuna de les FPGAs, que era l'objectiu al que voliem arribar.
   Per a dur a terme el place & routing de la FPGA de l'SmartFusion, utilitzem el Designer incorporat amb el paquet del Libero. Els resultats temporals són els mateixos, així que la freqüència a la que pot funcionar el circuit és la de l'etapa de la síntesi.

7. El back-annotation la realitza també el Designer del Libero, i crea els fitxers tanh_ba.sdf i tanh_ba.vhdl per a poder realitzar la simulació temporal, que és correcta.

8. La simulació temporal es realitza de nou amb el ModelSim Actel 6.5d, produint els resultats esperats.

9. El fitxer de programació de la FPGA, el genera el Designer del Libero, però per a programar la FPGA, s'utilitza el FlashPro, integrat amb el paquet del Libero. Encara que la FPGA està programada, no es pot posar realment a fer càlculs, ja que faltaria per dissenyar tot el procés de l'entrada de dades i la recollida de resultats, cosa que queda fora de l'àmbit del present projecte.

# Les FPGA utilitzades

Una vegada introduït els fonaments de les FPGA, veurem més a fons cadascuna de les FPGA que hem utilitzat.

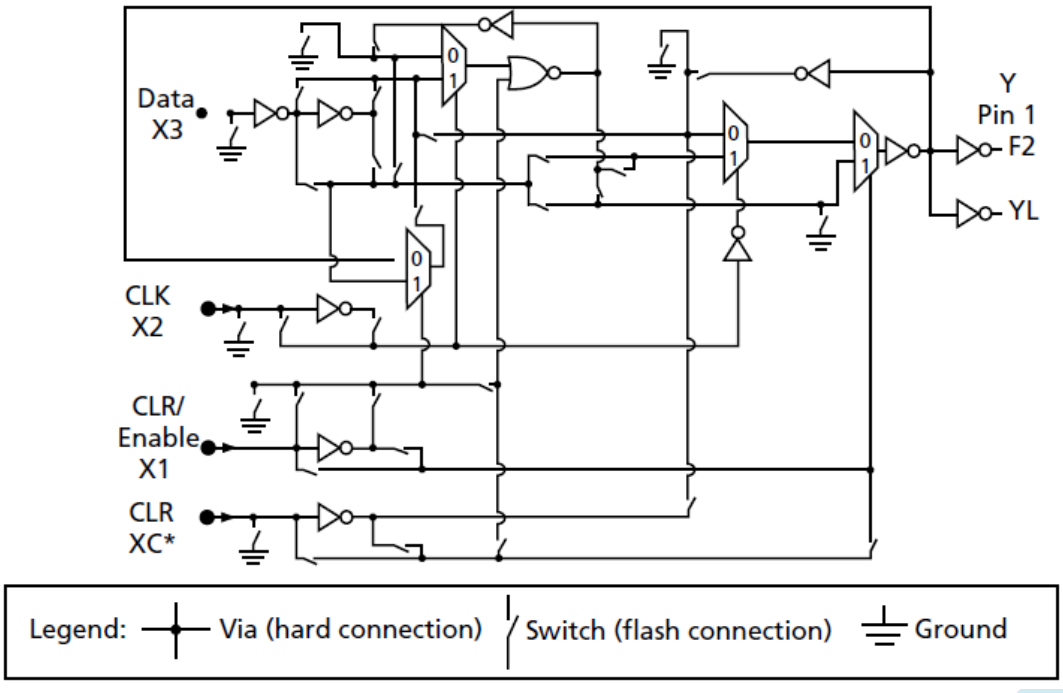
## FPGAs d'Actel

### *A2F500M3G*

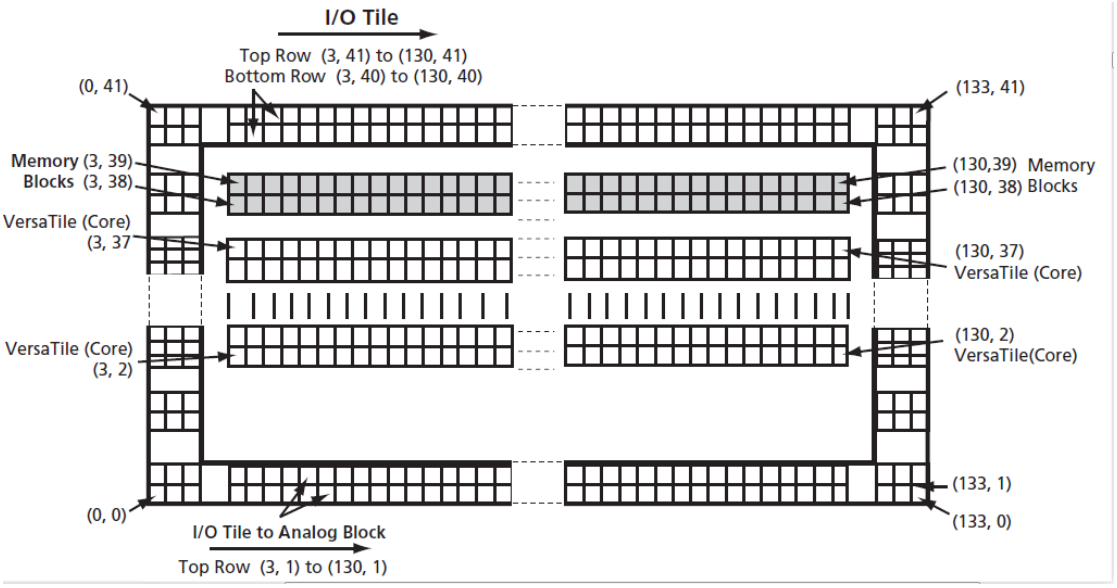
La primera de les FPGA sobre la que hem avaluat el disseny és una FPGA del fabricant ACTEL, inclosa en un paquet d'avaluació anomenat SmartFusion Evaluation Kit. Aquest paquet no consta només d'una FPGA. Incorpora un microprocessador ARM Cortex-M3 i va amb una placa amb perifèrics diversos. Permet escollir entre unes poques FPGAs basades en la mateixa tecnologia i que solament es diferèncien per la quantitat de recursos. En el nostre cas hem escollit el dispositiu A2F500M3G, amb un empaquetat 484 FBGA. Les característiques principals d'aquesta FPGA són les següents:

| | |
|---|---|
| Nombre de portes (system gate) | 500,000 |
| Nombre de biestables (D-flip-flops) | 11,520 |
| Blocs de memòria RAM (4,608 bits) | 24 |
| Freqüència màxima de funcionament | 80 MHz |

Cada cella de l'FPGA té una estructura com la següent:



Aquest tipues de celles d'Actel s'anomenen VersaTile. L'estructura de l'Array és la següent:




Per a més informació, es poden consultar els datasheets disponibles en la pàgina web d'Actel (www.actel.com).

Una vegada feta la síntesi del nostre disseny VHDL, els resultats tant de cost temporal com espacial són per a aquesta FPGA els següents:

| Starting Clock | Requested Frequency | Estimated Frequency | Requested Period | Estimated Period | Slack | Clock Type | Clock Group |
|---|---|---|---|---|---|---|---|
| tanh\|clk | 9.2 MHz | 7.8 MHz | 108.665 | 127.842 | -19.176 | inferred | Autoconstr_clkgroup_0 |

Core Cells : 3871 of 11520 (34%)
IO Cells : 36 of 128 (28%)

```
Compile report:
===============

    Microcontroller Subsystem  Used:      0  Total:      1   (0.00)
    Fabric                     Used:   3660  Total:  11520   (31.77)
    Fabric IO (W/ clocks)      Used:     36  Total:    128   (28.13)
    Fabric Differential IO     Used:      0  Total:     64   (0.00)
    Dedicated Analog IO        Used:      0  Total:     43   (0.00)
    Dedicated MSS IO           Used:      0  Total:     43   (0.00)
    GLOBAL (Chip+Quadrant)     Used:      4  Total:     15   (26.67)
    MSS GLOBAL                 Used:      0  Total:      3   (0.00)
    Fabric CCC                 Used:      0  Total:      1   (0.00)
    MSS CCC                    Used:      0  Total:      1   (0.00)
    RC oscillator              Used:      0  Total:      1   (0.00)
    XTL oscillator             Used:      0  Total:      1   (0.00)
    RAM/FIFO                   Used:      0  Total:     24   (0.00)
    User JTAG                  Used:      0  Total:      1   (0.00)



Global Information:

    Type             | Used    | Total
    -----------------|---------|-------------
    Chip global      | 4       | 3  (133.33)
    Quadrant global  | 0       | 12 (0.00)
    MSS global       | 0       | 3  (0.00)



Core Information:

    Type     | Instances     | Core tiles
    ---------|---------------|-----------
    COMB     | 3529          | 3529
    SEQ      | 131           | 131




I/O Function:

    Type                           | w/o register  | w/ register  | w/
DDR register
    -------------------------------|---------------|--------------|----
------------
    Input I/O                      | 19            | 0            | 0
    Output I/O                     | 17            | 0            | 0
    Bidirectional I/O              | 0             | 0            | 0
    Differential Input I/O Pairs   | 0             | 0            | 0
    Differential Output I/O Pairs  | 0             | 0            | 0
```

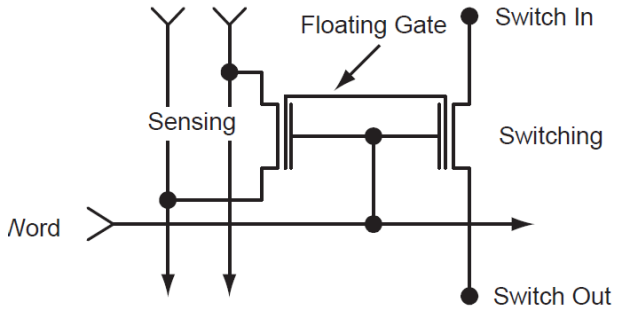
```
I/O Placement:

    Locked  :    0
    Placed  :    0
    UnPlaced:   36 (  100.00)
```

Les conclusions més importants dels resultats obtinguts poden veure's marcades en roig, i són la freqüència màxima a la que pot funcionar el circuit, que és de 7,8 MHz, i el nombre de bancs de recursos que s'haurien d'utilitzar per a implementar físicament el circuit, que serien 4, mala notícia ja que només es disposa de 3 aquesta FPGA. Així concloem que no ens serveix aquest dispositiu per al nostre propòsit perquè no disposa dels recursos suficients.

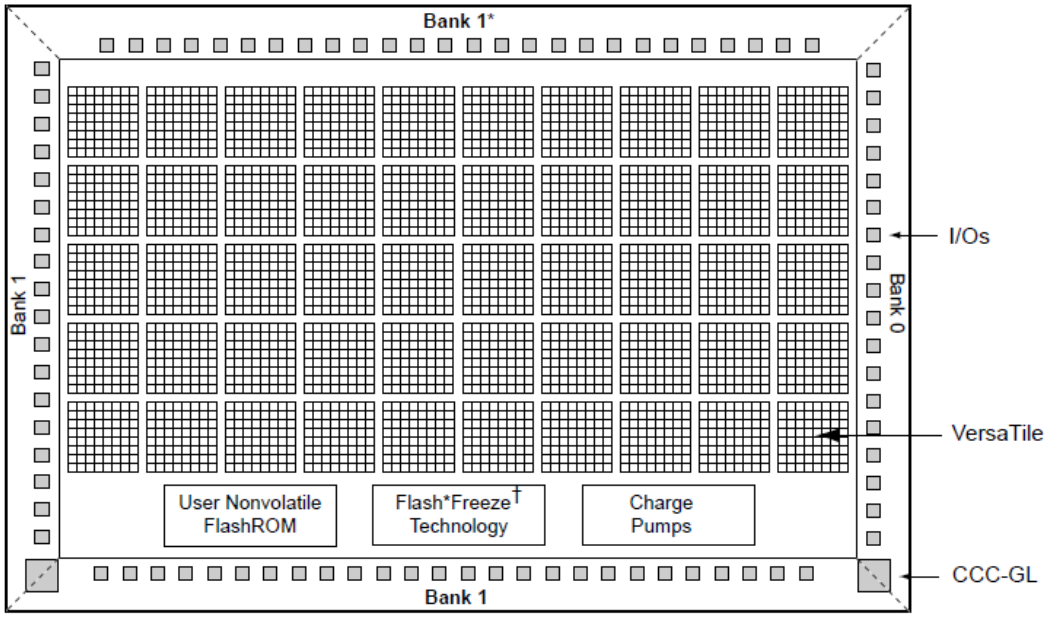
### *RT3PE600L*

La segona de les FPGAs escollida és una altra de les FPGA d'Actel, aquesta vegada és la RT3PE600L, de la família de les ProASIC3L, amb un empaquetatge 484 LGA. Aquesta FPGA utilitza switchos flash per a la seua programació, que la doten apta per a la reprogramació no volátil, i les seues cel·les són cel·les idèntiques a les anteriors (VersaTile de Actel). Algunes de les característiques d'aquesta FPGA són les següents:




Switch flash de la RT3PE600L

La seua estructura està formada per 30K bancs com el següent:




Banc de la RT3PE600L

| Starting Clock | Requested Frequency | Estimated Frequency | Requested Period | Estimated Period | Slack | Clock Type | Clock Group |
|---|---|---|---|---|---|---|---|
| tanh\|clk | 12.3 MHz | 10.5 MHz | 81.151 | 95.472 | -14.321 | inferred | Autoconstr_clkgroup_0 |

IO Cell usage:

| cell | count |
|---|---|
| CLKBUF | 1 |
| INBUF | 18 |
| OUTBUF | 17 |
| TOTAL | 36 |

Core Cells : 3621 of 13824 (26%)
IO Cells : 36

```
There were 0 error(s) and 0 warning(s) in this design.
=====================================================================
Compile report:
===============

    CORE                        Used:   3429  Total:  13824   (24.80%)
    IO (W/ clocks)              Used:     36  Total:    270   (13.33%)
    Differential IO             Used:      0  Total:    135   (0.00%)
```

```
    GLOBAL (Chip+Quadrant)     Used:      4  Total:     18   (22.22%)
    PLL                        Used:      0  Total:      6   (0.00%)
    RAM/FIFO                   Used:      0  Total:     24   (0.00%)
    Low Static ICC             Used:      0  Total:      1   (0.00%)
    FlashROM                   Used:      0  Total:      1   (0.00%)
    User JTAG                  Used:      0  Total:      1   (0.00%)
```

```
Global Information:

    Type            | Used   | Total
    ----------------|--------|-------------
    Chip global     | 4      | 6  (66.67%)
    Quadrant global | 0      | 12 (0.00%)
```

```
Core Information:

    Type    | Instances    | Core tiles
    --------|--------------|-----------
    COMB    | 3297         | 3297
    SEQ     | 132          | 132
```

```
I/O Function:

    Type                          | w/o register  | w/ register  | w/
DDR register
    ------------------------------|---------------|--------------|----
------------
    Input I/O                     | 19            | 0            | 0
    Output I/O                    | 17            | 0            | 0
    Bidirectional I/O             | 0             | 0            | 0
    Differential Input I/O Pairs  | 0             | 0            | 0
    Differential Output I/O Pairs | 0             | 0            | 0
```

```
I/O Technology:

                                    |   Voltages    |             I/Os
    --------------------------------|-------|-------|-------|--------
|--------------
    I/O Standard(s)                 | Vcci  | Vref  | Input | Output |
Bidirectional
    --------------------------------|-------|-------|-------|--------
|--------------
    LVCMOS12                        | 1.20v | N/A   | 19    | 17     |
0
```

```
I/O Placement:

    Locked  :   0
    Placed  :   0
    UnPlaced:  36 ( 100.00% )
```

Si observem els resultats obtinguts i els comparem amb els resultats per a l'anterior FPGA, observem que la freqüència a la que pot funcionar els circuit amb aquesta FPGA ha augmentat un poc, fins als 10,5 MHz, i el més important, aquesta FPGA, sí que té prou recursoso per a poder implementa físicament el circuit.

## FPGAs de Xilinx

### *XC6VHX565T*

La tercera de le FPGAs, és una FPGA de Xilinx, de la família Virtex6. Concretament és la XC6VHX565T, amb un empaquetat ff1923. Les seues principals característiques es poden veure a la següent taula:

| | |
|---|---|
| Cel·les lògiques | 566784 |
| CLBs (blocs lògics configurables) | 88560 |
| DSPs | 864 |
| E/E d'usuari | 720 |
| Freqüència màxima | 1600 MHz |

Cada CLB conté quatre LUTs (lookup tables) i huit biestables flip-flop.

Cada DSP consta d'un multiplicador, un sumador i amb acumulador de 25 x 18 bits en complement a dos.

Per a més informació podeu consultar el datasheet a www.xilinx.com

Els resultats de la síntesi són els següents:

```
Summary:
        inferred   1 Multiplier(s).
        inferred  67 Adder/Subtractor(s).
        inferred 121 D-type flip-flop(s).
        inferred  55 Comparator(s).
        inferred 479 Multiplexer(s).
        inferred  20 Combinational logic shifter(s).
        inferred   1 Finite State Machine(s).
Unit <tanh> synthesized.
```

Device utilization summary:

```
---------------------------
Selected Device : 6vhx565tff1923-2
Slice Logic Utilization:
 Number of Slice Registers:              164  out of  708480    0%
 Number of Slice LUTs:                   1395  out of  354240    0%
 Number used as Logic:                   1395  out of  354240    0%
Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:     1440
 Number with an unused Flip Flop:        1276  out of   1440    88%
 Number with an unused LUT:              45  out of   1440     3%
 Number of fully used LUT-FF pairs:      119  out of   1440     8%
 Number of unique control sets:          13
 IO Utilization:
 Number of IOs:                          36
 Number of bonded IOBs:                  36  out of    720     5%
 Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:               1  out of     32     3%
 Number of DSP48E1s:                     1  out of    864     0%
```

Timing Summary:

```
---------------
  Speed Grade: -2
  Minimum period: 16.909ns (Maximum Frequency: 59.140MHz)
  Minimum input arrival time before clock: 12.829ns
  Maximum output required time after clock: 0.659ns
```

Dels resultats obtinguts veiem que l'ús dels recursos és mínim, al voltant d'un 2% així que es podria replicar el disseny per a que no sols es fera un càlcul, sinó que es feren càlculs en paral·lel. Fent una estimació, podrien fer-se fins al voltant d'una vintena de càlculs a la vegada. Observem també que la freqüència del circuit ha augmentat unes cinc vegades fins als 59.140MHz degut a les majors prestacions d'aquest dispositiu.

### *XC6SLX150T*

L'altre dispositiu de Xilinx que hem utilitzat és de la família Spartan6. L'FPGA és la XC6SLX150T amb un empaquetat FGG900. Les seues principals carácterístiques són les següents:

| | |
|---|---|
| Cel·les lògiques | 147443 |
| CLBs (blocs lògics configurables) | 184304 |

| | |
|---|---|
| DSPs | 180 |
| E/E d'usuari | 540 |
| Freqüència màxima | 1000 MHz |

Cada CLB conté quatre LUTs (lookup tables) i huit biestables flip-flop.

Cada DSP consta d'un multiplicador, un sumador i un acumulador de 18 x 18 bits en complement a dos.

Per a més informació podeu consultar el datasheet a www.xilinx.com

Els resultats de la síntesi són els següents:

```
Summary:
        inferred   1 Multiplier(s).
        inferred  67 Adder/Subtractor(s).
        inferred 121 D-type flip-flop(s).
        inferred  55 Comparator(s).
        inferred 479 Multiplexer(s).
        inferred  20 Combinational logic shifter(s).
        inferred   1 Finite State Machine(s).
    Unit <tanh> synthesized.

Device utilization summary:
---------------------------
Selected Device :                   6slx150tfgg900-3
Slice Logic Utilization:
 Number of Slice Registers:         180  out of  184304    0%
 Number of Slice LUTs:              1474  out of  92152    1%
    Number used as Logic:           1474  out of  92152    1%
Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    1522
   Number with an unused Flip Flop:     1342  out of   1522   88%
   Number with an unused LUT:           48  out of   1522    3%
   Number of fully used LUT-FF pairs:   132  out of   1522    8%
   Number of unique control sets:       13
IO Utilization:
 Number of IOs:                     36
 Number of bonded IOBs:             36  out of    540    6%
Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:          1  out of    16    6%
 Number of DSP48A1s:                1  out of    180    0%

Timing Summary:
---------------
Speed Grade: -3
   Minimum period: 33.331ns (Maximum Frequency: 30.002MHz)
   Minimum input arrival time before clock: 25.240ns
   Maximum output required time after clock: 3.819ns
```

Al igual que passava amb l'anterior dispositiu de Xilinx, els recursos emprats en aquest són mínims i es podria replicar també el circuit al voltant d'unes vint vegades per a fer càlculs paral·lelament. La freqüència observem que ha baixat esta vegada fins als 30 MHz aproximadament.

## FPGAs d'Altera

### *EP3SE50*

La següent FPGA que hem provat es del fabricant Altera. El dispositiu es de la familia Stratix III, anomenat EP3SE50 amb un empaquetat FBGA484. Algunes de les seues característiques es poden veure a la següent taula:

| | |
|---|---|
| Cel·les lògiques | 338000 |
| CLBs (blocs lògics configurables) | 38000 |
| DSPs | 48 |
| E/E d'usuari | 296 |
| Freqüència màxima | 1300 MHz |

Cada DSP conté operadors per a 18 bits.

Els resultats de la síntesi són els següents:

```
Logic utilization                   5 %
Combinational ALUTs                 1,492 / 38,000 ( 4 % )
Memory ALUTs                        0 / 19,000 ( 0 % )
Dedicated logic registers           129 / 38,000 ( < 1 % )
Total registers                     129
Total pins                          36 / 296 ( 12 % )
Total virtual pins                  0
Total block memory bits             0 / 1,880,064 ( 0 % )
DSP block 18-bit elements           1 / 216 ( < 1 % )
Total PLLs                          0 / 4 ( 0 % )
Total DLLs                          0 / 4 ( 0 % )
Device  EP3SL50F484C2

Fmax                                63.67 MHz
```

Com es pot observar dels resultats, amb aquesta FPGA podríem replicar el circuit al voltant d'unes vint vegades de nou, i a més a més, la freqüència ha pujat fins als 63.67 MHz, la major fins al moment.

### *EP4CE15F23C6*

La última de les FPGA és una altra del fabricant Altera, en aquest cas de la família Cyclone IV. El dispositiu és el EP4CE15F23C6 amb un empaquetat FBGA484. Les característiques principals són les següents:

| | |
|---|---|
| Cel·les lògiques | 15408 |
| 18 x 18 multiplicadors | 56 |
| E/E d'usuari | 343 |
| Freqüència màxima | 600 MHz |

```
Total logic elements                    1804 / 15,408  ( 12 % )
Total combinational functions           1,804 / 15,408 ( 12 % )
Dedicated logic registers               129 / 15,408 ( < 1 % )
Total logic elements                    1,804 / 15,408 ( 12 % )
Total registers                         129
Total pins                              36 / 344 ( 10 % )
Total virtual pins                      0
Total memory bits                       0 / 516,096 ( 0 % )
Embedded Multiplier                     9-bit elements 1 / 112 ( < 1 % )
Total PLLs                              0 / 4 ( 0 % )

Fmax                                    33,75 MHz
```

Aquesta vegada, la utilització dels reursos de la FPGA, sols ens permetria replicar el circuit unes huit vegades, reduint la taxa de còmput respecte a alguns dels altres dispositius. A part, la freqüència a baixat fins als 33.75 MHz.

# Resultats, conclusions i possibles treballs futurs

Ací tenim una taula comparativa on apareixen els principals resultats als que hem arribat en el nostre projecte:

| FPGA | Fabricant | Recursos utilitzats (% del total disponibles) | Freqüència màxima estimada (MHz) | Preu aproximat al mercat ($ americans) |
|---|---|---|---|---|
| A2F500M3G | Actel | 133 .3 | 7.8 | 75.61 |
| RT3PE600L | Actel | 66.7 | 10.5 | 26.20 |
| XC6VHX565T | Xilinx | 8 | 59.14 | 7400.00 |
| XC6SLX150T | Xilinx | 8 | 30.0 | 197.34 |
| EP3SE50 | Altera | 4 | 63.67 | 761.00 |
| EP4CE15F23C6 | Altera | 12 | 33.75 | 34.71 |

Si tenim en compte el nombre de vegades que es pot replicar aproximadament el nostre circuit en cada FPGA i la freqüència màxima a la que el circuit pot funcionar en cadascuna d'elles, podem calcular un índex amb la següent fórmula, que serà un indicatiu de quina és la millor opció a escollir:

$$índex\ 1 = nombre\ de\ vegades\ que\ podem\ replicar\ circuit * freqüència\ \ màx.circuit$$

Si també tenim en compte el preu, podem calcular un segon índex amb la següent fórmula:

$$índex\ 2 = \frac{nombre\ de\ vegades\ que\ podem\ replicar\ circuit * freqüència\ \ màx.circuit}{preu\ al\ mercat}$$

Els resultats són els següents (un major índex indica major conveniència):

| FPGA | Índex 1 | Índex 2 |
|---|---|---|
| A2F500M3G | 0 | 0 |
| RT3PE600L | 10.5 | 0.4 |
| XC6VHX565T | 710 | 0.095 |
| XC6SLX150T | 360 | 1.82 |
| EP3SE50 | 1592 | 2.1 |
| EP4CE15F23C6 | 270 | 7.8 |

Aquest valors es poden prendre com a referència, ja que la fórmula dóna el mateix pes a cadascuna de les variables, però es podrien valorar els pesos adequats a cadascuna d'aquestes

variables. El que sembla clar és que les FPGAs d'Altera semblen les més adequades als nostres propòsits.

Els possibles treballs futurs que es podrien realitzar basant-se en aquest projecte i com a continuació d'aquest podrien ser, per exemple, canviar la manera en que es fan els càlculs, utilitzant altre tipus de recursos (taules d'emmagatzematge, DSPs per a coma flotant, processadors multi nucli, etc).

# Referències

LANG, Tomás i ERCEGOVAC, Milos. Digital Arithmetic. Morgan Kaufmann Publishers, 2004.

IEEE Standard VHDL Language Reference Manual. ANSI Std 1076-1993. Publicat per l'IEEE en 1994.

31st Edition, D. Zwillinger (ed.), CRC, Boca Raton, 2003, 910 pages, ISBN 1-58488-291-3

**http://www.xilinx.com**

**http://www.altera.com**

**http://www.actel.com**