Capítol 6

Disseny VLSI del processador dedicat

En el capítol 4 s'ha obtingut el disseny d'un processador dedicat al càlcul d'un mapa d'acumulació, que forma part de l'algorisme de detecció de simetria presentat en el capítol 3, amb l'objectiu d'accelerar el procés d'obtenció d'aquest mapa. L'arquitectura obtinguda incorpora una matriu de cel·les processadores de píxel de dimensió proporcional a la resolució $M \times N$ de la imatge que es vol tractar (veure Fig 4.14). El nombre d'unitats a implementar és tan elevat que fa inviable una implementació amb un nombre raonablement reduït de dispositius lògics programables. Hem d'explorar, per tant, les possibilitats d'una implementació amb tecnologia d'alta escala d'integració *VLSI*. El objectius que ens proposem són els següents:

- Adaptació del disseny del processador per un implementació *VLSI*
- Estudiar la possibilitat d'obtenir el processador en un únic circuit *ASIC*
- Determinar els camins crítics per tal d'obtenir la freqüència màxima de treball i la capacitat de processament.

Quan es vol fer servir tecnologia d'alta escala d'integració (*VLSI*) per implementar un circuit, cal respectar algunes normes de disseny així com certes restriccions de fabricació de les famílies tecnològiques VLSI, per tal d'assegurar, que un cop implementat el circuit, el seu funcionament serà correcte i de fabricació viable. Ja que el

disseny obtingut en el capítol 4 és genèric, no té en compte aquestes restriccions i s'han d'introduir, per tant, les modificacions pertinents per tal d'adaptar-lo.

Pretenem esbrinar les possibilitats d'obtenir el processador en un sol circuit *VLSI* per la qual cosa durem a terme un primera experiència amb una arquitectura reduï da a *32 X 32* ceł les processadores de píxel, que facilita el disseny i ens permet obtenir una primera aproximació de l'àrea de silici necessària. Depenent dels resultats ens plantejarem l'ampliació de la matriu.

Donat que el nostre objectiu immediat no és fabricar un circuit si no veure la viabilitat i prestacions del processador dissenyat, hem decidit treballar amb ceł les estàndard per tal de facilitar el disseny i reduir-ne el temps d'obtenció. És evident que una implementació a mida (*Full Custom*) donaria un aprofitament del silici òptim, donada la regularitat del nostre disseny, però l'esforç per obtenir el resultat que perseguim és molt més gran.

Finalment, un cop obtingut el disseny, estudiarem els retards en la propagació dels senyals i obtindrem el camí amb més retard (camí crític) que imposarà el període mínim de rellotge possible i en conseqüència la freqüència màxima de treball i la capacitat de càlcul.

Cal fer notar, que del que acabem de dir dedueix que no pretenem obtenir un implementació a punt per posar-se en fabricació, sinó que fem un estudi que pretén obtenir la viabilitat i prestacions de l'arquitectura obtinguda. Així, hem deixat per treball futur consideracions de viabilitat de fabricació com és el cas de les consideracions de dissipació del circuit i testabilitat.

També cal tenir en compte que els resultats aquí obtinguts estan limitats per les famílies VLSI que estan al nostre abast, que no són les més actuals a causa del seu cost econòmic. Així doncs la potència de càlcul i sobretot l'àrea de silici necessària seran millorables. La tecnologia emprada en la primera prova és la de la família MIETEC 0.7μ amb dues capes de metall. Aquesta és la tecnologia de més alta escala de integració de que hem disposat en el moment d'iniciar aquest estudi.

6.1 Descripció de l'entorn

Per tal d'obtenir el disseny *VLSI,* s'ha utilitzat l'entorn Cadence que subministra Europractice, amb un cost reduï t, a les institucions acadèmiques o laboratoris de recerca associades que volen treballar amb microelectrònica. En el moment de iniciar el disseny hem tingut accés a una tecnologia de 0,7µm la qual no era la de més densitat d'integració existent. En el nostre cas hem utilitzat les instal·lacions, amablement cedides per el Departament d'electrònica de la UPC.

Sota el nom de Dessign Framework II, s'agrupen un conjunt d'aplicacions Unix destinades a completar un disseny *VLSI,* des del primer fins a l'últim dels passos que cal recorre per aconseguir la implementació. Els avantatges que presenten aquestes agrupacions d'aplicacions són la compartició d'una interfície d'usuari única, i la utilització d'una base de dades comuna on s'emmagatzema tota la informació referent a un mateix disseny. La interfície única evita que l'usuari, hagi d'aprendre les tasques comunes de cada aplicació, ja que seran sempre les mateixes. La base de dades comuna fa innecessàries les transformacions en la representació de les dades, estalvia temps i evita riscos de pèrdua de continguts durat el procés.

Per tal de facilitar els processos de disseny i de fabricació, les diferents firmes que es dediquen a la fabricació de circuits integrats proporcionen *kits* de disseny que s'incorporen al *Dessign Framework II*. Aquests *kits* consisteixen en un conjunt de llibreries, rutines, i fitxers descriptius de les diferents tecnologies suportades per el fabricant.

Hi ha dues vies per les quals es pot iniciar el desenvolupament d'una solució VLSI amb aquest entorn. La més formal és emprar el llenguatge de descripció de hardware per definir el comportament. A partir d'aquesta descripció es sintetitza de forma automàtica una llista de connexions, una vista esquemàtica, i una altra simbòlica.
Una alternativa és descriure el circuit per mitjà d'un esquema, mitjançant l'eina de captura d'esquemes i utilitzar directament les cel·les estàndard (*SC)* proporcionades per el fabricant o dissenys *FC (Full custom)* i fins i tot mixtes *FC/SC*

La primera opció té l'avantatge que és molt fàcil de portar a altres tecnologies, ja que un procés automàtic de síntesis transforma la descripció genèrica del llenguatge de descripció en una *Netlist* (llista de components i connexions) on es fan servir les cel·les del fabricant que se li indiquin.
La segona supera la primera en optimització si la complexitat no és elevada. En aquest cas un dissenyador experimentat sempre obtindrà un circuit més òptim que un procés automàtic de síntesis. El resultat però està lligat, a una tecnologia concreta i en molts casos és més laboriosa.

El fet que el nostre disseny és crític en àrea, a causa de l'alta replicació d'unitats, obliga a garantir un alt nivell d'optimització, per la qual cosa hem optat per un disseny directe en cel·les estàndard, tret d'alguns casos específics on és més òptima una síntesi automàtica. Aquest és el cas de les unitats constituïdes per funcions combinatòries ja que l'eina és molt hàbil en la síntesi automàtica en la simplificació de funcions.

El disseny obtingut s'ha de sotmetre a verificació per tal de comprovar que s'ajusta a les especificacions inicials. Per a la verificació del funcionament lògic s'ha utilitzat el simulador *Verilog-XL*, per a la verificació de les restriccions temporals el programa *Veritime.*

El verificador temporal comprova que totes les senyals respecten les restriccions de temps d'estabilització i manteniment de les cel·les que ho requereixen. També permet obtenir els camins crítics del circuit. Un camí crític és el recorregut d'un senyals amb el temps de propagació més llarg entre dos punts del circuit considerats com origen i destí respectivament. El camí crític amb el temps de propagació més alt de tots, determina la freqüència màxima de treball del circuit i condiciona, per tant, la capacitat de processament de l'arquitectura. Els resultats obtinguts es poden visualitzar en diagrames temporals a través del visor d'ones de forma que es facilita la seva interpretació. Aquest visor permet selecciona els senyals que es volen visualitzar i escollir l'escala temporal.
Abans de passar a l'obtenció del circuit físic del disseny verificat amb cel·les estàndard, l'entorn disposa d'una eina (*Hierarchy Browser*) que ens permet visualitzar els diferents nivells jeràrquics que té el nostre disseny, i decidir amb quins nivell de jerarquia es farà el procés de ubicació. Per altra banda, converteix les llistes de connexions (*Netlist*) en

els formats que utilitzen els processos típics posteriors consistents en la ubicació física dels mòduls i de les cel·les en el silici (*Placement*) i el seva connexió *(Routing*).

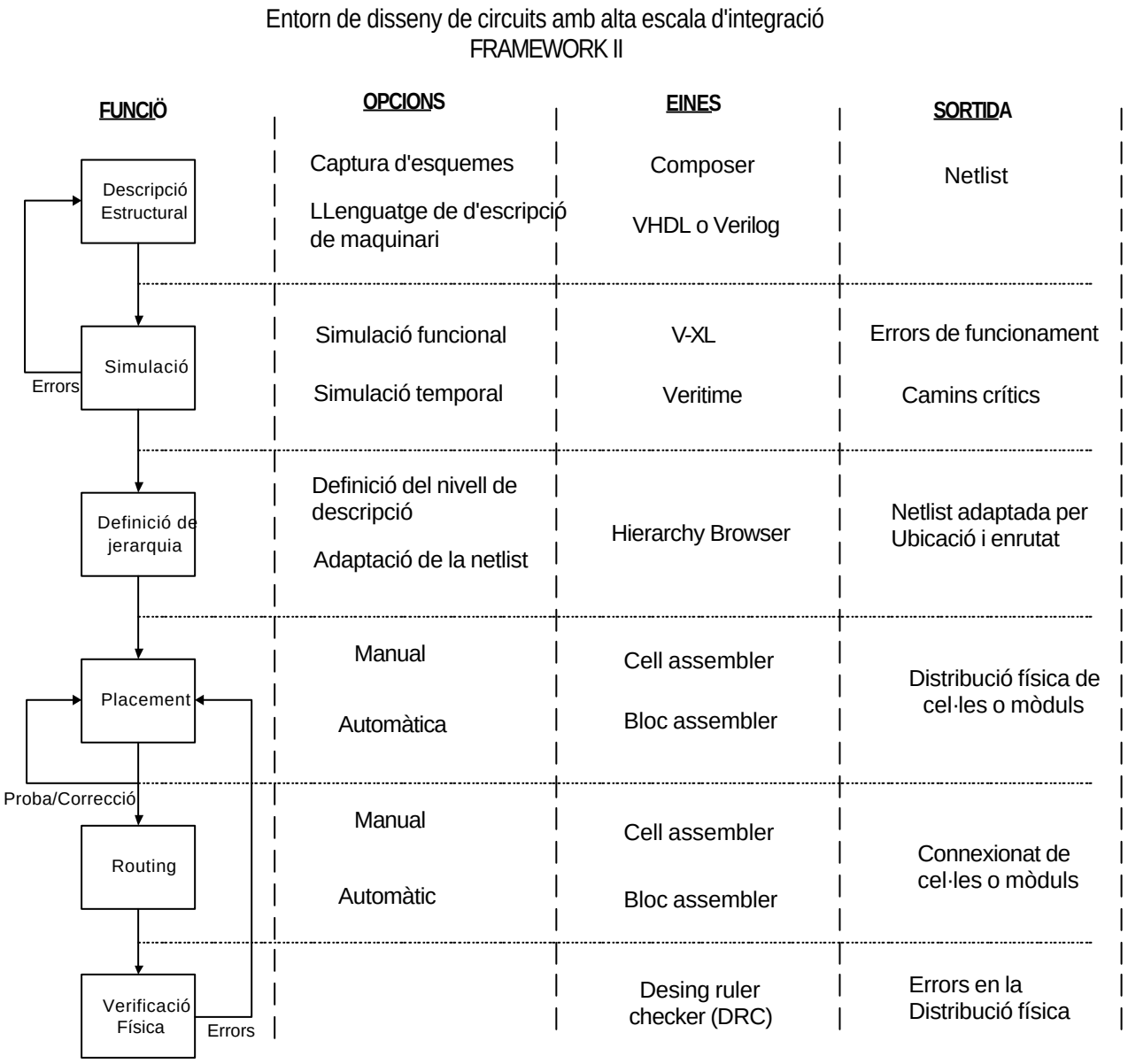
Es poden tenir ubicacions planes, on només hi ha cel·les estàndard, o ubicacions amb mòduls i submòduls amb un cert grau de jerarquia. En general, el disseny és més efectiu si el grau de jerarquia és menor. La raó d'aquest fet és que les eines automàtiques de ubicació i connexió obtenen un major grau d'eficiència per el fet de tenir un major grau de llibertat. Tot i així, ja que el nostre disseny està format per una distribució en files i columnes d'unitats ben definides, s'haurà de suportar un cert grau de jerarquia si volen que el resultat final mantingui la distribució.

Tal com acabem de dir, per completar el disseny és precisa una eina que distribueixi en el espai físic de silici disponible, els mòduls i les cel·les estàndard del circuit i que reservi l'espai per el canals de connexió. L'entorn disposa d'una eina de *Placement* que pot treballar a dos nivells. A nivell de cel·la estàndard (*Cell Assembler*) ubica directament cel·les estàndard. A nivell de bloc (*Bloc Assembler*) ubica mòduls de cel·les agrupades en processos anteriors. L'eina pot funcionar de forma automàtica, però també permet treballar de forma manual per donar la possibilitat al dissenyador d'augmentar l'eficiència. Com és sabut, les eines automàtiques d'ubicació física sovint no completen el procés o, en determinades circumstàncies, desaprofiten l'espai de forma que és precís un procés manual final de tal manera que el procés resulta ser semiautomàtic.

Un cop completat el procés de *Placement* cal connectar les cel·les i els mòduls del circuit entre si, a partir de la *Netlist* obtinguda en els processos inicials. Com hem dit anteriorment l'eina *Routing* és l'encarregada de fer aquesta tasca i de la mateixa manera que el cas anterior, el procés es pot fer a dos nivells connectant cel·les estàndard individuals (*Cell Assembler*) o mòduls de cel·les agrupades en processos de *Placement* i *Routing* anteriors. El procés de *Routing* tria automàticament els canals de connexió i genera les pistes a les capes de metall que materialitzen les connexions indicades a la *Netlist* citada anteriorment. També com en el cas anterior i per els mateixos motius l'eina dóna la possibilitat de connexió manual, convertint el que dóna procés en semiautomàtic.

Encara que pugui semblar sorprenent les eines automàtiques de *Placement* i *Routing* es salten alguns cops de forma aleatòria les regles de disseny i restriccions físiques que imposa la tecnologia. També en processos manuals es podem violar les restriccions. Per tant l'entorn disposa d'una eina de verificació (*Desing Ruler Checker*) que detecta aquest tipus d'errors. De forma que aquest pas final dóna lloc a un procés iteratiu de correcció verificació.

En la taula 6.1 es poden veure les diferents eines de l'entorn amb la seves funcions i opcions així com un diagrama resum del procés de disseny que hem utilitzat.



Taula 6.1: Entorn de disseny

6.2 Adaptació del disseny a la tecnologia d'alta escala de integració VLSI

La norma general sobre disseny de circuits *VLSI* obtinguda de l'experiència i de les restriccions de fabricació de les famílies tecnològiques, aconsellen implementar els circuits seqüencials de forma síncrona. Com que la nostre arquitectura ja disposa d'un senyal de rellotge que sincronitza el moment en que es carreguen els registres de les cel·les processadores de píxel, no hem de fer cap adaptació en aquest sentit. Només caldrà ajustar el període d'aquest senyal, a un temps mínim igual al temps de propagació del camí més llarg del circuit un cop acabat el disseny. Caldrà disposar també d'un senyal de *Reset* asíncron que generi la inicialització de tots els circuits

En el apartats següents mostrarem les modificacions de les diferents unitats que composen la nostra arquitectura per tal d'adaptar-les a la tecnologia *VLSI*. Discutirem les diferents opcions a l'hora d'implementar les funcions de les diferents unitats i presentarem les millors opcions per tal d'estalviar àrea de silici que, en alguns casos, pot suposar un sacrifici de velocitat.

6.2.1 Cel·la processadora de píxel

En aquest apartat mostrem les modificacions que cal fer a la cel·la processadora de píxel resultant del disseny realitzat en el capítol 4 i que es pot veure a la Fig.4.12, per tal de convertir-la en un disseny que permeti una implementació amb tecnologia d'alta escala d'integració *VLSI*.

Com a consideració general prèvia cal dir que alhora d'escollir les cel·les estàndard dins la llibreria que proporciona el fabricant per implementar els diferents components de la cel·la de píxel, s'han escollit les que tenien les entrades i sortides sense amplificador/regeneradors (*Buffers*: portes amplificadores de senyal amb baixa capacitat d'entrada i alta capacitat de sortida). Aquestes cel·les sense *Buffers* tenen una capacitat força elevada a les seves entrades, i suporten poca capacitat a les seves sortides, però s'han escollit perquè eren lleugerament més petites. Això no comporta cap tipus de problema en el cas que ens ocupa ja que, tret dels senyals de sortida de la cel·la que

tractarem de forma diferent, la resta de connexions són locals a la mateixa cel·la de píxel tal com es pot veure a la Fig. 4.12. (com és sabut la llargada de les connexions determina la magnitud de les capacitats paràsites de càrrega).

No apareixen tampoc problemes de *fan-out*, perquè no hi cap senyal local de sortida que es connecti a més de dues entrades. Tot i que l'estalvi a cada cel·la no és molt gran, aquesta decisió proporciona un estalvi global d'àrea molt considerable ja que, com hem dit, la cel·la processadora de píxel ha de ser replicada $N \times M$ cops.

Els senyals del bus de sortida de la cel·la són els únics que no són locals i per tan s'hauran d'adaptar per suportar capacitats de sortida importants a causa de la distància entre les cel·les i l'etapa de sortida (Fig. 4.14). Per altra banda, aquestes sortides han de ser de tres estats per tal de controlar la connexió o desconnexió de la cel·la al bus de sortida. Com es pot veure en el disseny original de la Fig. 4.12 aquests senyals surten a través de *Buffers* de tres estats. Això comporta una nova problemàtica generada per la pròpia tecnologia. Ja hem dit que aquest tipus de portes ocupen més espai que les portes normals, però el control de tres estat agreuja encara més el problema, tenint en compte que aquest augment d'àrea es multiplicarà per les $N \times M$ cel·les de l'arquitectura.

Per tal de millorar aquesta problemàtica proposem modificar la filosofia de funcionament del processador tenint en compte la tasca que realitza dins de l'algorisme de detecció de simetria. Recordem que el mapa d'acumulació que calcula l'arquitectura s'utilitza per tal d'acotar la llargada dels eixos locals virtuals de simetria. L'acotació s'aconsegueix comparen els valors d'acumulació, en el mapa, dels píxels que formen els eixos virtuals amb un llindar que es calcula en funció del pes de l'eix virtual que es tracta, tal com es va concloure en el apartat. 3.8.2. Si incloem aquesta comparació dins la cel·la aprofitant el comparador que ja existeix, obtenim com mostrarem tot seguit, una reducció d'àrea amb l'avantatge que hem descarregat l'algorisme de la comparació esmentada.

A la Fig 6.1 es mostra la cel·la de píxel modificada on hem afegit un multiplexor que tria entre la comparació dels increments de fila i columna, en el processament de segments per el càlcul del mapa d'acumulació, o la comparació del valor d'acumulació dels

registres de cel·la amb el llindar esmentat, en les operacions d'accés als resultats. Per el control del multiplexor utilitzem, lògicament, el senyal lectura *RD* que discrimina precisament entre les operacions de processament de segments i les d'accés. En aquest últim cas la comparació necessita de la sortida "més gran o igual ", per la qual cosa s'ha modificat el comparador. Com les dues comparacions són exclusives en el temps, utilitzarem les mateixes línies de sortida de les unitats de càlcul de columna, per tal de fer arribar el llindar del valor d'acumulació al les cel·les (Fig. 6.8). Aquesta modificació comporta lògicament afegir un nou multiplexor a les unitats de càlcul de columna, però això no representa un problema ja que aquestes unitats tenen baixa replicació.

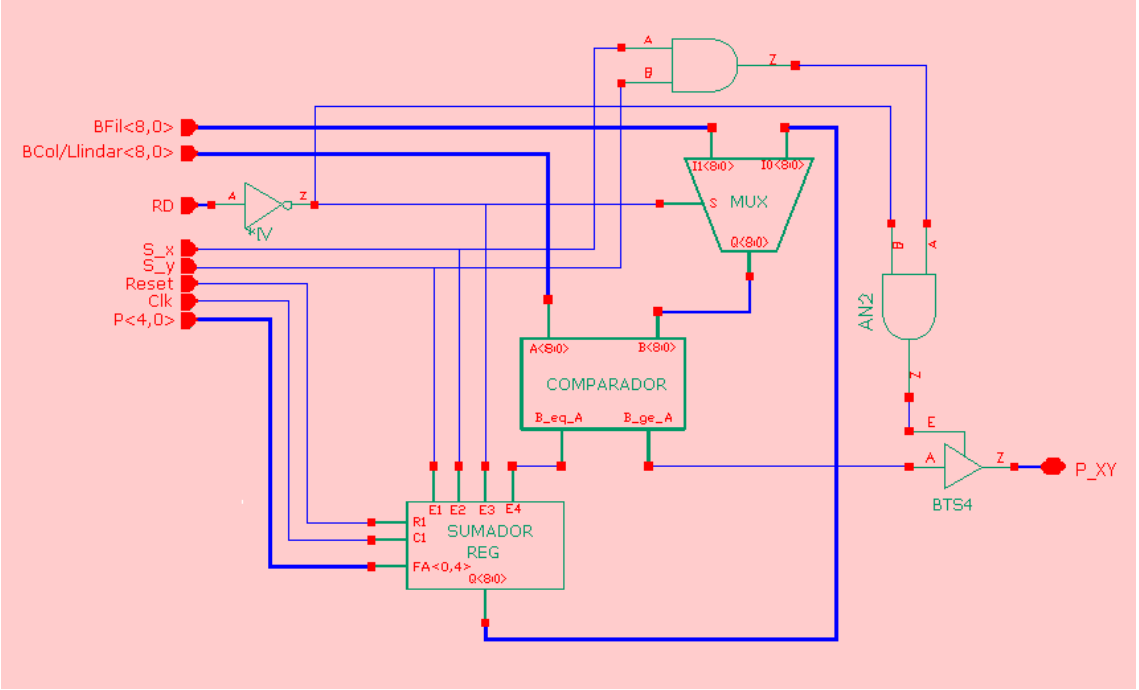


Figura 6.1: Cel· la de píxel modificada per una implementació VLSI

Amb aquestes modificacions hem substituït a cada cel·la 7 *Buffers* de sortida de tres estats, per un únic *buffer* i 8 multiplexors 2:1. Si bé en la tecnologia de què disposem aquest canvi dóna un balanç de millora d'àrea no molt significativa, si que obtenim una reducció global important; per un costat, per l'alta replicació de la cel·la dins l'arquitectura i per l'altra per el fet que el bus de sortida de la cel·la s'ha simplificat a un únic senyal, que indica si el valor d'acumulació de la cel·la supera un llindar especificat. Això significa una reducció de 7 línies de sortida a cada columna amb els respectius

separadors /amplificadors de tres estats de l'etapa de sortida del circuit, és a dir, $N \acute{} 7$ línies de bus i $N \acute{} 7$ separadors /amplificadors de tres estats (veure Fig. 4.14).

Respecte l' implementació dels diferents mòduls que composen la cel·la, cal dir que el multiplexor no és altra cosa que l'agrupació de 8 multiplexor 2:1 estàndards de la llibreria, mentre que el comparador s'ha implementat com un restador amb un detector de zero, on el cas "més gran o igual " és directament del senyal *carry*. Aquest disseny no és els més ràpid que es podria obtenir, però si que és l'òptim en àrea utilitzada. S'ha triat aquest model perquè aquesta unitat té una alta replicació dins l'arquitectura de manera que no es pot malgastar gens d'àrea, sacrificant si cal una mica de velocitat.

Respecte al disseny del registre de la cel·la processadora de píxel que memoritza el valor d'acumulació, s'ha construït a partir de l'agrupament de set registres estàndard tipus D d'un bit, amb senyal d'esborrat i que estan implementats en la pròpia llibreria.

Finalment, recordem que el mòdul sumador ha de sumar els 8 bits del valor acumulat en el registra de píxel amb els 4 bits de la ponderació del segment que s'està tractant, tal com es va descriure a l'hora de dissenyar la cel·la de píxel en el apartat 4.2.1. Aquí, aprofitarem el fet que una de les paraules a sumar té els quatre bits de la part alta a zero, per tal de simplificar els sumadors implicats. Obtenim així un reducció d'àrea que tot i que no és molt significativa a nivell de la cel·la individual, si que ho és, una vegada més, a nivell global, a causa de la replicació a que està sotmesa aquesta unitat dins de l'arquitectura. A la Fig. 6.2 mostrem que la reducció, a cada cel·la, és de 3 portes a cada un dels elements sumadors, per els quatre bits de més pes, conseqüentment la reducció global és de $12 \times N \times M$ portes. A títol d'exemple, per una imatge de 256×256 significa una reducció 786.432 portes.

Per altra banda, com es veurà en el apartat següent els senyals globals que arriben a totes les cel·les de píxel, com és el cas dels senyals de rellotge *Clk*, el *Reset* i el *RD* reben un tractament especial alhora de distribuir-les per la matriu de cel·les

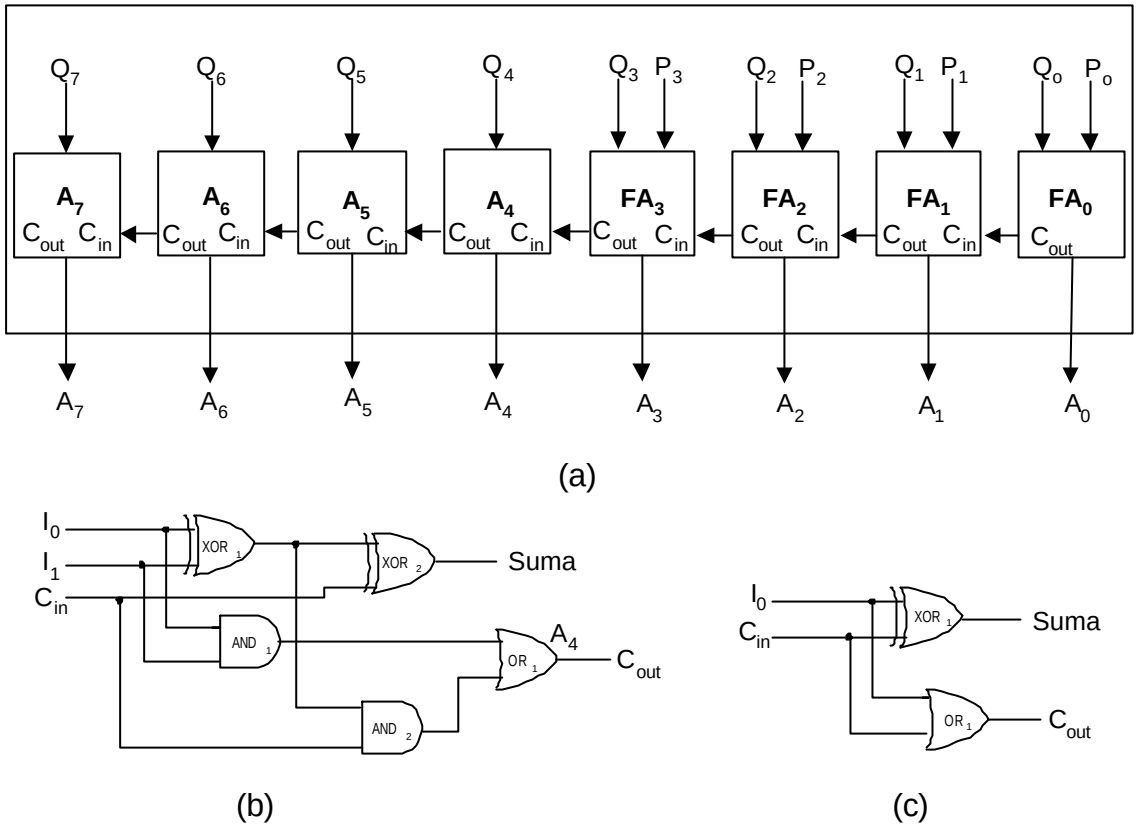


Fig. 6.2. Modul sumador. (a) Diagrama de blocs. (b) Element sumador complet. (c) Element sumador simplificat

processadores de píxel. No obstant, tal com es pot veure a la Fig 6.1 el senyal de *RD* té la problemàtica afegida que es distribueix dins la cel·la de píxel suportant un total de tres entrades. Això significa que aquest senyal està carregat per un total de $3 \times N \times M$ entrades. Per tal de facilitar la distribució d'aquest senyal global, reduint la seva càrrega, s'ha optat per separar-la amb una porta. S'ha utilitzat un inversor enlloc de separadors/amplificador, tal com es pot veure a la Fig 6.1. Les portes inversores ocupen molt menys silici que els *buffer*s. D'aquesta forma s'aconsegueix que la càrrega, per cada cel·la de píxel, sobre aquest senyal sigui només d'una porta i amb un cost de silici inferior.

La resta de senyals que arriben dels bussos no són senyals globals, ja que es generen en diferents unitats que suporten només la càrrega d'una fila o columna de cel·les. Aquest és el cas dels valors dels increments de fila i columna, que es generen a les unitats de càlcul, de les línies de descodificació de fila i columna (veure Fig. 4.14) i del llindar de

comparació que aprofita el mateix bus que l'increment de columna de la unitat de càlcul de columna. Així doncs, no s'ha fet cap tractament d'aquest senyals a la cel·la ja que,

com acabem de dir, les unitats que les generen disposaran, a les seves sortides, *buffer*s que poden suportar aquesta càrrega (veure Fig 6.8 i 6.9). Si disposéssim amplificadors en aquest senyals no aportaríem millores, però sí costos innecessaris d'àrea.

6.2.2 Representació física de la cel· la processadora de píxel

Un cop finalitzat el procés de disseny i validat per mitjà de simulacions, es passa a la generació dels processos de *Placement i Routing*, que ens proporciona una representació física de la cel·la (*Layout*) de la qual podrem obtenir-ne les seves dimensions

El procés de creació del *Placement* s'inicia normalment a partir de les descripcions esquemàtiques o de les descripcions funcionals generades amb llenguatges de descripció de hardware. L'eina permet decidir quins seran els submòduls en que partirem tot el circuit. Aquesta decisió repercuteix en l'aprofitament de l'àrea disponible. L'experiència demostra que una divisió en mòduls massa petits no proporciona un bon aprofitament de l'àrea disponible ja que no deixem un grau de llibertat suficient a l'eina per tal de distribuir les cel·les estàndard de forma eficient.

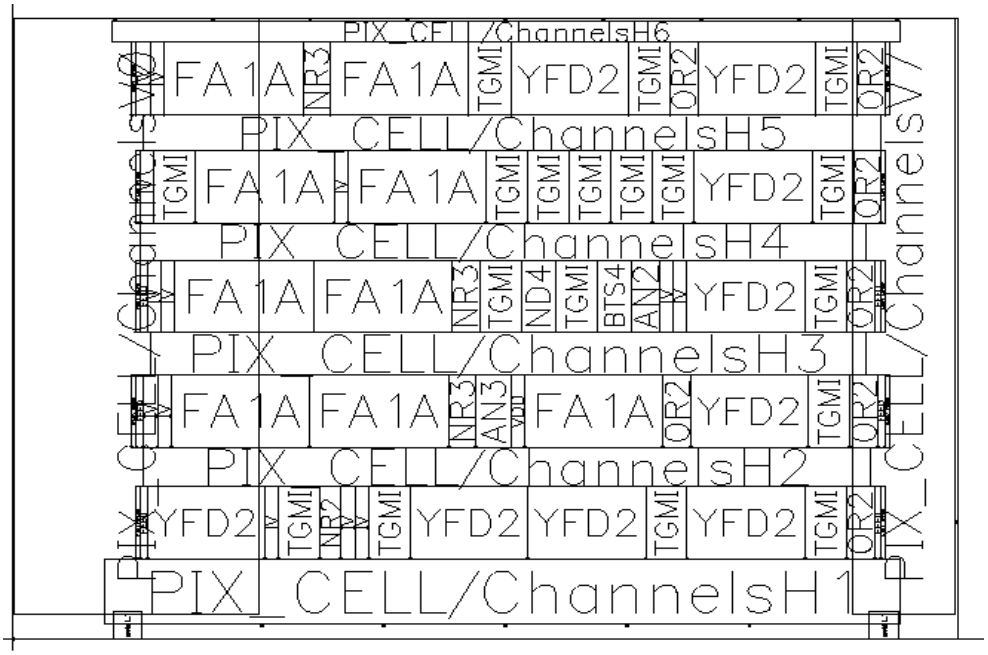


Figura 6.3: Distribució de les cel·les estàndard i els canal de connexió dins de la cel·la processadora de píxel.

Seguint aquest criteris s'ha decidit que la cel·la processadora de píxel serà un del mòduls en què es dividirà el circuit. Primer perquè està formada per un nombre prou elevat de cel·les estàndard i segon perquè és una unitat lògica del circuit que es replica molts cops amb una distribució estructurada

A la Fig 6.3 es pot veure com el procés *Placement* ha distribuït les cel·les estàndards en files de cel·les contigües i ha reservat de forma alternada els canals per on es faran les connexions. Algunes files contenen minúscules cel·les que en realitat són salts de canal per tal de facilitar l'interconnexió de cel·les estàndard situades en files diferents Finalment a la Fig 6.4 es pot veure com el procés *Routing* ha generat l'interconnexió de les cel·les a partir de les línies de metall disposades dins els canals de connexió reservats per aquest fi.

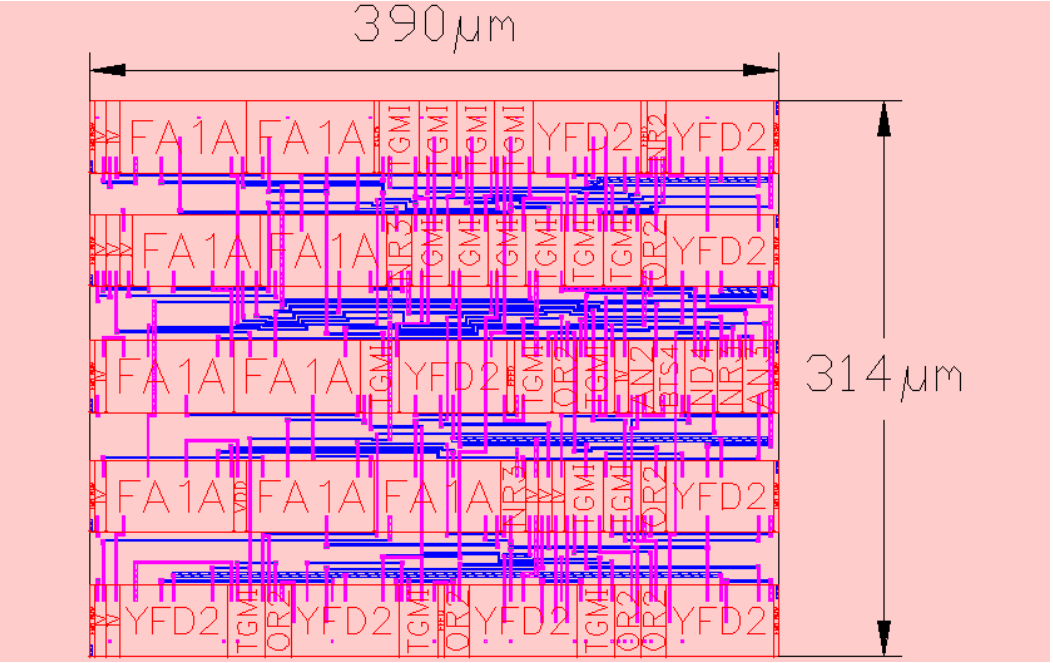


Figura 6.4: Representació física de la cel·la processadora amb les línies de metall

Com es pot veure a la Fig 6.4 les dimensions de la cel·la resultant d'aquest procés són de 390µm X 314µm que representa una àrea de 122.460µm^2. A títol orientatiu això significa que la matriu *32x32* que volem dissenyar tindria unes dimensions 32 cops més grans sense tenir en compte el silici necessari per els canals de connexió, és a dir, 1,24 cm X 1,00 cm. Aquestes dimensions són considerables però no excessives ja que la

matriu de cel·les processadores representen més del 95% de totes les unitats del circuit la qual cosa ens fa pensar que, tot i que desconeixem la incidència dels busos en l'ocupació, queda un marge suficient per encabir la resta d'unitats.

6.2.3 Matriu de cel·les processadores

Un cop ja s'ha obtingut la cel·la de píxel, el següent pas és l'obtenció de la matriu de cel·les processadores mitjançant la replica de *32x32* cel·les. Donada la distribució en files i columnes del nostre disseny, sembla obvi que una distribució física de les cel·les processadores de forma matricial, facilitarà la interconnexió dels diferents busos interns. Al mateix temps, aquesta distribució coincideix amb la distribució dels píxels a la imatge. Això representa una avantatge obvia, entre elles facilitar als algorismes l'accés a les dades. Segons això, és evident que la millor forma de situar els busos és aprofitant els espais que hi ha entre files i columnes (veure Fig 6.13).

És evident que els senyals que han d'arribar a totes les cel·les de píxel han de recorre molt espai dins del silici i suportar la càrrega d'un gran nombre d'entrades (*32x32)*. Les línies de metall tant llargues, tenen una gran capacitat que s'ha de sumar a la de les entrades d'un gran nombre de cel·les estàndard. Això vol dir que la porta que ha de generar el senyal ha de suportar una capacitat excessiva. Tot i així, s'han de distingir diferents casos més o menys crítics en funció de la càrrega que han de suportar els senyals.

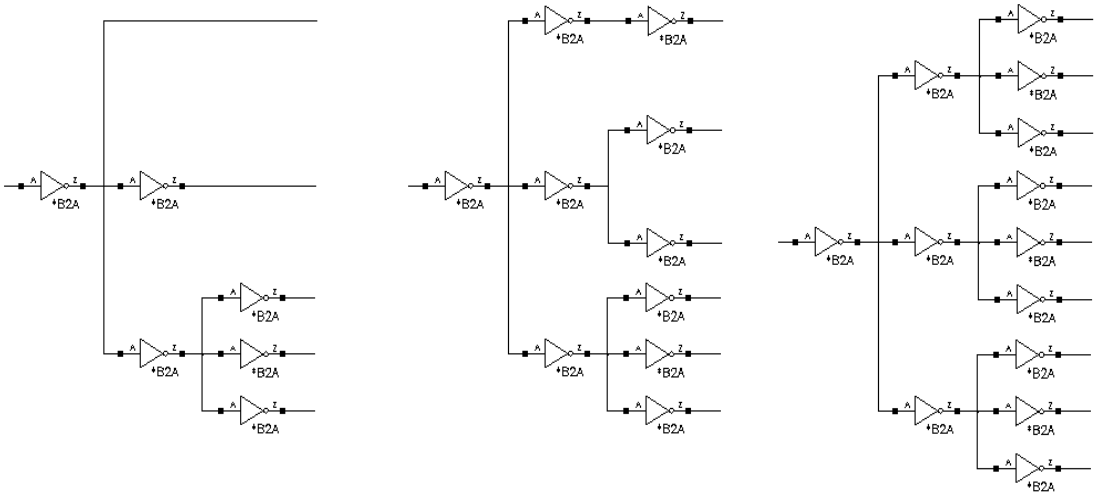
En el cas dels busos que porten els valors dels increments de fila i columna des de les unitats de càlcul fins a les cel·les de píxel, el problema del excés de carrega es fàcil de resoldre pel fet que aquest busos només han de suportar la carrega d'una fila o columna de cel·les de píxel (veure Fig 4.14). Segons això, la solució consisteix en afegir *buffers* suficientment grans a la sortida de les unitats de càlcul (veure Fig 6.8 i 6.9)

El cas del bus que porta el valor de ponderació fins les cel·les de píxel, és una mica més complex, per el fet que els senyals han d'arribar directament a totes les cel·les de píxel de la matriu. Aquí la solució adoptada consisteix en dividir el bus mitjançant *buffers* de forma que generin un sub-bus de columna per cada una de les *N* columnes de la matriu de cel·les de píxel. D'aquesta forma cada *buffer* només suporta l'entrada de les cel·les

d'una columna. Aquí el cost es més elevat que en el cas anterior, per el fet de tenir que generar una nova distribució de sub-busos de columna. Cal tenir en compte, però que en tractar-se d'un bus de 4 bits el cost addicional és de *4xN=4x32=128 buffers*.

El senyal *RD* que selecciona el mode d'operació presenta la mateixa problemàtica que les línies del bus anterior i, per tant, adoptem la mateixa solució que consisteix a disposar un *buffer* per cada columna de la matriu de cel·les processadores.

Finalment, hem de dir que les solucions adoptades fina ara funcionen bé pels senyals poc crítics, que es propaguen i acabant al seu destí si es dóna el temps suficient, tot i la dispersió de capacitats, (problema conegut a la bibliografia com "slew rate"). Però és problemàtic per el cas de senyals com el rellotge, el *Reset*. Aquests senyals a part del problema de la gran quantitat de cel·les a les que s'han de connectar, tenen la



(a) (b) (c)

Figura 6.5: Exemple de distribució en forma d'arbre. (a) Incorrecta per diferència de profunditat. (b) Incorrecta per desequilibri entre les branques (c) Correcta

problemàtica afegida que han d'arribar a totes les cel·les al mateix temps. Per tal d'assegurar un bon funcionament s'ha optat per una estratègia de distribució dels *buffers* en forma d'arbre. Totes les branques de l'arbre han d'estar equilibrades (cada nivell de cada branca ha de tenir el mateix nombre de separadors i totes tenen la mateixa profunditat, és a dir, totes les branques tenen el mateix nombre de nivells). A la Fig.6.5. mostrem una distribució en arbre correcta i els errors que es poden cometre en desequilibrar les branques o la profunditat

Un altre línia que té característiques especials és la sortida de les cel·les de píxel a través del seu *buffer* de tres estats, que es connecten amb la línia de columna que va a l'etapa de sortida del circuit (veure Fig 6.1 i 4.14). Aquí s'ha de garantir la connexió, a la línia de sortida, d'una única cel·la de píxel per cada columna, en cas contrari es pot crear un curt circuit que podria destruir els *buffers* de sortida de les cel·les de píxel. El nostre disseny té en compte aquesta problemàtica, doncs el control de connexió es realitza de forma centralitzada a partir dels descodificadors de fila i columna. Així per una adreça de cel·la donada, només s'activa una sola línia de selecció de fila i columna respectivament. Per aquest fet no pot produir-se una doble connexió sobre una mateixa línia de sortida.

Per altra banda, a causa de la pròpia tecnologia, no es poden deixar les línies de bus al aire perquè es creen corrents estàtics que són perjudicials. Per resoldre aquest problema s'ha afegit un *buffer* de tres estats a cada bus, que fixa un zero lògic si no hi ha cap unitat connectada al bus.

6.2.4 Unitats de càlcul

El disseny genèric de les unitats de càlcul que calculen els increments de fila i columna que es van obtenir en el apartat 4.4.2 i 4.4.3 s'han d'adaptar els requeriments d'un disseny *VLSI*, tal com també s'ha fet en cas de la cel·la processadora de píxel. Com es va mostrar en el disseny genèric hi ha una unitat de càlcul per cada fila (FD_i) i una per cada columna (CD_i) de la matriu de cel·les de píxel (veure Fig.4.11). Es composen d'un restador, un multiplexor, i una taula LUT (veure Fig.4.9(b) i 4.10(b)).
L'arquitectura també incorpora dos unitats de càlcul comunes a tota la matriu (veure Fig.4.11), una per totes les files (FC) i una per totes les columnes (CC), constituï des totes dues per una taula LUT, un multiplexor (veure Fig.4.9 (a) i 4.10 (a)).

Els multiplexors que es fan servir a totes les unitats de càlcul són iguals i s'han dissenyat de la mateixa forma que els de les cel·les processadores de píxels.
El restador s'ha dissenyat de la mateixa forma que el del comparador de la cel·la processadora de píxel, és a dir amb propagació de *carry* (sacrifiquem temps a canvi d'àrea) però sense el detector de zero. Cal dir que tot i que l'estructura dels restadors és

igual per totes les unitats de càlcul, el fet que un dels operands de la resta és l'índex de fila *i* o columna concreta *j* que tracta la unitat (veure Fig 4.9(b) i 4.10(b)), fa que el disseny de les unitats de càlcul sigui individual per cada fila i columna de la matriu.

L'últim dels elements clau de les unitats de càlcul són les taules *LUT* que implementen els càlculs dedicats $Round(m*i)$ per la columna i, $Round((j/m)$ per la fila j, i els càlculs comuns de columna $Round(m*x_1)$ i de fila $Round(y_1/m)$ (Fig 4.9 i 4.10). Totes aquestes expressions tenen una problemàtica comuna que consisteix en el fet que el pendent de les rectes *m* no està acotada i té una variació no lineal de tal manera que a increments significatius en el valor de *m* li corresponen canvis d'angle tan petits que no es poden representar en el pla discret. Això fa que no sigui possible una

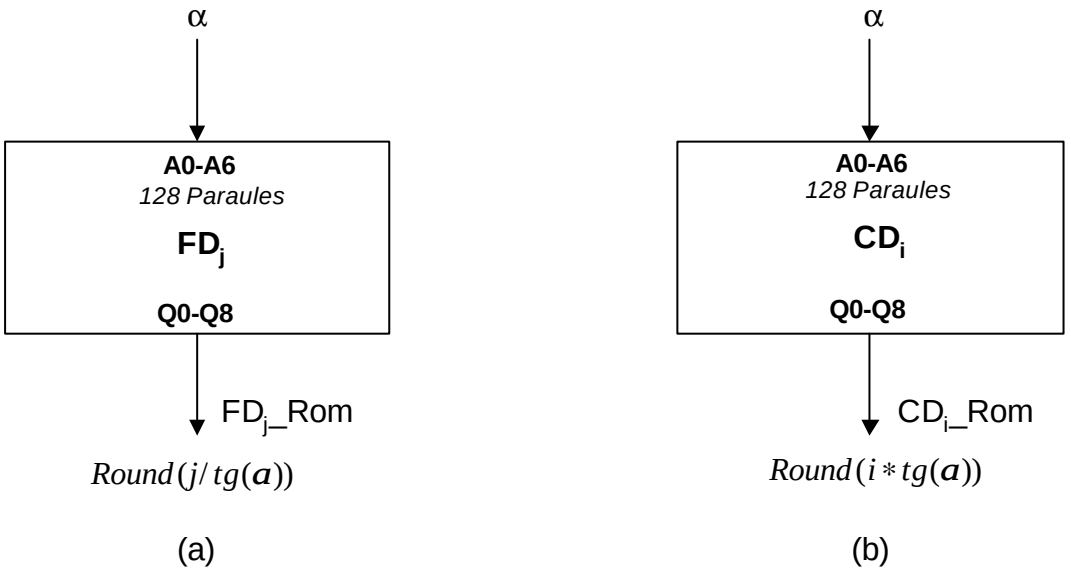


Fig.6.6 memories de càlcul dedicades. (a) De fila FD_j. (b) De columna CD_i

implementació de les taules *LUT* a partir d'aquestes expressions. Per tal de resoldre aquestes problemàtiques utilitzarem l'angle de la recta α per expressar el pendent *m* a partir de la transformació $m = tg(\alpha)$. D'aquesta forma les memòries que implementen les taules *LUT* tindran com entrada (és adir, com adreça de la memòria) l'angle de la recta a tractar (α), el qual té un variació lineal i un valor acotat de pendents possibles, expressades en angle, en el pla discret considerat .

Si considerem el píxel com l'increment d'angle mínim visible tindrem $M+N$ pendent per els angles compresos dins de l'interval [0, 90] graus i $M+N$ pendents més per els

angles compresos dins de l'interval [-90, 0] que dóna un total de $2*(M+N)$ pendents en funció de la resolució de la imatge. Així, en el nostre cas que volem obtenir una matriu quadrada de *32X32* píxels tindrem $4\times(32+32)=128$ angles.

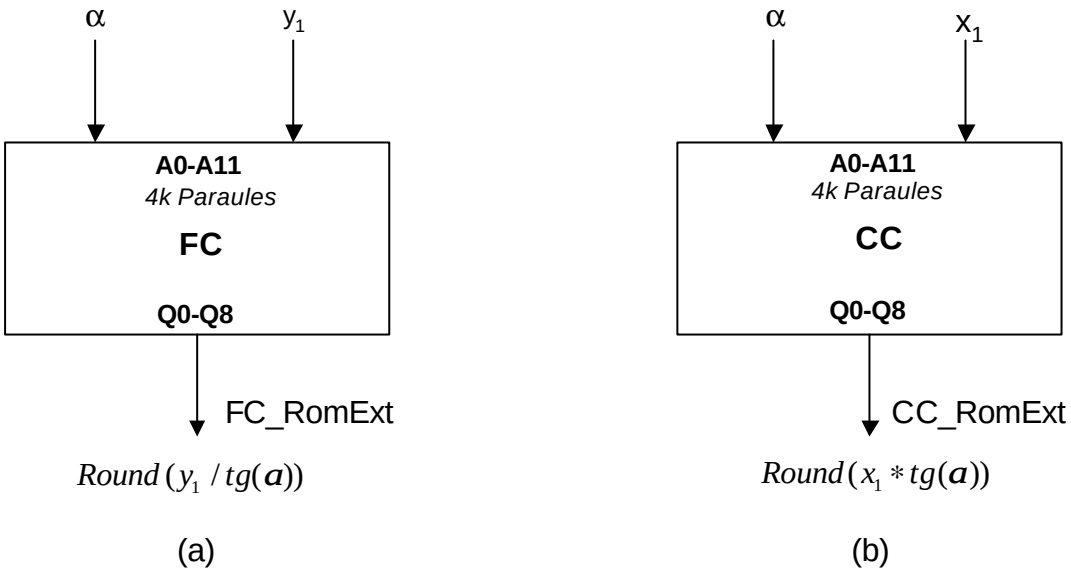


Fig.6.7 memories de càlcul comunes. (a) De fila FD_j. (b) De columna CD_i

Les taules *LUT* de càlcul de la fila *j* que calcula l'expressió ($Round((j/tg(\alpha))$) i de la columna *i* que calcula l'expressió $Round(tg(\alpha)*i)$ tindran com a única variable d'entrada l'angle (α) (Fig 6.6). Per tant hauran de tenir 128 paraules, fet que representa una capacitat suficientment petita per permetre la replicació de les $M+N$ unitats de càlcul necessàries. Les 128 paraules es codifiquen amb 7 bits d'adreça (A_0-A_6) on el bit més significatiu s'utilitza per codificar el signe de l'angle, mentre que els 6 bits restants codifiquen el valor de l'angle com es pot veure a la Fig. 6.6. Així, de la paraula 0 fins a la 31 hi ha els angles compresos entre 0 i 45 graus i de la paraula 32 fins a la 63 hi ha els angles entre 45 i 90 graus. Les 64 paraules restants de la part alta repeteixen la distribució d'angles però per a l'angle amb signe negatiu. Les paraules 31 i 32 corresponen ambdues a un angle de 45 graus. Això s'ha fet perquè les *LUT* quedin simètriques i en un futur poder aprofitar la propietat trigonomètrica de les tangents expressada per $tg(\alpha)=tg(90-\alpha)$ i poder-les fer més petites.

La codificació del signe de l'angle amb un bit també permetrà en el futur fer les taules LUT més petites, aprofitant la propietat trigonomètrica expressada per $tg(\alpha)=tg(-\alpha)$ a través d'una petita modificació del hardware. Amb les dos simplificacions descrites es disminuirà la capacitat de les memòries tres quartes parts.

Contràriament, les taules LUT comunes de càlculs de fila i columna que calculen les expressions $Round(y_1 / tg(\alpha))$ i $Round(tg(\alpha) * x_1)$ no s'han de replicar doncs són comunes per totes les files o columnes de la matriu. Per contra tenen com a variables d'entrada l'angle (α) i la coordenada inicial x_1 o y_1. La primera es codifica amb 7 bits i la segona per una matriu de *32X32* es codifica amb 5 bits per la qual cosa es precisa de

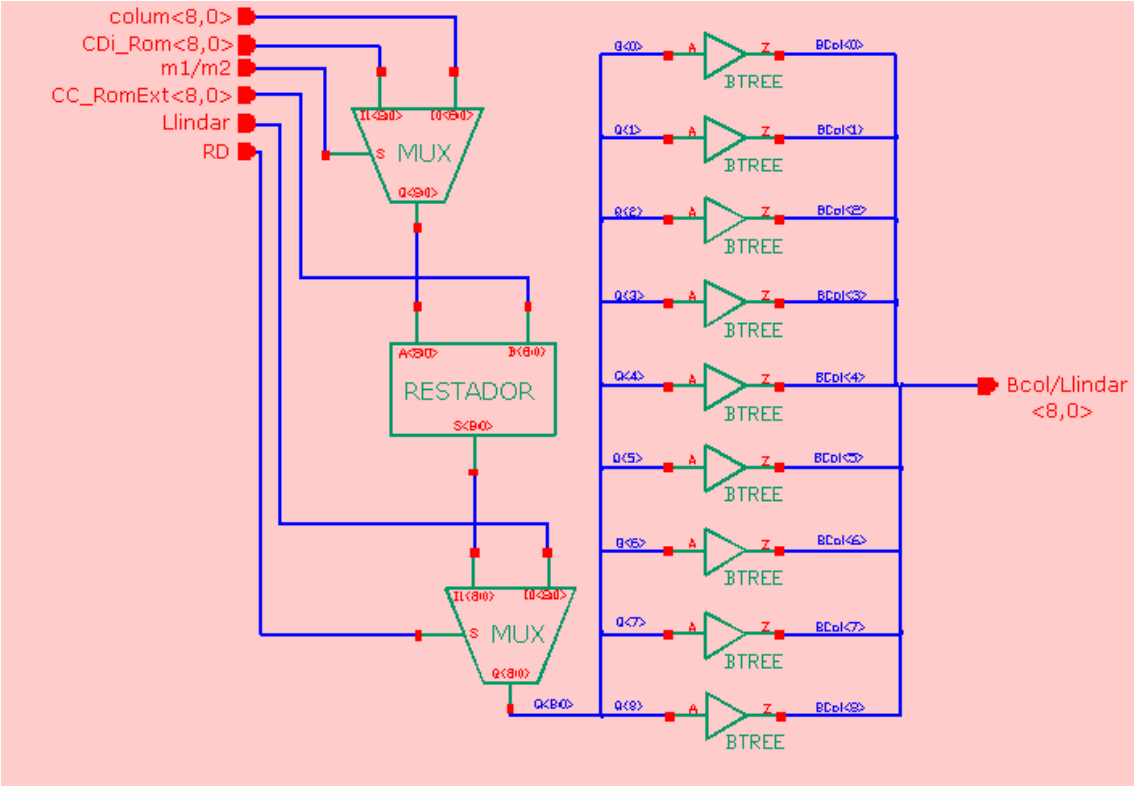


Figura 6.8: Esquema de la unitat de càlcul de columna obtinguda per l'eina de captura

12 bits que representen 2^{12} posicions, és a dir, 4K paraules tal com es mostra a la Fig. 6.7. Tenint en compte que en futures implementacions, de més alta resolució, aquesta memòria creix considerablement i pot dificultar l'obtenció del circuit, decidim extreure aquestes dues memòries ROM al exterior del circuit i disposar dues noves entrades al circuit *FC_RomExt* i *CC_RomExt* tal com mostrem a la Fig. 6.8 i 6.9. Cal afegir que tan les memòries dedicades com les comunes s'han implementat amb un bus de sortida sobredimensionat de 9 bits pensant en futures implementacions (per el nostre cas que considera una matriu de 32X32, 5 bits de sortida serien suficients), ja que el cost de silici addicional és petit.

A nivell d'implementació les taules Lut són memòries només d'escriptura no volàtils (*ROM*) que es poden obtenir de forma molt òptima si es fa a nivell de transistors, per això la majoria de fabricants permeten dissenyar aquests blocs com a capses negres a partir d'unes plantilles pre-dissenyades. El problema ha sorgit al no disposar de cap tecnologia que permetés mitjançant les eines al nostre abast, crear memòries simples sense sortida de tres estats. Per aquesta raó, les memòries que s'han fet servir aquí tenen

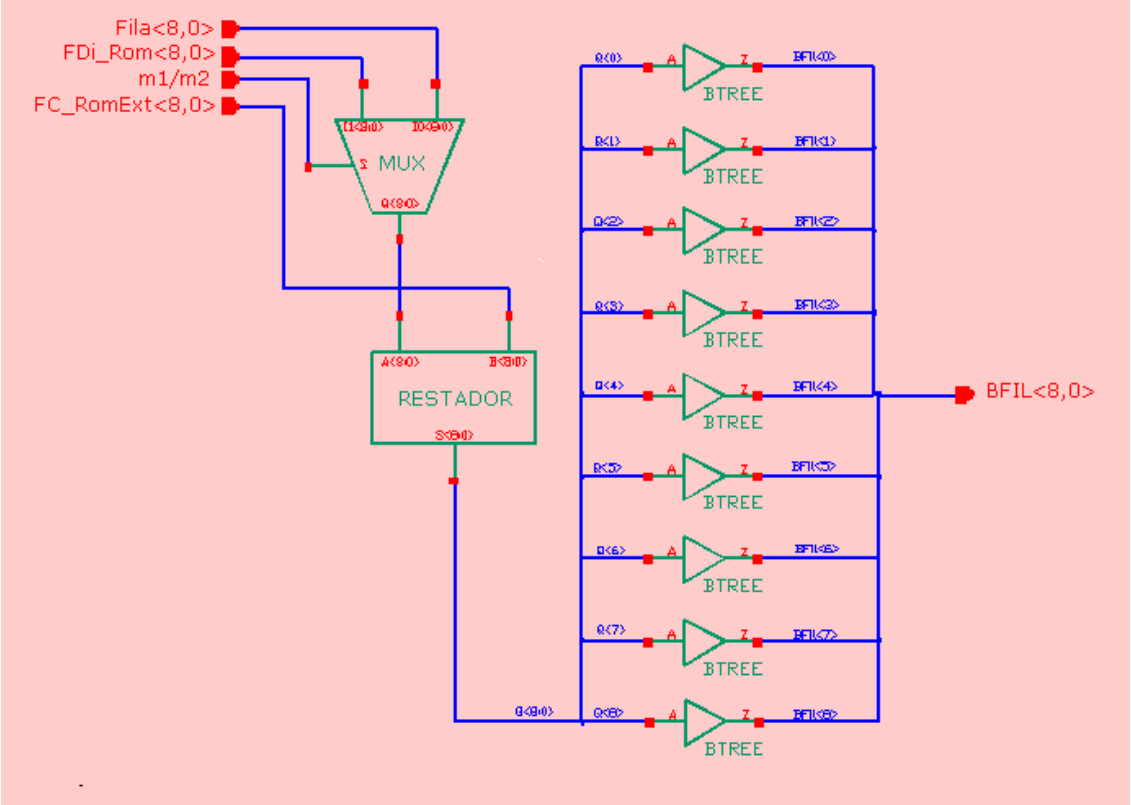


Figura 6.9: Esquema de la unitat de càlcul de fila obtinguda per l'eina de captura

aquest tipus de sortida tot i que en el cas particular que aquí tractem no es necessari i fa que malgastem una mica d'àrea. El senyal de control de connexió/desconnexió de la sortida s'ha fixat de forma que les memòries mai estigui en alta impedància.

A la Fig.6.8 mostrem l'esquema d'una unitat de càlcul de columna obtinguda amb l'eina de captura d'esquemes on s'ha exclòs la taula *LUT* ja que, com hem dit anteriorment, es dissenyen a part. Fem notar que hem separat cada un dels bits del resultat amb un *buffer* de dos estats, ja que aquestes línies han de suportar la càrrega de totes les cel·les de píxel de la columna que tractem. Cal recorda que aquesta unitat, a diferencia de les unitats de fila, disposa d'un multiplexor addicional, controlat per el senyal de lectura *RD*. Aquest multiplexor permet triar, en el temps, les dues operacions mútuament

exclusives: el processament de segments rectilinis per el càlcul del mapa d'acumulació i l'extracció de resultats a partir de la comparació del llindar d'acumulació, tal com s'ha desenvolupat a l'apartat 6.2.1. De la mateixa manera, a la Fig.6.9 mostrem l'esquema d'una unitat de càlcul de fila obtinguda amb l'eina de captura d'esquemes on, per el mateixos motiu anterior, s'ha exclòs la taula *LUT* i s'han separat els bits de sortida amb *buffers*.

6.2.5 Representació física de les unitats de càlcul

A la Fig.6.10 es pot veure la representació física de la unitat de càlcul de columna en cel·les estàndard, obtinguda en el procés de *Placement*, i les línies metàl·liques de connexió ubicades en els canals de connexió disposat per aquest, obtingudes en el procés de Routing. Per altra banda, fem notar que les dimensions d'aquesta unitat (*328,8mm X 277,8 mm*) dóna un àrea de *91340.6mm*^{2} lleugerament inferior a la de la cel·la de píxel amb l'avantatge que aquí la replicació és considerablement més baixa i igual al nombre de columnes N de l'arquitectura en funció de la resolució que tractem. Concretament en el nostre cas de resolució 32×32, les unitats de càlcul de columna

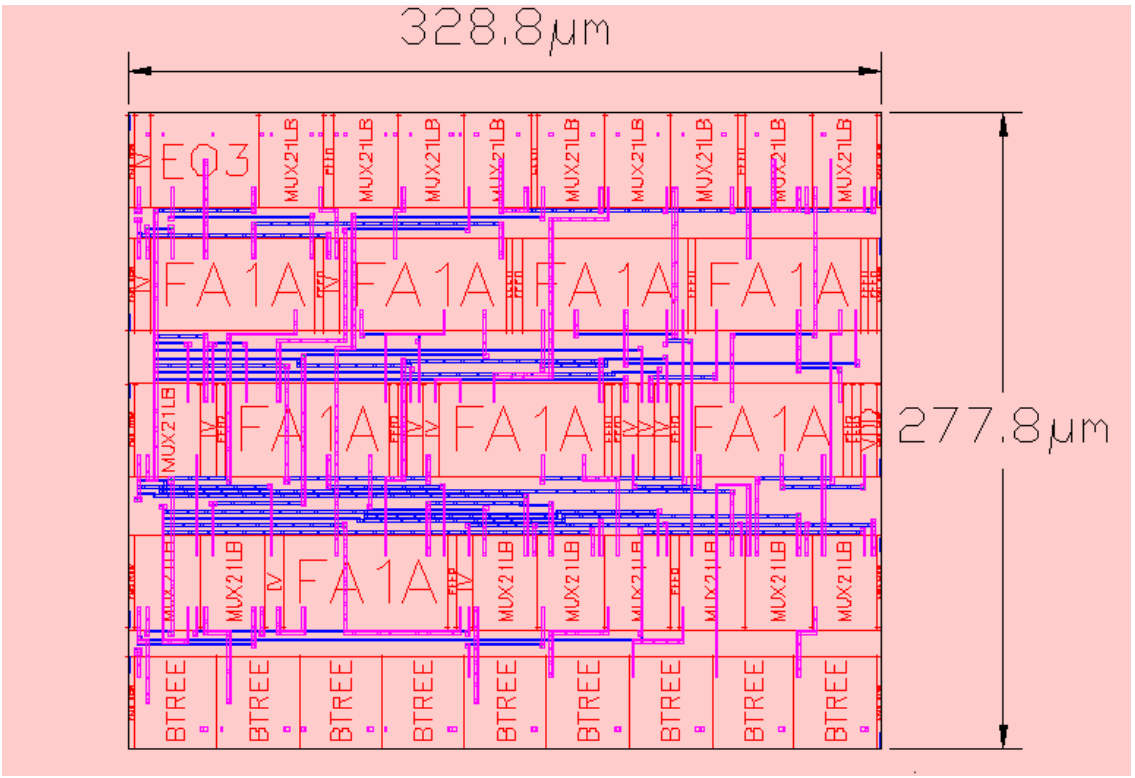


Figura 6.10: representació física de la unitat de càlcul de columna

ocuparan $10521{,}6\,mm \times 277{,}8mm$. En aquest càlcul s'ha tingut en compte que la distribució d'aquestes unitats ha de ser horitzontal segons la distribució de les cel·les processadores en files i columnes. Això s'ha decidit així per tal de facilitar la interconnexió i distribució dels busos a la matriu, tal com es va exposar a l'apartat 2.2.3. (veure Fig .6.13). Aquestes dimensions que representen una àrea de *2,9* mm^2 són poc significatives respecta l'àrea utilitzada per la matriu de cel·les (*1,24* cm^2 segons l'apartat 6.2) ja que representa al 2,3% d'aquesta.

Finalment, a la Fig.6.11 es pot veure la representació física (*Layout*) de la unitat de càlcul de fila amb les cel·les estàndard i les línies de connexió, obtinguda en el procés de *Placement* i Routing. Podem observar que les dimensions *264mm X 256,2mm* són lleugerament inferiors a la unitat de càlcul de columna per el fet que aquestes últimes disposen d'un multiplexor addicional que selecciona si es fa arribar a la cel·la processadora el llindar d'acumulació o el resultat de la unitat de càlcul. Aquí, l'àrea utilitzada és de *67636,8mm*^{2}, és a dir, *13703* mm^2 més petita que en el cas anterior. La replicació també és baixa i igual al nombre de files M de forma que en el cas que ens

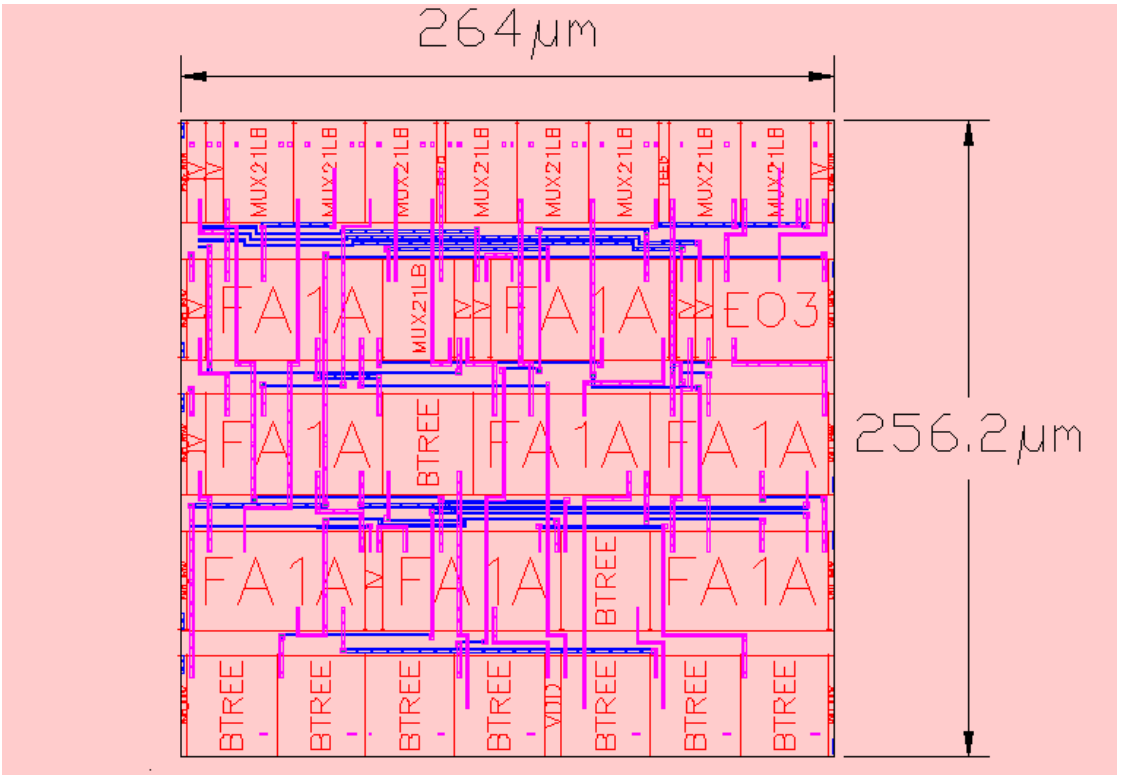


Figura 6.11: Representació física de la unitat de càlcul de fila

ocupa, de resolució 32x32 l'ocupació total serà de *264mm X 8198.4mm* tenint en compte que aquí la distribució ha de ser vertical. Aquesta ocupació representa un 1,7% respecta el que ocupa la matriu de cel·les processadores i per tant com en el cas anterior és poc significativa

6.2.6 Descodificador/acotador de fila o columna

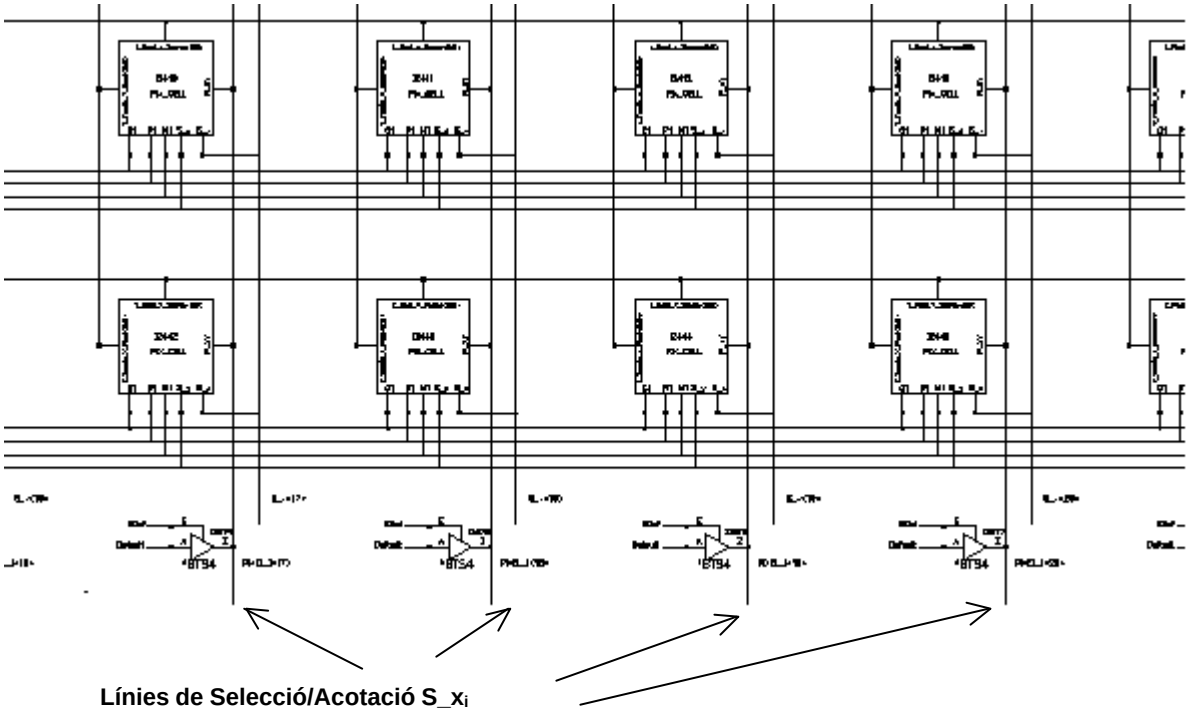
Com ja s'ha explicat en el capítol 4 aquestes unitats comparteixen en el temps dos funcions diferenciades. Per un costat, durant el processament de segments rectilinis, per tal d'obtenir el mapa d'acumulació, acoten les rectes a tractar a partir dels punts inicial i final del segment especificat, inhibint el procés a les cel·les de píxel que es troben per sota i per sobre d'aquests punts, segons es va descriure a l'apartat 4.2.7 i que es pot confirmar a l'esquema de la cel·la de píxel Fig.6.1. Per l'altra costat, en el procés d'accés aleatori als resultats emmagatzemats a les cel·les de pixels, aquestes unitats habiliten les línies de selecció de fila (*S_x*) i de columna (*S_y*) de la matriu de cel·les processadores obrint d'aquesta forma la sortida, tal com es pot veure a l'esquema de la cel·la processadora de píxel Fig.6.1.

Donada l'exclusió mútua de les dues tasques anteriors, tal com s'ha dit en l'apartat 4.2.7, aquestes dos funcions comparteixen les línies de sortida *S_x, S_y* de les unitats que aquí tractem, mentre que la selecció de la tasca es controla a partir de la senyal de lectura *RD* (veure taula 4.1)

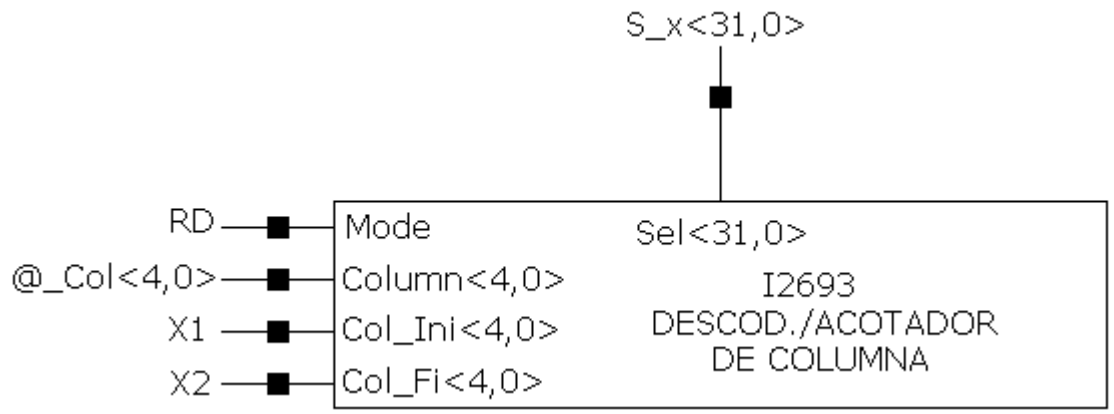
Es tracta doncs de dissenyar un mòdul amb dos unitats que treballen de forma exclusiva en el temps, la que realitza la tasca d'acotació que hem anomenat *Acotador,* i la que realitza la tasca d'accés aleatori, que hem anomenat *Descodificador*. Des de un punt de vista d'implementació, les dues unitats consisteixen en un conjunt de funcions combinatòries (una per cada fila *M* o columna *N*) d'una única sortida, però que comparteixen les mateixes entrades. En el cas de *l'acotador* les entrades comunes a totes les funcions són el punt inicial i final del segment a processar (X_1, Y_1, X_2, Y_2) mentre que per la unitat de descodificació són l'adreça de la fila o columna de la cel·la de píxel a la que es vol a accedir (veure Fig 4.14).

La Fig 6.12 mostra la configuració d'entrada/sortida del mòdul Descodificador/Acotador de columna i la seva relació amb la matriu de cel· les processadores.

El mòdul de fila és idèntic, la diferència rau en que a les entrades presentem la coordenada inicial i final de fila Y_1, Y_2 i l'adreça de fila @_fila, enlloc de la coordenada inicial i final de columna *X1*, *X2* i l'adreça de columna @_col, respectivament. A la sortida obtenim les línies de selecció/acotació de fila S_xenlloc de les de columna S_y.



(a)



(b)

Figura 6.12: Descodificador/Acotador de columna.(a) Connexió a la matriu de cel· les processadores (b). Entrada/Sortida del mòdul

La implementació independent de cada una d'aquestes funcions no és una solució òptima, ja que al compartir les mateixes entrades es poden trobar sub-funcions comunes entre elles, eliminant d'aquesta forma les redundàncies. El problema d'una implementació global radica en la complicació que suposa trobar factors que puguin compartir dues o més d'aquestes funcions independents. Per solucionar aquesta problemàtica, s'ha optat per sintetitzar-les a partir d'una descripció feta amb llenguatge de descripció de hardware, concretament amb el llenguatge *Verilog*.

Les eines que fan aquest tipus de síntesis, a partir d'un llenguatge de descripció de hardware, són molt hàbils a l'hora de simplificar aquest tipus de funcions. Poden trobar factors comuns i fer que es comparteixin amb algorismes heurístics força complicats a causa de la gran quantitat de variables que hi ha. Permeten variar el funcionament

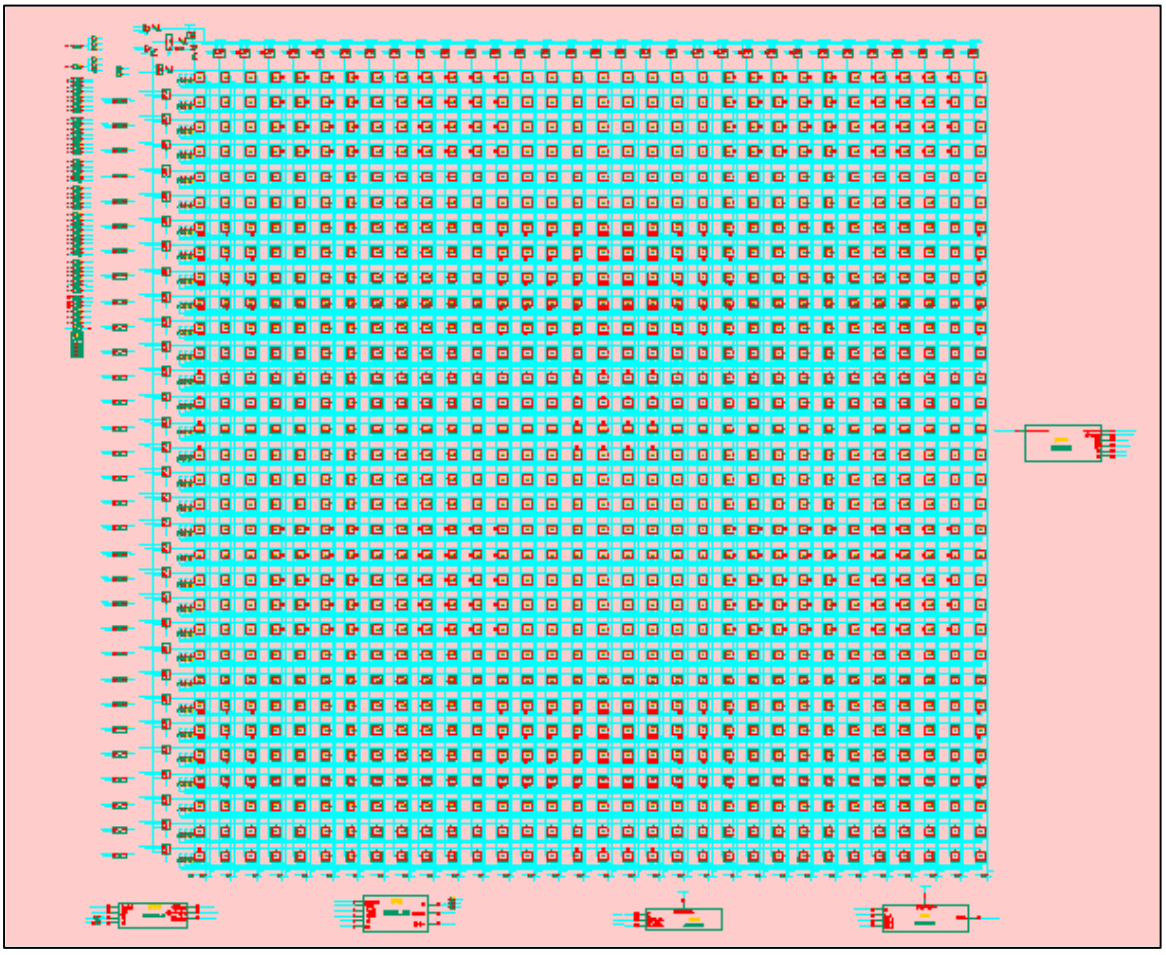


Figura 6.13: Visió esquemàtica del circuit dissenyat

d'aquests algorismes per mitjà d'alguns paràmetres. Donen opcions al dissenyador, tal com decidir si es vol un circuit optimitzat en àrea o en velocitat, quina ha de ser la càrrega màxima que poden tenir les entrades o quina ha de ser la càrrega mínima que han d'oferir les sortides. Un cop decidit això, el software intentarà implementar un circuit que s'ajusti el màxim a les especificacions. En el nostre cas la restricció més important és d'àrea utilitzada, per això s'ha començat sintetitzant el mòdul amb un àrea mínima. El resultat ha estat molt dolent en quant a la velocitat de funcionament, com ja era d'esperar. La solució final ha consistit en fer iteracions successives fins a trobar un circuit amb una relació entre el cost en l'àrea i la velocitat de funcionament, satisfactòria.

No presentem aquí una vista de la representació física del la unitat Descodificador/acotador ja que la forma d'implementació automàtica de les funcions combinatòries fa que aquesta sigui absolutament ininteł ligible i per tant irrellevant. Per altra banda, com es pot suposar, l'àrea de silici ocupada per aquestes unitats és molt petita respecta el circuit total. Concretament cada unitat és lleugerament més gran que una simple ceł la de píxel i representen aproximadament el 0.15 % de l'ocupació de la matriu de ceł les de píxel

6.2.7 Etapa d'Entrada/sortida

Les modificacions introduï des a la ceł la de píxel a l'apartat 6.2.1, per tal d'adaptar-la a un disseny VLSI, han canviat la forma d'obtenir el resultats del mapa d'acumulació, reduint la sortida de les ceł les a una única línia d'un bit que indica si el valor d'acumulació supera un determinat llindar. Donada la distribució matricial de les ceł les de píxel escollida, tindrem una línia de sortida per cada columna (veure Fig 4.14). Així, l'etapa de sortida consistirà en *32 buffer*s de tres estats controlats per el senyal de descodificació de columna i el senyal de control de lectura RD mentre que al disseny original precisava *32X 7* amplificadors.

Respecta l'etapa d'entrada no hi ha canvis en el disseny original de la Fig. 4.14, està constituï da per tants *buffer*s de tres estats com entrades al circuit. El control s'efectua també per el senyal de control *RD* però de forma complementaria, és a dir, quan el

procés no és de lectura *RD=0*. Segons això tindrem *32 buffers* per les entrades de punt inicial i final x_1, y_1, x_2, y_2 *i 4 buffer*s per la ponderació *P*.

6.3 Caracteristiques del circuit obtingut

Un cop dissenyats tots els mòduls que composen el nostre circuit obtenim un circuit global del qual ens interessen fonamentalment dos paràmetres: l'àrea de silici requerida i la freqüència màxima de treball. El primer s'obté directament de la representació física, es ha dir de dels processos de *Placement* i *Routing* de tots els mòduls que composen el circuit i que hem descrit al presentar l'entorn de treball en el apartat 6.1. Aquest resultat ens permetrà saber si l'arquitectura que hem dissenyat es pot ubicar en un sol circuit (encapsulat). El segon paràmetre l'obtindrem de l'anàlisi del camins crítics.

A la Fig.6.13 mostrem una representació esquemàtica del circuit obtingut de la captura d'esquemes de l'entorn de desenvolupament.

6.3.1 Àrea de silici requerida

Tal com acabem de dir dels processos de *Placement* i *Routing*, un cop corregits els errors detectats en el procés de verificació física, s'obté una representació global del circuit que proporciona les dimensions totals de silici necessari. A la Fig. 6.14 mostrem un tall d'aquesta representació. La imatge real no es pot presentar aquí, ni reduint-ne l'escala ja que aquest tipus d'imatges estan generades per ser explorades de forma assistida en un monitor i les dimensions són excessives per ser representades en un paper de dimensions normals. Les dimensions resultants tot i estar situades en el extrem del que és permissible, es a dir, una àrea pròxima a un quadrat *2x2*cm abonen la viabilitat del disseny, doncs no estem treballant amb una de les tecnologies de més alta escala de integració. En aquest sentit, explorarem més endavant l'ordre de magnitud de les millores que s'obtenen de la migració cap una tecnologia amb més alta escala d'integració de la que hem disposat, gràcies a una actualització de les llibreries de l'entorn durant el procés de disseny tal com mostrarem en el apartat 6.3.3

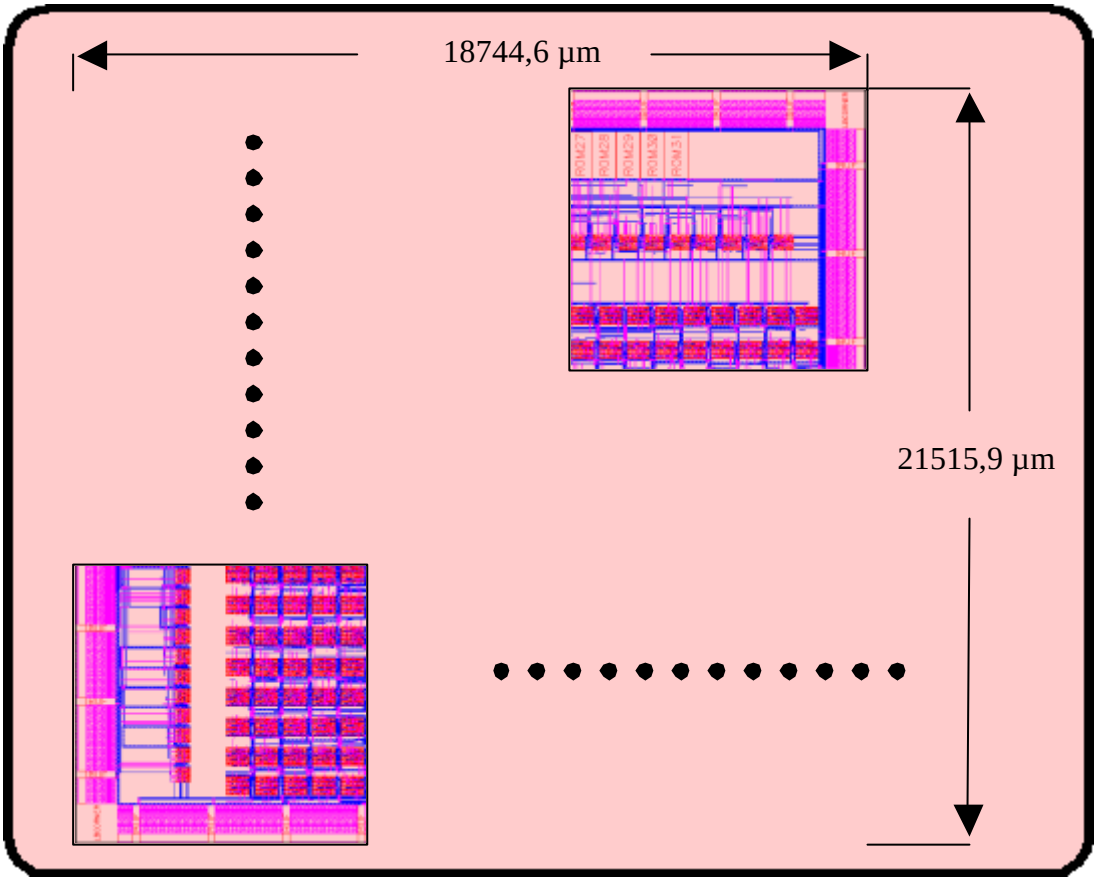


Figura 6.14: Tall del la representació física i dimensions del circuit amb tecnologia de 0.7 µm.

Per altra banda, cal fer notar que a part de la millora anterior queda la possibilitat de millorar l'àrea ocupada sacrificant el grau de paral·lelisme. Si enlloc de fer treballar totes les cel·les de píxel amb un paral·lelisme total, fem treballar, successivament en paral·lel, una sola fila de la matriu de cel·les, es podria reduir considerablement l'àrea de silici necessària, acceptant que la seqüencialitat introduï da comporta un sacrifici en temps de processament i una millora per el que fa a l'àrea de silici requerit. Els resultats en la freqüència màxima de treball que presentarem en el apartat següent diran si aquesta possibilitat és factible i podem pensar en implementacions que suportin resolucions superiors com ara *128X128* o *256X256* amb un temps de processament adequats.

6.3.2 Freqüència màxima de treball i capacitat de càlcul de processador

Com hem dit en altres ocasions la freqüència màxima de treball ve donada fonamentalment per els retards en la propagació dels senyals, que es produeixen a les portes que constitueixen els diferents blocs del circuit, a causa les limitacions tecnològiques de la commutació. En aquest retard cal afegir el temps d'estabilització i manteniment dels senyals a les entrades.

Per altra banda, com és sabut, les línies de connexió metàl liques de longitud elevada també limiten la propagació dels senyals però en un percentatge molt més petit. Hem de dir però, que el nostre disseny no introdueix cap altre pèrdua de temps addicional ja que el processament dels segments rectilinis es produeix de forma totalment paral lela. Aquesta última qualitat és potser la més destacable del circuit.

Per tal de garantir que tots els senyal arribin al seu destí i s'estabilitzin abans que es produeixi la transició del senyal de rellotge, el període d'aquesta senyal a de ser més gran que el retard de propagació més llarg del circuit. Altrament els registres de les cel les de píxel corren el risc de carregar-se amb valors erronis.

El camí, dins del circuit, que produeix el retard més llarg s'anomena camí crític i òbviament determina la màxima freqüència de treball. En termes del nostre processador, que en ser totalment paral lel, es capaç de processar un segment rectilini en un únic període de rellotge, aquesta freqüència representa el nombre de segments rectilinis que es poden processar per unitat de temps.

Com que el processador que hem dissenyat està especialitzat en el càlcul del mapa d'acumulació de segments rectilinis, el nombre de segments que pot processar per unitat de temps es pot utilitzar com a mesura de la seva potència de càlculs i podem escriure:

$$Freq\_Treball_{\max} = \frac{1}{Tr_{\max}} = \frac{1}{Tc} \qquad \{6.1\}$$

$$Potència\ de\ càlcul = \frac{1}{Tps_{\max}}$$

$Tc = període\ mínim\ del\ senyal\ de\ rellotge$

$Tr_{\max} = retard\ de\ propagació\ màxim\ (camí\ crític)$

$Tps_{\max} = Temps\ màxim\ de\ \ pocessat\ d'un\ segment$

Com que el nostre processador és totalment paral·lel es compleix:

$$Tps_{\max} = Tc$$

La potència de càlcul resulta ser:

$$Potència\ de\ càlcul = \frac{1}{Tps_{\max}} = \frac{1}{Tc} = (Freq\_treball_{\max}) \quad segments/seg. \quad \{6.2\}$$

Segons això la determinació del camí crític es converteix en una tasca primordial ja que fixa la freqüència de treball i la potència de càlcul del circuit. Per tal d'obtenir aquest camí l'entorn disposa d'una eina de simulació (*Veritime*), que donada una senyal d'origen en un punt determinat del circuit i un punt destí, calcula els retards que es produeixen per tots els possibles camins alternatius que uneixen aquests dos punts, generats per les diferents opcions de funcionament. L'eina indican el camí amb el retard més llarg, és a dir, el camí crític entre els dos punts anteriors.

Per tal de fixar els punt origen i destí d'anàlisi s'ha de tenir en compte que els camins que els uneixen no continguin cap element simple síncron, tal com biestables, doncs aquests dispositius propaguen les senyals a partir d'un senyal de rellotge extern que el simulador no pot tenir en compte. En el circuit obtingut els únics elements que funcionen de forma síncrona són els registres de les cel·les de píxel, per tan els camins d'anàlisi tindran com a origen una entrada al circuit i com a destí l'entrada dels biestables del registre de la cel·la de píxel. Per tal d'obtenir el temps total fins a la sortida s'haurà d'afegir els temps de propagació d'aquest elements així com els temps d'estabilització i manteniment dels senyals a l'entrada.

Cal dir que l'eina de simulació no suporta el càlcul del temps de propagació a les memòries *ROM*. Quan el recorregut del senyal origen es troba una memòria d'aquest tipus, s'ha dividir l'anàlisi en dos trams, el d'abans i el de després de la memòria. Posteriorment afegir el temps de propagació de les memòries, obtingut del manual del fabricant. Aquest cas apareix en totes les senyal d'entrada de les unitats de càlcul ja que s'han implementat amb taules (LUT) a partir de memòries ROM (veure Fig.6.6 i 6.7).

S'ha de tenir en compta, per altra banda, que totes les entrades i sortides del circuit han de disposar d'un contacte de connexió a l'exterior, que en microelectrònica s'anomena "*Pad"* i proporciona la superfície metàl·lica necessària per tal de soldar les entrades/sortides del circuit als terminals de connexió de l'encapsulat. Aquest "*Pad"* porten associada una interfície d'adaptació en funció del tipus d'entrada a la que es connecten. La llibreria proporciona els diferents tipus de *Pads* que en el cas que ens ocupa són CMOS. Evidentment aquesta interfície introdueix un petit retard que el simulador tindrà en compte. Per aquesta raó la majoria de senyals origen partiran d'un *Pad*.

Per tal d'obtenir la millor aproximació del camins crítics s'han fet tres simulacions sota diferents condicions més o menys desfavorables. A continuació mostrem les condicions dels tres casos.

Cas1

- Estimació de capacitats i resistències dels cables amb l'algorisme Synopsys WorstCase. (Algorisme pessimista però compatible amb Synopsys)
- Temperatura 25.0 °C.
- Alimentació a 5.0 V.
- Procés a una velocitat típica.

Aquest conjunt de condicions de simulació facilitarà una visió de quines restriccions temporals haurien de respectar els diferents senyals per aconseguir que el circuit funcioni en unes condicions poc desfavorables.

Cas2

- Estimació de capacitats i resistències dels cables amb l'algorisme Synopsys WorstCase. (Algorisme pessimista però compatible amb Synopsys)
- Temperatura 125.0 °C.
- Alimentació a 4.5 V.
- Procés a una velocitat lenta.

Augmentant la temperatura, disminuint la tensió d'alimentació i triant la velocitat més lenta, obtenim uns restriccions temporals exigents que permetran que el circuit funcioni sota qualsevol condició de treball.

Cas3

- Estimació de capacitats i resistències dels cables amb l'algorisme WorstCase. (Algorisme molt pessimista. Indica error de *fanout* en alguns punts que es consideren exagerats i s'ignoraran.)
- Temperatura 125.0 °C.
- Alimentació a 4.5 V.
- Procés a una velocitat lenta.

Aquesta simulació serveix per saber a quina velocitat podríem fer anar el circuit i assegurar que funcionaria en el cas més pessimista en que s'ajunten totes les condicions desfavorables.

6.3.2.1 Determinació del camí crític en el processament de segments rectilinis

Per tal d'obtenir el camí crític per el cas en qual el circuit està en mode de processament de segments rectilinis s'han de determinar les senyals origen i els punt destí que introduirem al simulador. Partim de la llista de senyals del circuit de la taula 6.1 que intervindran en la determinació del camí crític. Aquestes senyals segons el que hem dit anteriorment consistiran en senyals d'entrada al circuit (veure Fig 4.14), sortides de les memòries *ROM* de càlcul o senyals de control.

A continuació introduï m al simulador els senyals origen i destí que defineixen camins que no intercepten ni memòries ni elements asíncrons. El simulador ens proporciona automàticament el temps més llarg entre aquests dos punts per totes les opcions de funcionament, és a dir, el temps crític de cada senyal.

A la Taula 6.2 mostrem els resultats obtinguts per els tres casos de simulació descrits en el apartat anterior.

Pin	Tamany	Descripció
X_1	8 bits	Coordenada x del punt inicial del segment.
Y_1	8 bits	Coordenada y del punt inicial. del segment
X_2	8 bits	Coordenada x del punt final del segment
Y_2	8 bits	Coordenada y del punt final del segment
Alfa (α)	7 bits	Angle del segment que substitueix la pendent *m* (veure AP.6.2.4)
m_1/m_2	1 bit	cas segons el pendent (*cas1 0£m £1, cas2 -¥£m£-1, 1£m ³¥)*
CC_RomExt	9 bits	Sortida de la ROM externa de càlcul comú de columna.
FC_RomExt	9 bits	Sortida de la ROM externa de càlcul comú de fila
CD_i_Rom	9 bits	Sortida de la ROM de la unitat de càlcul dedicada de columna.
FD_j_Rom	9 bits	Sortida de la ROM de la unitat de càlcul dedicada de fila.
P	4 bits	Ponderació del segment rectilini
RD	1 bit	Selecciona el mode de lectura o processament de segment
Clk	1 bit	Senyal de rellotge dels registres de les celles de pìxel.
Reset	1 bit	Reset dels registres de les celles de pìxel.

Taula 6.1: Senyals que intervindran en la determinació del camí crític

Origen	Desti	Cas1	Cas2	Cas3
Pad X_1	Estabilització de pin D dels biestables(1)	14.8 ns	29.2 ns	64.2 ns
Pad Y_1	Estabilització de pin D dels biestables	14.2 ns	28 ns	62.1 ns
Pad X_2	Estabilització de pin D dels biestables	5.1 ns	11.8 ns	23.1ns
Pad Y_2	Estabilització de pin D dels biestables	4.9 ns	11.3 ns	22.6 ns
Pad FC_RomExt	Estabilització de pin D dels biestables	14.7 ns	29.2 ns	64.1 ns
Pad CC_RomExt	Estabilització de pin D dels biestables	14.9 ns	29.6 ns	64.9 ns
Pad m_1/m_2	Estabilització de pin D dels biestables	16.1 ns	31.9 ns	72.9 ns
Pin FD_i_Rom	Estabilització de pin D dels biestables	14.1 ns	28 ns	60.8 ns
Pin CD_i_Rom	Estabilització de pin D dels biestables	13.7 ns	27 ns	58.9 ns
Pad P	Estabilització de pin D dels biestables	1.7 ns	3.4 ns	6.9 ns
Pad RD	Estabilització de pin D dels biestables	14.2 ns	28.1 ns	68.3 ns
Pad RD	Estabilització de pin S del multiplexor de cel· la	8.6 ns	17.5 ns	40.5 ns
Pad Clk	Pin CP dels biestables(2)	1.9 ns	4.1 ns	11.3 ns
Pad Reset	Pin CD dels biestables(3)	1.9 ns	4.1 ns	11.3 ns

(1) D - Senyal d'entrada dels biestables.
(2) CP- Senyal de rellotge dels biestables. (Activa al flanc de pujada)
(3) CD- Senyal d'esborrat dels biestables. (Activa a nivell baix

Taula 6.2: Temps crítics de propagació entre diferents punts del circuit per els tres casos de simulació.

Recordem que els senyals de rellotge *Clk* i de *Reset* van als registres de totes les celles processadores i s'han separat, per tal d'equilibrar els temps de propagació i les capacitats de càrrega, amb una estructura de inversors en arbre de tres nivells (Fig. 6.5), segons es va descriure en l'apartat 6.2.3, la qual cosa introdueix retards en aquestes línies. El camí crític de l'estructura en arbre s'haurà d'afegir per tal de garantir que els registres gravaran valors estabilitzats. Per aquesta raó apareixen els retards dels senyals de *Clk* i de *Reset* a la taula.

Finalment per tal d'obtenir el temps total necessari per gravar en els registres el valor d'acumulació s'han de tenir en compte les restriccions temporals dels propis biestables per tal d'evitar que pugin entrar en meta-estabilitat. La taula 6.3 mostra aquestes restriccions.

Els resultats temporals de la simulació anterior conjuntament amb les dades dels temps de propagació de les memòries *ROM* de càlcul i les restriccions temporals dels registres, permeten calcular el temps màxim de propagació dels senyals entre diferents punts del circuit. El camí que resulti ser el més llarg, en termes de temps de propagació, serà el camí crític global del circuit i determina el temps que s'ha d'esperar des que presentem les dades al circuit fins que validem els resultats en els registres de les cel·les de píxel.

Descripció	**Temps**
Temps d'estabilització de D respecte el flanc ascendent de CP - TD_S	0.534 ns
Temps de manteniment de D respecte el flanc ascendent de CP - TD_H	0.039 ns
Temps d'amplada de CP a nivell *'0'* - TCP_L	0.817 ns
Temps d'amplada de CP a nivell *'1'* - TCP_H	0.711 ns
Temps d'amplada de CD a nivell *'0'* - TCD_L	0.749 ns
Temps des del flanc ascendent de CD fins l flanc ascendent de CP - T_{CDCP}	1.279 ns
D - Senyal d'entrada dels biestables. CP- Senyal de rellotge dels biestables. (Activa al flanc de pujada) CD- Senyal d'esborrat dels biestables. (Activa a nivell baix	

Taula 6.3: Restriccions temporals dels registres de la cel·la de píxel

El moment de validació de les dades el proporciona el flanc ascendent del senyal de rellotge que activa els biestables, per tant el temps anterior és també el període mínim del senyal de rellotge que garanteix un bon funcionament del circuit.

El sistema de càlcul consisteix en sumar, per cada senyal d'entrada al circuit, el pitjor temps d'estabilització a l'entrada *D* dels registres (T_{critic}) amb el temps de manteniment que els biestables necessiten per evitar la meta-estabilitat TD_S. Al resultat se li resta el temps de propagació del senyal de rellotge des del *Pad* d'entrada fins l'entrada de validació CP dels biestables de la taula 6.3 anterior, i que anomenem T_{ClkD}, ja que aquest temps afavoreix els retards. El temps de manteniment a l'entrada D dels biestables no cal tenir-lo en compte ja que és tant petit que encara que retiréssim el senyal d'entrada immediatament després de generar el flanc ascendent del senyal de rellotge no el violaríem. Així doncs podem calcular el temps màxim per cada entrada TCH_S com :

$$TCH_S = T_{critic} + TD_S - TClk_D$$

TCH_S és, per tant, el temps de estabilització, per els diferents senyals d'entrada, respecte el flanc ascendent del senyal de rellotge del circuit, el qual genera l'enregistrament dels resultats en els registres de les cel·les de píxel

Com TD_S *i* $TClk_D$ son constant per cada cas de simulació es pot escriure:

$$TCH_s = T_{critic} + TD_s - TClk_D = T_{critic} + 0.534 - 1.9 = T_{critic} - 1.366 \quad \{\text{Cas1}\}$$

$$TCH_s = T_{critic} + TD_s - TClk_D = T_{critic} + 0.534 - 4.1 = T_{critic} - 3.566 \quad \{\text{Cas 2}\}$$

$$TCH_s = T_{critic} + TD_s - TClk_D = T_{critic} + 0.534 - 11.3 = T_{critic} - 10.766 \quad \{\text{Cas 3}\}$$

A la taula 6.4 mostrem els temps resultants, per els tres casos de simulació proposats, on es pot veure com els senyals que generen el camí crític són els que entren a les memòries *ROM* externes de càlcul comú, és a dir, *Alfa(a)* i X_1.

La taula 6.5 mostra els temps de propagació del camí crític globals del circuit per els tres casos de simulació.

SENYAL	CAS 1 $TCH_S = T_{critic} - 1.366$		CAS 2 $TCH_S = T_{critic} - 3.566$		CAS 3 $TCH_S = T_{critic} - 10.766$	
	T_{CRITIC}	TCH_S ns	T_{CRITIC}	TCH_S ns	T_{CRITIC}	TCH_S ns
X_1 (1)	14.8	13.43	29.2	25.63	64.2	53.434
Y_1 (1)	14.2	12.83	28.0	24.43	62.1	51.334
X_1 (2)	**3.9+14.9**	**17.43**	**3.9+29.6**	**29.93**	**3.9+64.9**	**58.03**
Y_1 (2)	3.9+14.7	17.23	3.9+29.2	29.53	3.9+64.1	57.23
X_2	5.1	3.73	11.8	8.23	23.1	12.33
Y_2	4.9	3.53	11.3	7.73	22.6	11.83
Alfa(α)(3)	3.9+14.7	17.23	3.9+29.2	29.53	3.9+64.1	57.23
Alfa(a)(4)	**3.9+14.9**	**17.43**	**3.9+29.6**	**29.93**	**3.9+64.9**	**58.03**
Alfa(α)(5	3.9+14.1	16.63	3.9+28	28.33	3.9+60.8	53.93
Alfa(α)(6)	3.9+13.7	16.23	3.9+27	27.33	3.9+58.9	52.03
m_1/m_2	16.1	1473	31.9	28.33	72.9	62.13
P	3.7	2.33	7.4	3.83	16.9	6.13
RD	14.2	10.634	17.5	13.93	40.5	29.74

(1) per camins que no inclouen memòries ROM de càlcul
(2) per camins que inclouen memòries ROM cal afegir els retard de les memòries.
(3) a l'entrada de la Rom externa de càlcul comú de fila
(4) a l'entrada de la Rom externa de càlcul comú de columna
(5) a l'entrada de la Rom de càlcul dedicada de fila
(6) a l'entrada de la Rom externa de càlcul dedicada de columna

Taula 6.4: Temps d'estabilització dels senyals d'entrada al circuit per el tres casos de simulació

Simulació	Temps Crític
Cas1	17.43ns
Cas2	29.93ns
Cas3	58.03

Taula 6.5: Temps de propagació del camí crític del circuit

6.3.2.2 Càlcul de la freqüència màxima de treball i de la potència de càlcul

Amb els resultats anteriors estem en condicions de calcular les freqüències màximes a les quals pot treballar el circuit per el tres casos de simulació proposats. Partim de l'equació {6.1 } del apartat 6.3.2:

$$Freqüencia\ màxima\ de\ treball == \frac{1}{Tc}$$

$$Tc = retard\ de\ propagació\ del\ camí\ crític$$

Així si considerem la potència de càlcul com el nombre de segments que el processador pot tractar per unitat de temps, tal com es va definir en el apartat 6.3.2, podem aplicar la l'equació {6.2}allí obtinguda :

$$Potència\ de\ càlcul = (Freq\_treball_{\max})\quad segments/seg.$$

Que proporciona els resultats que mostrem a la taula 6.5.

Simulació	$Freq\_treball_{max}$	Potencia de càlcul
Cas1	57.3 MHz	57.3 Msegments/seg
Cas2	33.MHz	33.Msegments /seg
Cas3	17.2 MHZ	17.2 Msegments /seg

Taula 6.6: Freqüència màxima de treball i potència de càlcul per el tres casos de simulació

6.3.2.3 Temps d'accés a les dades

Per tal d'obtenir el temps d'accés de les dades emmagatzemades en els registres de les celles de píxel hem d'obtenir els camins crítics dels senyals d'entrada que intervenen en aquest accés. A la taula 6.7 mostrem aquest senyals. Com que no intercepten cap element síncron, doncs només validen la sortida de la cel·la seleccionada (veure Fig.6.1) els camíns crític es poden obtenir de forma automàtica, directament de l'eina de simulació. Hem de proporcionar-li com a punts d'origen dels camins, les entrades de la taula i com a destí l'estabilització dels senyals en els "pads" de l'etapa de sortida.

Pin	Amplada	Descripció
@_Fila	5 bits	Adreça de la fila de la cel· la de píxel que es vol accedir.
@_Col	5 bits	Adreça de la columna de la cel· la de píxel que es vol accedir.
Llindar	7 bits	Llindar de comparació del valor d'acumulació
RD	8 bits	Coordenada x del punt final del segment

Taula 6.7: Senyals que intervindran en la determinació del camí crític d'accés

Hem de recordar però, que segons les modificacions introduï des a l'arquitectura de les celles de píxel, en el procés de reducció del disseny VLSI (veure apartat 6.2.1) no s'obtenen els valors dels registres de les celles sinó el bit *"més gran o igual"* resultant de la comparació d'aquest valor amb un llindar que introduï m externament al circuit.
Per altra banda, s'ha de tenir en compte que molts accessos, segons l'algorisme de detecció que hem descrit en els apartats 3.8.1, 3.8.2, es faran amb al mateix llindar de comparació. Això significa que tindrem dos temps d'accés, segons que el llindar de comparació estigui estabilitzats d'una operació de lectura anterior o que s'hagi d'esperar l'estabilització d'aquests senyals per un canvi de llindar.
Finalment tenint en compte que les operacions de lectura i processament de segments són mútuament exclusives podem fer els càlculs amb el senyal de lectura *RD* permanentment estabilitzada o no. Aquest cas dóna un tercer temps d'accés a tenir en compte.

A la taula 6.8 podem veure el resultats de la simulació per el camins que intervenen en l'accés de lectura i per els tres supòsits de simulació.

Origen	**Desti**	Cas1	Cas2	Cas3
Pad @F	Estabilització al pad de sortida del bus de dades	1.82 ns	3.27 ns	6.75 ns
Pad @C	Estabilització al pad de sortida del bus dades	2.05 ns	3.41 ns	7.14 ns
Pad Llindar	Estabilització al pad de sortida del bus de dades	3 ns	6.94 ns	13.68ns
Pad RD	Estabilització al pad de sortida del bus de dades	3.1 ns	7.14 ns	14.08ns

Taula 6.8: Temps d'estabilització dels senyals d'entrada que intervenen en l'accés, per el tres supòsits de simulació

Com era d'esperar el pitjor temps de resposta el tenim respecte el senyal de mode d'operació *RD*, és a dir, quan l'accés es produeix per primer cop de manera que s'ha d'esperar la propagació del llindar i la commutació del mode. Contràriament en operacions successives el mode i el llindar estan estabilitzats i obtenim temps molt millor respecte les adreces de fila @F i columna @C.

La taula 6.8 dóna directament els tres temps d'accés esmentats que resumim a la taula 6.9

Temps d'accés	Cas1	Cas2	Cas3
Respecta l'adreça de fila o columna de cel· la	2.05 ns	3.41 ns	7.14 ns
Respecta el Llindar de comparació	3 ns	6.94 ns	13.68ns
Respecta el mode d'operació *RD*	3.1 ns	7.14 ns	14.08ns

Taula 6.9: Temps d'accés del circuit respecte els diferents senyals que intervenen i per els tres supòsits de simulació

Segons això la freqüència màxima de treball, quan les condicions són més favorables i en operacions de lectura successives es de:

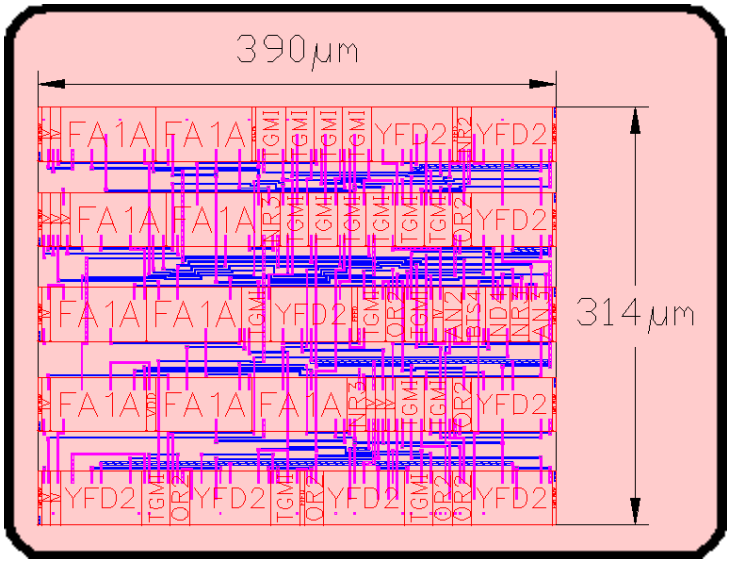
$$Max(F_{accés}) = 487\ Mpíxel / seg.$$

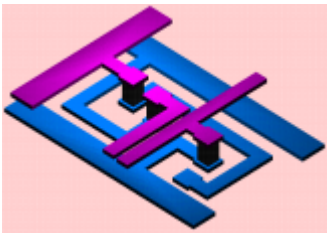
6.3.3 Migració a una tecnologia de més alta escala d'integració

Al final del disseny del nostre circuit hem tingut accés a una tecnologia que millora l'escala d'integració de 0,7µm a 0,35µm i passa de dos capes de metall a tres. Per tal de poder explorar les possibilitats de millora d'aquesta tecnologia hem redissenyat la cel·la processadora de píxel ja que aquesta unitat es la de més alta replicació de l'arquitectura. Concretament, $M \times N$ unitats en funció de la resolució de la imatge, que en el cas que ens ocupa de *32X32* unitats, representa el 85% de totes les unitats del circuit.

Els resultats han estat espectacularment favorables ja que si el factor de reducció d'àrea que es podia esperar és de $2 \times 0.7/0.35 = 4$ en realitat gràcies a la inclusió d'una nova capa de metall, el factor ha augmentat a 7,7. A la la Fig 6.13. es pot veure una comparació entre les dimensions obtingudes per les dues tecnologies. Si extrapolem aquesta reducció a tot el circuit obtenim l'àrea total de silici necessària per implementar un circuit en aquesta tecnologia i per una resolució de *32X32*

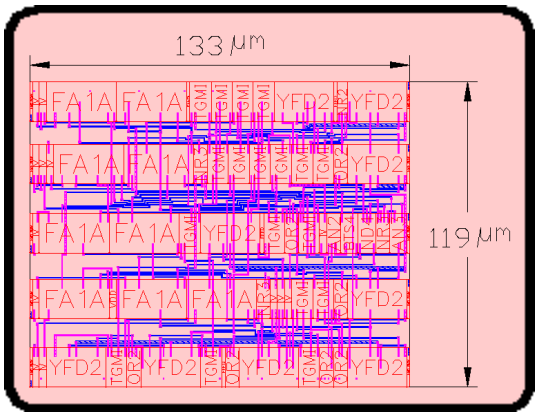
$$Area_{0.35\mu m} = Area_{0.7\mu m} / factor\ reductor = (18744{,}6\,\mu m \times 21515{,}9\,\mu m) / 7.7 = 0.52\,cm^2$$

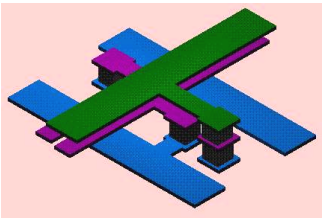




0.7 µm i dos capes de metall

(a)





0.35µm i tres capes de metall

(b)

Figura 6.13: Comparació de dimensions de la cel· la processadora de píxel. (a) Tecnologia 0,7µm amb dues capes de metall (b) Tecnologia 0,35µm amb tres capa de metall

que representa un quadrat de 0.72cm X 0.72 cm que és absolutament implementable i permet plantejar augmentar la resolució a més del doble, és a dir, incorporar una matriu de cel·les de píxel superior a 64X64.

6.3.4 Conclusions

Per tal d'obtenir un circuit d'alt nivell d'integració amb cel·les estàndard a partir de l'arquitectura proposada en el capítol 4, s'ha adaptat el disseny als requeriments d'aquesta tecnologia. Per tant, la majoria de modificacions estan lligades a restriccions

físiques relacionades amb les capacitats de càrrega de les línies de connexió, l'adaptació de consum entre entrades i sortides i evitar la dispersió en el temps de propagació dels senyals globals per tal d'aconseguir la sincronització.

L'altre restricció tecnològica que s'ha tingut molt en compte és l'àrea de silici utilitzada a causa de l'alt nombre de unitats que incorpora el disseny. En aquest sentit s'ha intentat utilitzar sempre els elements de la llibreria que, sense comprometre el marge de seguretat de funcionament, utilitzen menys silici. Alguns cops aquesta opció només ha estat possible sacrificant velocitat.

També amb l'objectiu d'estalviar àrea hem modificat l'accés als registres de les cel·les de píxel. Si bé en el disseny original es va disposa un accés de tipus aleatori, aquí hem utilitzat el fet que el mètode proposat en el capítol 3 no necessita conèixer el valor dels registres de les cel·les de píxel sinó saber si aquest valor supera un llindar. La modificació a consistit en aprofitar el comparador existent a les cel·les de píxel per realitzar aquesta nova comparació i treure un bit de confirmació o negació.

Amb aquestes modificacions hem aconseguit obtenir un disseny verificat del processador dedicat al càlcul del mapa d'acumulació de segments rectilinis, proposat en el capítol 3 per una resolució de 32X32 en un sol circuit i adaptat a la tecnologia VLSI basat en cel·les estàndard. Hem simulat el comportament temporal del circuit i obtingut el camí crític i en conseqüència la freqüència màxima de treball i la capacitat de càlcul.

Els resultats obtinguts ens han permès treure les següents conclusions.

- Amb una tecnologia de 0,7µm amb dos capes de metall, la implementació del circuit amb una matriu de 32X32 cel·les de píxel, ens porta a unes dimensions pràcticament al límit acceptable.

- La migració a una tecnologia de 0,35 µm amb tres capes de metall proporciona una millora superior a la esperada, gràcies a la incorporació de la tercera capa de metall. Aquesta millora fa viable l'obtenció d'un circuit que suporti una matriu de cel·les de píxels superior a 64X64.

- Per suportar resolucions superiors s'ha de canviar la tecnologia de cel·les estàndard per una tecnologia *FC* (*Full Custom*) que fa un us més òptim del silici.

- Donada l'alta replicació de la cel·la de píxel, la millora d'àrea que es podria obtenir per un disseny a mida més òptim (*Full Custom*) podria donar un augment molt considerable de la resolució suportada per l'arquitectura, mantenint la resta d'unitats tal com estan. Això significaria utilitzar una tecnologia mixta.

- La simulació temporal ha mostrat que, tot i algun sacrificis de velocitat realitzats per tal d'estalviar àrea, les prestacions temporals del circuit són molt bones a causa del 100% de paral·lelisme de l'arquitectura.

- La bondat dels resultats temporals donen la possibilitat de sacrificar velocitat reduint el grau de paral·lelisme. Això aportaria una reducció en l'àrea de silici necessari, de manera que podríem obtenir resolucions encara més elevades.