Systema Naturae, 2004, Vol. 6, pp. 153-190

IL PROGETTO EMBRYONICS: MECCANISMI DI SVILUPPO IN ORGANISMI ARTIFICIALI

Gianluca Tempesti, Daniel Mange, Enrico Petraglio, André Stauffer
Logic Systems Laboratory
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
gianluca.tempesti@epfl.ch

Indice

1 Introduzione
2 Alcuni concetti di base di elettronica
  2.1 Field-Programmable Gate Arrays
3 Il progetto Embryonics e le cellule artificiali
4 L'algoritmo "Tom Thumb" per la divisione di cellule artificiali
  4.1 La divisione cellulare nell'universo della biologia
  4.2 Un universo per la divisione di cellule artificiali
  4.3 L'algoritmo "Tom Thumb"
    4.3.1 Condizioni iniziali
    4.3.2 Costruzione della cellula
    4.3.3 La divisione cellulare
    4.3.4 Crescita di un organismo multicellulare
    4.3.5 Priorità di connessione
5 La differenziazione cellulare in un organismo artificiale
  5.1 La divisione cellulare nell'universo della biologia
  5.2 La differenziazione cellulare in Embryonics
  5.3 Generalizzazione delle cellule artificiali
  5.4 Un esempio di differenziazione cellulare
6 Conclusione
Bibliografia

1 Introduzione

Un essere umano è composto approssimativamente da 60 mila miliardi ($6x10^{13}$) di cellule. In ogni istante, in ognuna di queste cellule, il genoma, un nastro di 2 miliardi di caratteri, è interpretato per produrre le proteine necessarie alla sopravvivenza dell'organismo, un processo che si ripete in continuazione dalla concezione alla morte dell'individuo.

Questo processo, impressionante per complessità e precisione, è basato su meccanismi completamente discreti: la struttura chimica del DNA (il substrato chimico del genoma) è una sequenza di quattro basi, indicate con le lettere A (adenina), C (citosina), G (guanina), e T (timina). Ogni gruppo di tre basi è interpretato nella cellula per produrre un aminoacido particolare, uno dei componenti della proteina finale.

Un programma informatico, dal canto suo, è una sequenza di due stati, indicati con le cifre 0 e 1, divisa in gruppi (le istruzioni) che sono interpretati dai processori per eseguire una data funzione. La somiglianza tra il genoma naturale e un programma informatico è immediata ed è una della motivazioni per la ricerca nel campo dell'hardware bio-ispirato [7][8][17][24] in generale e per il progetto Embryonics (elettronica embrionale) in particolare.

Questo progetto vuole sviluppare una nuova metodologia per la concezione di circuiti elettronici digitali ispirata dai processi ontogenetici che guidano lo sviluppo degli organismi multicellulari in natura. Esso è stato descritto in numerose pubblicazioni, compresa una in questa stessa rivista [9][11][12][18][19]. La nostra soluzione si fonda su un'architettura multicellulare basata su quattro livelli di complessità: le popolazioni, gli organismi, le cellule e le molecole (Figura 1). Come vedremo, il nostro approccio ci permette non solo di rispettare (anche se indirettamente, a causa delle numerose differenze di base tra l'elettronica e la biologia) molte delle definizioni di base della biologia, ma anche di sfruttare alcuni dei meccanismi specializzati alla base dello sviluppo ontogenetico di un organismo.

La motivazione di un tale tentativo dovrebbe essere ovvia, considerando che gli organismi multicellulari sono esempi impressionanti di sistemi massivamente paralleli: le $6x10^{13}$ cellule di un corpo umano, elementi relativamente semplici, lavorano insieme per esibire comportamenti estremamente complessi (tra i quali il più impressionante è, naturalmente, l'intelligenza). Se consideriamo la difficoltà di programmazione dei

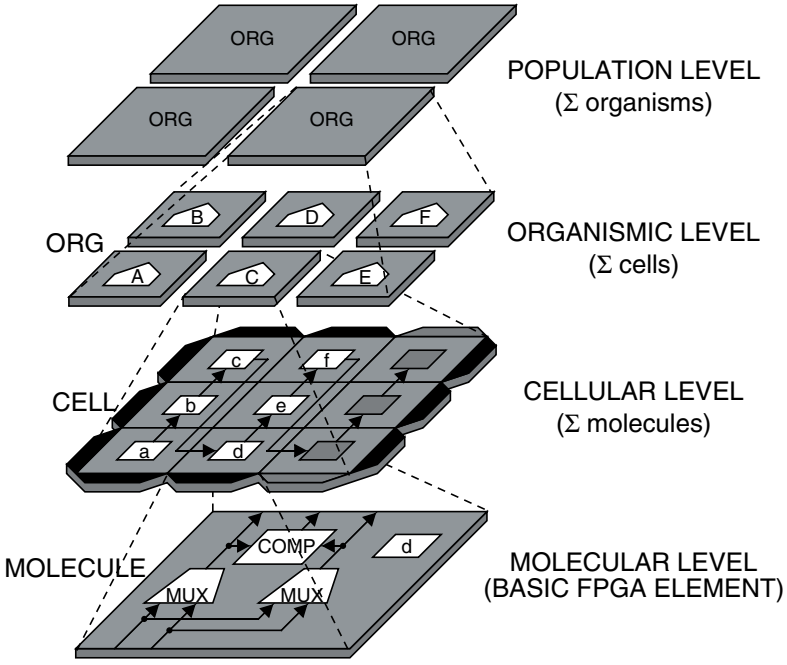


Figura 1 - I quattro livelli del progetto Embryonics: popolazioni, organismi, cellule e molecole.

computer paralleli (una difficoltà che ha causato un declino nella loro popolarità), l'ispirazione biologica potrebbe suggerire degli approcci per trattare il parallelismo massivo nell'elettronica. Inoltre, la sorprendente capacità degli organismi biologici di sopravvivere a danni considerevoli può essere di grande interesse nella concezione di circuiti digitali, la cui crescente complessità sta facendo emergere il problema della tolleranza ai difetti (l'abilità di funzionare nonostante la presenza di difetti nel substrato di silicio di un circuito).

Visto nel suo complesso, è nostra opinione che questo approccio a quattro livelli presenti una discreta somiglianza con i sistemi biologici, somiglianza che potrebbe in futuro essere ancora aumentata, per esempio, con l'introduzione di meccanismi quali l'evoluzione o l'apprendimento al livello degli organismi o delle cellule, o anche con l'aggiunta di un quinto livello (atomico) al sistema.

Nella conclusione del nostro precedente articolo pubblicato in questa

rivista, avevamo identificato un punto debole nel parallelo tra i nostri sistemi e gli organismi multicellulari. Le nostre cellule artificiali, infatti, se da molti punti di vista sembrano rispettare molte delle caratteristiche fondamentali delle cellule naturali, non erano in grado di dividersi in modo autonomo, vale a dire senza l'assistenza di un entità esterna all'organismo o più precisamente senza una sequenza di configurazione introdotta dall'esterno della matrice cellulare (termini che saranno definiti nella parte introduttiva di questo articolo).

Per trovare una soluzione a questo problema abbiamo sviluppato un meccanismo di divisione cellulare che può essere controllato dalle cellule stesse, senza un intervento esterno. Tale meccanismo è il soggetto di questo articolo. Cominceremo (Sezione 2) con un breve riassunto, tratto dal nostro precedente articolo [18], delle cognizioni di base di elettronica che stimiamo indispensabili per comprendere alcune delle scelte e delle soluzione da noi adottate. Dopo aver dato una visione d'insieme del progetto Embryonics in generale e della struttura delle nostre cellule artificiali in particolare (Sezione 3), presenteremo l'algoritmo *Tom Thumb* da noi sviluppato per realizzare la divisione di tali cellule all'interno di un circuito elettronico (Sezione 4). Prima di concludere l'articolo (Sezione 6), illustreremo come questo algoritmo può essere integrato alla differenziazione cellulare (Sezione 5) per realizzare la crescita di organismi arbitrariamente complessi nel nostro sistema.

2 Alcuni concetti di base di elettronica

In questa sezione, tratta dall'articolo precedentemente pubblicato in questa stessa rivista [18], cercheremo di definire alcuni dei principali termini usati nel resto dell'articolo, nella speranza che una tale spiegazione, benché molto superficiale, sarà di aiuto per meglio comprendere il soggetto. Molti tentativi di volgarizzazione più completi sono naturalmente disponibili. Per esempio, [13] è un'introduzione abbastanza completa e di facile lettura, allo sviluppo di circuiti digitali, mentre [6] è un testo più "serio" e rigoroso (ma molto più tecnico).

L'operazione di un computer è basata su due concetti essenziali: l'*hardware* e il *software*. Una chiara comprensione della differenza tra questi due concetti è essenziale e non sempre così ovvia come potrebbe apparire.

L'hardware digitale (non si tratterà in questo articolo di hardware analogico) è basato su un elemento di base, il transistore (*transistor*), che può essere considerato come un semplice interruttore che permette o impedisce il passaggio di corrente elettrica attraverso una pista metallica. L'apertura e la chiusura degli interruttori appropriati permettono alle piste di avere valore 1 o 0, dove 1 rappresenta il voltaggio operativo del circuito, convenzionalmente 5 volt, anche se la maggioranza dei circuiti attuali lavorano su voltaggi più bassi e 0 rappresenta la massa del circuito, 0 volt. Tutti i circuiti digitali sono, ad una prima approssimazione, collezioni di interruttori.

Il software è il mondo dell'informazione, in forma digitale. Per i computer, esistono due tipi principali d'informazione: programmi e dati. I programmi sono sequenze d'istruzioni che indicano al processore di un computer quali operazioni eseguire. I dati sono il soggetto dell'operazione del processore, e possono rappresentare immagini, suoni, testo, ecc.

È comunque cruciale ricordarsi che *tutto* il software, indipendentemente del suo significato ultimo (istruzioni, suoni, immagini, ecc.), è una sequenza di bit, cioè una sequenza unidimensionale di cifre 0 e 1 (le uniche che possono essere trattate dall'hardware). Non esiste alcuna differenza intrinseca tra sequenze che rappresentano, per esempio, il programma di elaborazione testi usato per scrivere questo articolo e il testo dell'articolo stesso. È il processore (e quindi l'hardware) che deve *interpretare* il significato delle sequenze e operare di conseguenza.

Il processore, l'esempio più rappresentativo di hardware digitale, è quindi responsabile della manipolazione (*processing*) dei dati: esso riceve una sequenza di bit che corrisponde ad una sequenza d'istruzioni, la interpreta per determinare la propria funzionalità, ed esegue le operazioni indicate sui dati appropriati.

È utile puntualizzare che, in realtà, i processori rappresentano solo una piccola percentuale di tutti i circuiti digitali in uso corrente. La grande maggioranza dei circuiti sono infatti destinati a eseguire (molto rapidamente) una singola specifica operazione. Tali circuiti sono chiamati *application-specific* (dedicati ad un'applicazione particolare), mentre i processori sono definiti *general-purpose* (a utilizzo generale).

Mentre il software è un concetto che sta diventando sempre più corrente, i metodi di concezione dei circuiti digitali non sono generalmente conosciuti dal pubblico e ovviamente questo articolo non è una sede adeguata per una descrizione dettagliata. Nonostante ciò, dedicheremo

qualche paragrafo ad una definizione di alcuni dei termini principali usati in questo articolo.
Anche se l'elemento di base dei circuiti elettronici digitali restano i transistor, la concezione di circuiti complessi attraverso la loro manipolazione diretta è estremamente difficile. Di conseguenza, i transistori sono raggruppati per formare *elementi logici* di varia complessità, capaci di implementare semplici funzioni (Figura 2). Tra gli elementi logici più comuni, menzioneremo:

- l'inversore (*inverter*), la cui uscita è l'inverso dell'entrata;
- la porta E (*AND gate*), la cui uscita è 1 se e solo se tutte le entrate sono 1;
- la porta O (*OR gate*), la cui uscita è 0 se e solo se tutte le entrate sono 0;
- il *multiplexer*, la cui uscita assume il valore di una delle entrate, selezionata da una variabile di controllo;
- il *flip-flop* è l'elemento di memoria di base, capace di memorizzare il valore di un singolo bit;
- il registro (*register*) è un gruppo di flip-flop, messi uno accanto all'altro per memorizzare multipli bit;
- il registro a scorrimento (*shift register*) è un tipo particolare di registro in cui tutti i flip-flop sono collegati in modo tale che il valore da memorizzare nel registro entra sequenzialmente da un lato. Come vedremo, questo tipo di registro è fondamentale nell'algoritmo di divisione cellulare che presenteremo in quest'articolo.

Un circuito che non contiene alcun elemento di memoria (flip-flop, registri) è chiamato combinatorio (*combinational*), in caso contrario sequenziale (*sequential*). I circuiti sequenziali sono controllati da un orologio (*clock*) che determina la frequenza alla quale i valori sono caricati negli elementi di memoria.

La concezione di circuiti digitali consiste dunque nel mettere insieme questi elementi per realizzare funzioni che possono essere estremamente complesse. La distinzione tra hardware e software sta tuttavia diventando sempre meno netta in seguito all'introduzione di una nuova generazione di circuiti, chiamati *FPGA*. Questi circuiti, descritti più sotto, sono privi di funzionalità ma possono essere configurati con una sequenza di bit (del software, quindi) per dare loro una qualsiasi struttura ovvero una qualsiasi funzionalità. Gli FPGA, permettendo al software di alterare l'hardware, hanno aperto la strada allo sviluppo di hardware bio-ispirato.

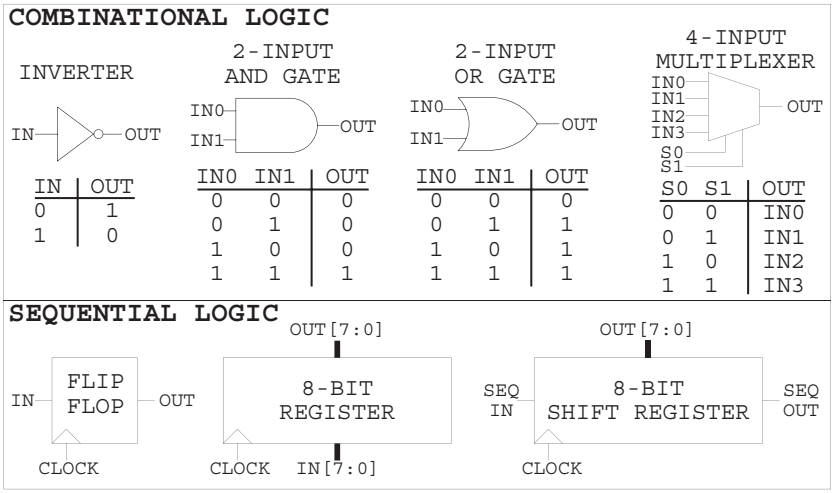


Figura 2 - Alcuni elementi logici usati nella concezione di circuiti digitali

2.1 Field-Programmable Gate Arrays

L'ostacolo principale alla realizzazione di hardware bio-ispirato è stata finora la necessità di tali sistemi di modificare la struttura dell'hardware per implementare proprietà quali l'auto-replicazione o l'evoluzione. Fino a pochi anni fa tali modifiche erano essenzialmente impossibili: un circuito era concepito e costruito per eseguire una funzione specifica, dal momento che i transistor e le piste metalliche, una volta disegnati sul substrato in silicio di un circuito, non petevano più essere modificati.
Verso la fine degli anni ottanta, però, un nuovo tipo di circuito fu introdotto sul mercato. Questi circuiti, chiamati FPGA (*Field-Programmable Gate Arrays*, o matrici di porte programmabili sul campo) [1][13][25], sono matrici bidimensionali di elementi logici che possono essere *configurati* (cioè programmati via software) per realizzare qualsiasi funzione logica (vale a dire qualsiasi circuito logico digitale).
Mentre la struttura esatta e la taglia di un elemento di FPGA possono variare in modo considerevole da un fabbricante ad un altro, alcune caratteristiche fondamentali rimangono costanti (Figura 3):

- Ogni elemento può implementare una *funzione programmabile* che consiste di logica combinatoria più almeno un elemento di memoria

(flip-flop) per i circuiti sequenziali. La complessità e la struttura della funzione programmabile possono variare considerevolmente da un FPGA ad un altro.
- Lo scambio di informazioni tra gli elementi è realizzato con una serie di *connessioni programmabili*, la cui complessità è anch'essa variabile.
- La funzionalità e le connessioni di un elemento sono controllati dalla sua *configurazione*, una sequenza di bit (normalmente memorizzati in un registro a scorrimento) che indica le parti della logica funzionale e le connessioni che devono essere attive. La *sequenza di configurazione* , cioè la somma delle configurazioni di tutti gli elementi, determina il comportamento globale del circuito.

Una data sequenza di configurazione assegnerà dunque funzioni differenti ad ogni elemento e connetterà insieme gli elementi per realizzare il comportamento globale desiderato. Con la configurazione appropriata, un FPGA può realizzare qualsiasi circuito logico digitale, a condizione naturalmente che un numero sufficiente di elementi sia disponibile. Inoltre, nella maggior parte dei casi, gli FPGA sono riprogrammabili, vale a dire che la loro configurazione può essere cancellata e sostituita con un'altra, cambiando così il circuito realizzato dall'FPGA.

Ovviamente la notevole versatilità degli FPGA ha un prezzo: la rapidità. Circuiti realizzati con un FPGA sono necessariamente molto più lenti di un circuito integrato, visto che devono operare a una frequenza molto più bassa. In certi casi, però, la versatilità degli FPGA può compensare questo inconveniente. Questo avviene quando la rapidità non è un fattore essenziale e il circuito necessita alterazioni dinamiche in corso di funzionamento, oppure quando il vantaggio di avere processori specializzati, spesso paralleli, riesce a compensare la diminuzione della frequenza di funzionamento. Per esempio, certe operazioni matematiche che richiederebbero molti cicli d'orologio in un processore ad uso generale possono essere eseguite da un processore specializzato in un solo ciclo, anche se più lento.

Il progetto Embryonics spera di sfruttare quest'ultima considerazione, cercando di compensare la lentezza relativa degli FPGA con l'uso di sistemi di calcolo paralleli specializzati per un'applicazione particolare. La riprogrammabilità degli FPGA è infatti la soluzione ideale al problema di realizzare macchine bio-ispirate, dal momento che permette di modificare la struttura hardware di un circuito attraverso una modifica del software (la sequenza di configurazione).

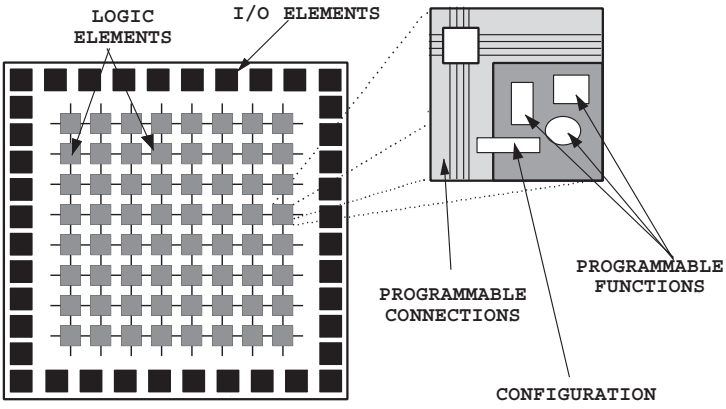


Figura 3 - Architettura di base di un FPGA.

3 Il progetto Embryonics e le cellule artificiali

Lo sviluppo di un organismo biologico multicellulare coinvolge una serie di processi che guidano la crescita dell'individuo, cioè lo sviluppo di un organismo da una singola cellula madre (lo *zigote*) a un individuo adulto. Lo zigote si divide e ognuna delle due cellule risultanti contiene una copia del genoma; questa *divisione cellulare* continua (ogni nuova cellula si divide in due altre cellule, e così via) finché ogni nuova cellula acquisisce una funzionalità (per esempio, cellula del fegato, dell'epidermide, ecc.) a seconda della sua posizione in relazione alle cellule vicine (*differenziazione cellulare*). Chiamiamo *ontogenesi* l'insieme dei processi che determinano l'evoluzione di un organismo dall'embrione all'individuo adulto.

Per trovare un approccio pratico alla concezione di sistemi di calcolo ispirati all'ontogenesi degli organismi biologici multicellulari, abbiamo tentato di identificare alcune tra le caratteristiche principali di tali organismi:

- In biologia, un organismo è una matrice di cellule che operano in parallelo per generare processi globali (cioè processi che coinvolgono l'organismo intero). Per rispettare l'analogia, il nostro organismo artificiale è un sistema di calcolo che consiste anch'esso di una matrice

di elementi che operano in parallelo per eseguire un compito globale (un'applicazione). La *matrice cellulare* è nel nostro caso ovviamente bidimensionale per rispettare le caratteristiche dei circuiti elettronici. La grandezza (il numero di cellule) di un organismo è programmabile e, se un numero sufficiente di cellule sono disponibili, gli organismi si replicano automaticamente. Inoltre, dal momento che ogni copia dell'organismo ha la stessa funzionalità, questo meccanismo può essere usato per dare al sistema una tolleranza intrinseca agli errori.

- In biologia, ogni cellula contiene il genoma, cioè la descrizione dell'organismo, interpretato per determinare la funzionalità della cellula. Nel nostro sistema ogni cellula artificiale è un processore semplice ma universale, capace cioè di eseguire qualsiasi applicazione voluta, che interpreta ed esegue lo stesso programma di tutte le altre cellule nell'organismo, il nostro *genoma artificiale*.
- In biologia, nessuna cellula usa l'intero genoma, ma accede solo alla porzione necessaria per la propria funzione. In modo simile, nessun processore eseguirà tutte le istruzioni del programma, ma userà la propria posizione all'interno della matrice per identificare quale parte del programma è rilevante. Dal momento che il genoma è duplicato in ogni cellula e che tutte le cellule hanno la stessa struttura hardware, una cellula morta può essere rimpiazzata da un'altra semplicemente ricalcolando la sua posizione nella matrice, un meccanismo che conferisce al circuito un'ulteriore resistenza agli errori.
- In biologia, la struttura delle cellule si adatta alla funzione che esse devono realizzare, una versatilità ottenuta dal fatto che le cellule biologiche sono assemblate a livello molecolare. In aggiunta all'equivalente elettronico di una cellula, abbiamo dunque definito il concetto di *molecole artificiali* come piccoli elementi elettronici che possono essere messi insieme per realizzare una cellula artificiale. Trovare un equivalente elettronico delle molecole non è difficile: i già menzionati FPGA sono perfettamente adatti per assolvere questo compito. La molecola del nostro sistema è quindi l'elemento di un FPGA da noi progettato; di conseguenza, anche la struttura hardware delle nostre cellule artificiali, realizzata con una matrice bidimensionale di molecole, può dunque essere adattata all'applicazione. Un semplice meccanismo di test e di riconfigurazione permette la sostituzione di molecole morte con molecole di riserva, assicurando così un ulteriore livello di tolleranza ai difetti.

L'ispirazione biologica ci ha quindi portati a definire il nostro organismo

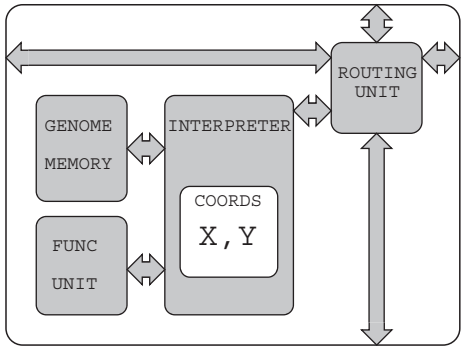


Figura 4 - Struttura generale delle nostre cellule artificiali.

artificiale come una matrice di processori di struttura identica, dal momento che ogni cellula deve poter eseguire qualsiasi parte del menoma. I processori eseguono parti diverse dello stesso programma, in funzione della propria posizione nella matrice, identificata da un sistema di coordinate cartesiane (Figura 4). Il parallelo tra le nostre cellule artificiali e il loro equivalente biologico è relativamente solido [18][11]. Come in natura, le nostre cellule sono capaci di moltiplicarsi (divisione cellulare) se un numero sufficiente di molecole sono disponibili e di differenziarsi, dato che la loro funzionalità varia a seconda della loro posizione. Esse contengono un nucleo (la memoria del genoma e il suo circuito di decodifica), un citoplasma (il processore stesso), e un insieme di meccanismi che permettono loro di scambiarsi segnali che alterano il loro comportamento. La matrice cellulare può inoltre continuare ad operare in presenza di una quantità non triviale di difetti con un processo simile a quello biologico della cicatrizzazione.

A prima vista, un tale sistema può sembrare molto inefficiente secondo la concezione tradizionale dei circuiti digitali: memorizzare una copia dell'intero genoma in ogni processore è inutile, dal momento che ogni processore ne eseguirà solo una parte. Tuttavia, accettare gli inconvenienti dell'ispirazione biologica ci permette di sfruttare i suoi vantaggi. Una delle caratteristiche più interessanti degli organismi biologici è la loro robustezza, una conseguenza di quella stessa ridondanza che sembra così inutile: visto che ogni cellula contiene una copia del genoma, essa

può teoricamente rimpiazzare qualsiasi altra cellula. Così se una o più cellule dovessero morire a causa di un trauma (per esempio, una ferita), esse possono essere rigenerate a partire da una qualsiasi cellula viva. Per analogia, se uno o più processori dovessero "morire" (in conseguenza, per esempio, dell'apparizione di un difetto nell'hardware) essi possono essere rimpiazzati da altri processori intatti nella matrice.

La ridondanza introdotta dalla presenza di copie multiple dello stesso programma fornisce quindi un supporto intrinseco all'auto-riparazione, uno degli obiettivi principali del nostro progetto: grazie a una serie di cellule di riserva, cioè cellule che sono inattive durante il funzionamento normale del circuito ma che sono identiche a tutte le altre e contengono lo stesso programma, possiamo (Figura 5) riconfigurare la matrice in modo da evitare uno o più processori difettosi. Naturalmente, come negli esseri viventi, un numero eccessivo di cellule morte causa la morte dell'intero organismo.

Inoltre, se la funzionalità di una cellula dipende dalle coordinate, l'auto-riproduzione ne risulta molto semplificata: calcolando le coordinate in modo ciclico (Figura 6), possiamo ottenere molteplici copie di un organismo con una singola copia del programma, ammettendo naturalmente che un numero sufficiente di processori sia disponibile. A seconda dell'applicazione e delle necessità dell'utilizzatore, questa caratteristica può essere utile o per ottenere una migliore performance (multipli organismi che lavorano in parallelo su dati diversi) o una più grande robustezza (le uscite di più processori operanti sugli stessi dati possono essere paragonate per verificare il loro corretto funzionamento).

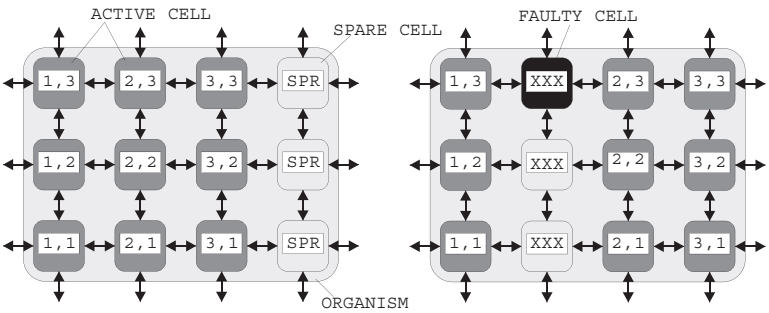


Figura 5 - Quando una delle cellule di un organismo muore, la colonna che la contiene è disattivata, e le coordinate sono ricalcolate in tutta la matrice.

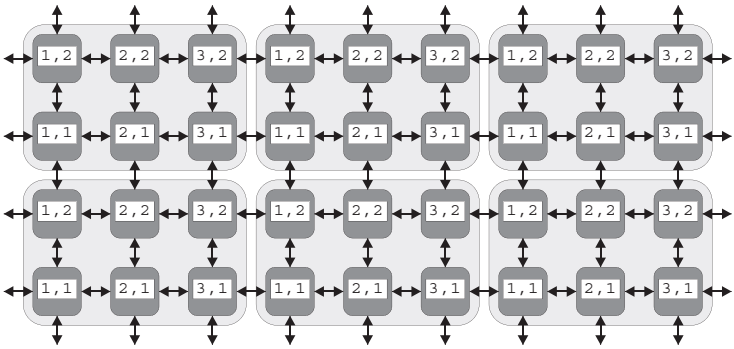


Figura 6 - Il calcolo ciclico delle coordinate genera automaticamente copie multiple dell'organismo, a condizione che un numero sufficiente di cellule siano presenti.

Intorno al genoma infatti è presente una serie di circuiti la cui funzione, simile a quella del ribosoma nelle cellule naturali, è di interpretare le istruzioni del programma genetico. È interessante notare inoltre che, per le caratteristiche di funzionamento degli FPGA, ognuna di queste parti è formato da unità simili a quelle che costituiscono il genoma, così come i ribosomi in natura sono costituiti da elementi analoghi a quelli del DNA.Il genoma è quindi una parte essenziale delle nostre cellule artificiali, ma non la sola: come nelle cellule biologiche, il genoma contiene le "istruzioni" per il funzionamento della cellula, ma queste informazioni da sole non sono sufficienti. Per dimostrare le caratteristiche dei nostri sistemi cellulari, abbiamo sviluppato una macchina (Figura 7), chiamata *BioWall* [20][21],

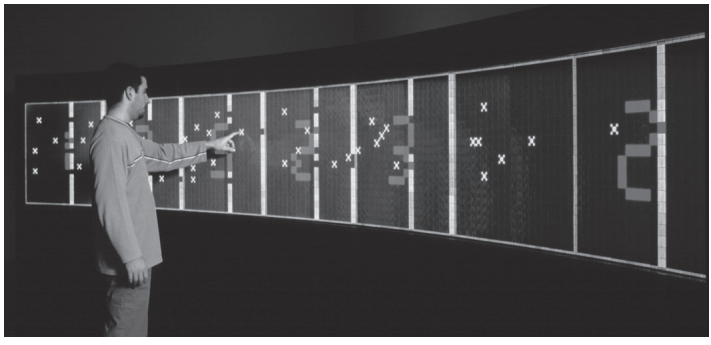


Figura 7 - Il BioWall, una macchina per il prototipaggio di sistemi celulari bio-ispirati.

che ci ha permesso di realizzare una serie di applicazioni che esibiscono le proprietà di auto-riparazione (in presenza di cellule di riserva) e di auto-replicazione (se la matrice è sufficientemente grande). Questo supporto è stato anche usato per sviluppare e verificare nuovi algoritmi di divisione e differenziazione cellulare, che vedremo nelle prossime sezioni.

4 L'algoritmo "Tom Thumb" per la divisione di cellule artificiali

La realizzazione di meccanismi di crescita nel mondo dei circuiti elettronici è un'impresa complessa. La divisione cellulare, in particolare, è un processo che in biologia si basa su meccanismi prettamente *materiali*, nel senso che operano attraverso la manipolazione e la trasformazione di oggetti fisici. Purtroppo la tecnologia attuale non permette tale manipolazione nei circuiti elettronici. I meccanismi di crescita che possono essere realizzati in questo contesto sono quindi necessariamente delle approssimazioni basate sulla manipolazione dell'*informazione* piuttosto che della materia.

Per nostra fortuna la tecnologia degli FPGA, descritta più sopra, permette una transizione relativamente diretta tra l'informazione e la materia, cioè tra la sequenza di configurazione del circuito e il circuito realizzato dall'FPGA. Questa transizione è la base del nostro approccio per realizzare una divisione cellulare artificiale: se si considera il supporto fisico (l'FPGA) come fornito a priori, le nostre cellule artificiali si riducono ad una sequenza di bit che descrive allo stesso tempo la struttura fisica della cellula e l'informazione genetica in essa contenuta.

In questa sezione, dopo aver identificato quali aspetti del processo biologico di divisione cellulare abbiamo cercato di imitare nell'ambito dei nostri circuiti, daremo la definizione di un "universo artificiale" che corrisponde essenzialmente alla configurazione di un FPGA. Illustreremo poi un algoritmo da noi sviluppato per realizzare la divisione cellulare in questo universo.

4.1 La divisione cellulare nell'universo della biologia

Prima di descrivere il nostro nuovo algoritmo per la divisione di cellule artificiali, è necessario identificare le caratteristiche principali della divisione cellulare in biologia che abbiamo usato come fonte d'ispirazione.

Per citare un libro di testo assai conosciuto ([3], p. 206):

"When a unicellular organism divides to form duplicate offspring, the division of a cell reproduces an entire organism. But cell division also enables multicellular organisms, including humans, to grow and develop from a single cell, the fertilized egg. Even after the organism is fully grown, cell division continues to function in renewal and repair, replacing cells that die from normal wear and tear or accidents. For example, dividing cells in your bone marrow continuously supply new blood cells. The reproduction of an ensemble as complex as a cell cannot occur by mere pinching in half; the cell is not like a soap bubble that simply enlarges and splits in two. Cell division involves the distribution of identical genetic material (DNA) to two daughter cells. What is most remarkable about cell division is the fidelity with which the DNA is passed along, without dilution, from one generation of cells to the next. A dividing cell duplicates its DNA, allocates the two copies to opposite ends of the cell, and only then splits into two daughter cells".

In conclusione, per quanto riguarda questo aspetto del nostro progetto, i due ruoli fondamentali della divisione cellulare che ci interessano sono:

- La costruzione di due cellule figlie per realizzare la crescita di un organismo o per riparare un organismo esistente (*traduzione* del genoma).
- La distribuzione di un insieme identico di cromosomi per creare una copia del genoma della cellula madre e "programmare" così la cellula figlia (*trascrizione* del genoma).

Ovviamente il processo di divisione cellulare in biologia va ben al di là di questa visione molto semplificata. Tuttavia, considerando le differenze fondamentali tra il mondo della biologia e quello dell'elettronica, una tale semplificazione è indispensabile.

4.2 Un universo per la divisione di cellule artificiali

Come visto in precedenza, il nostro approccio è basato su un sistema gerarchico di complessità crescente, nel quale gli organismi (sistemi di calcolo) sono realizzati da un insieme di cellule (semplici processori, tutti identici) a loro volta costituite da elementi più semplici, le molecole (gli elementi programmabili di un FPGA).

In altre parole, i nostri sistemi sono basati sulla ripetizione in ogni

cellula della stessa sequenza di configurazione molecolare. Il processo di divisione cellulare in questo contesto corrisponde quindi alla duplicazione della configurazione molecolare di una cellula per ottenere una o più cellule figlie.

Per definire in modo sistematico questo processo e per verificare la sua efficacia, è utile definire un "universo artificiale" nel quale opera un algoritmo di nostra concezione, chiamato "Tom Thumb" [13][14][15] (in italiano, Pollicino). Questo universo corrisponde alla parte dell'FPGA che contiene e propaga la configurazione ed è definito da un *contenitore*, da un *contenuto* e da un *insieme di regole*.

Il contenitore è uno spazio a due dimensioni (un limite imposto dalla tecnologia dei circuiti elettronici) diviso in righe e colonne. Ogni intersezione tra una riga e una colonna rappresenta una molecola o, più esattamente, la memoria di configurazione di una molecola. Per semplicità, la memoria di configurazione verrà rappresentata da tre settori di quattro bit ciascuno (Figura 8). Tuttavia, è importante notare che tanto il numero quanto la grandezza di tali settori possono essere aumentati senza modificare in nostri algoritmi. Il tempo nel nostro universo scorre ad intervalli discreti, detti *iterazioni* (t = -1, 0, 1 , 2, …).

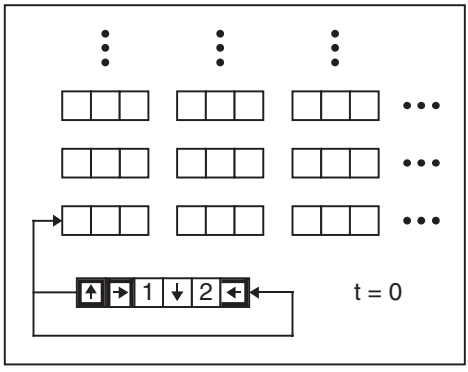


Figura 8: Il contenitore dell'universo artificiale. Ogni molecola è rappresentata dalla sua memoria di configurazione, suddivisa in tre posizioni distinte. Un registro a scorrimento esterno contiene la descrizione della cellula.

Il contenuto di questo universo è costituito da un numero finito di simboli. Ogni simbolo è rappresentato da un carattere esadecimale tra 0 ed E (il

carattere F non è usato), cioè tra 0000 e 1110 in codice binario (Figura 9). Questi simboli rappresentano la mancanza di dati (*empty data*, 0), una configurazione molecolare (*molcode data*, *M* = 1 a 7) o un'indicazione di percorso (*flag data*, *F* = 8 a E). La configurazione molecolare corrisponde

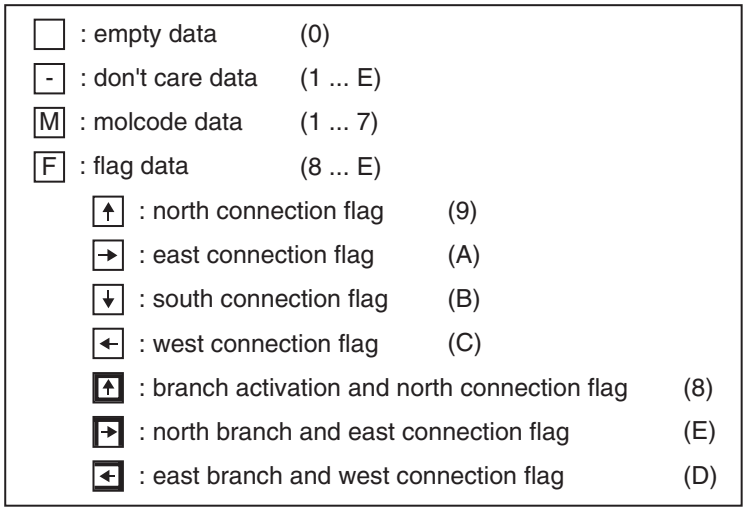


Figura 9: I possibili valori per i caratteri che costituiscono la descrizione della cellula.

alla configurazione "standard" di un FPGA e assegna alla molecola la sua funzione, mentre le indicazioni di percorso sono usate per indirizzare il flusso della configurazione all'interno della cellula. Inoltre, ogni carattere può essere *mobile* (fondo bianco nelle figure) o *fisso* (fondo grigio): i caratteri mobili si spostano in permanenza all'interno della cellula e sono usati per la divisione cellulare, mentre i caratteri fissi sono usati per determinare la funzionalità delle molecole e per indirizzare i caratteri mobili.

Infine, l'insieme di regole definisce il comportamento del contenuto all'interno dell'universo. Per l'algoritmo "Tom Thumb", sono sufficienti 9 regole per costruire le cellule e per realizzare la divisione cellulare.

4.3 L'algoritmo "Tom Thumb"

4.3.1 Condizioni iniziali

La più piccola cellula compatibile con il nostro algoritmo consiste di 4 molecole organizzate in due linee e due colonne (Figura 8). Come menzionato, ogni molecola può contenere tre caratteri esadecimali nelle sue tre posizioni di memoria e quindi ogni cellula consiste di 12 caratteri esadecimali.

La caratteristica fondamentale che contraddistingue e separa una cellula dal resto della matrice è il modo in cui le posizioni di memoria sono collegate tra loro. A differenza del sistema descritto in [18], l'algoritmo Tom Thumb non prevede la presenza di una membrana cellulare esplicita: una cellula è internamente consistente in quanto le memorie di configurazione delle molecole che la costituiscono sono collegate tra loro in modo da formare un circolo chiuso ("loop"). In questa sezione esamineremo in dettaglio la costruzione di un tale circolo in una cellula minima.

All'inizio del processo di divisione cellulare ($t = 0$) la descrizione della prima cellula è conservata all'esterno del sistema in un registro a scorrimento che ruota di una posizione da destra a sinistra ad ogni iterazione. Questo registro contiene sei caratteri esadecimali, vale a dire la metà delle posizioni di memoria della cellula.

4.3.2 Costruzione della cellula

Ad ogni iterazione il carattere all'estrema sinistra del registro esterno è inviato alla molecola in basso a sinistra del nostro circuito (Figura 8), dove viene memorizzato nella posizione sinistra della memoria molecolare . La costruzione della cellula, vale a dire la memorizzazione dei caratteri fissi e la definizione del percorso che i caratteri mobili devono seguire, si basa su tre possibili scenari che si presentano quando un carattere arriva nella posizione sinistra di una molecola (Figura 10) e che rappresentano le prime 3 regole del nostro universo artificiale:

- Se le posizioni di centro e di destra sono entrambe vuote, all'iterazione seguente i caratteri scorrono di una posizione verso la destra (*shift data*, regola 1).
- Se la posizione di destra è vuota e le posizioni di centro e di sinistra contengono entrambe delle indicazioni di percorso (*flag data F*), all'iterazione seguente i caratteri scorrono di una posizione verso la destra (*load flag*, regola 2). In questo scenario, l'indicatore *F* nella

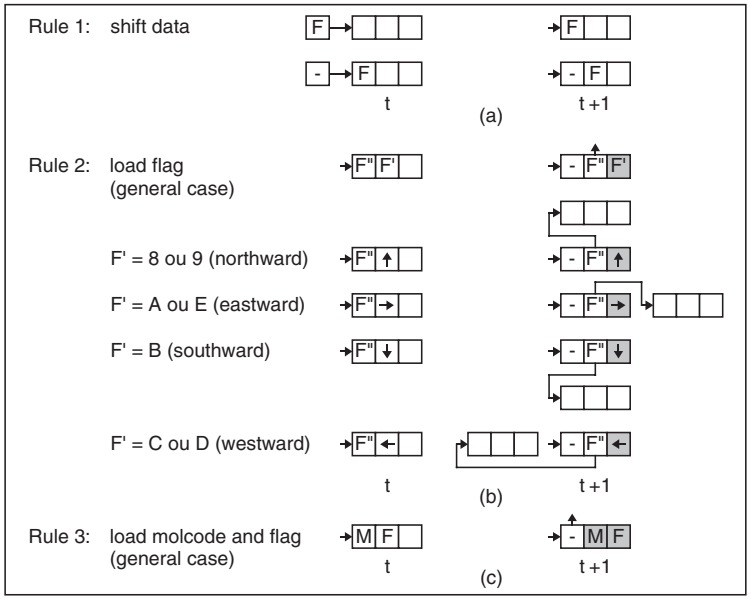


Figura 10 - Le prime tre regole per la costruzione di una cellula.

posizione di centro sarà spostato nella posizione di destra, dove sarà trasformato da carattere mobile in carattere fisso. La funzione di questo carattere sarà di definire una nuova connessione tra la posizione di memoria centrale e la molecola seguente nell'ordine di configurazione della cellula (una delle quattro molecole vicine, secondo il valore di *F*, indicato nelle figure da una freccia). Da questo momento in poi, ogni carattere che si trovi nella posizione di memoria centrale verrà inviato alla molecola seguente all'iterazione successiva.

- Se la posizione di destra è vuota e le posizioni di centro e di sinistra contengono un'indicazione di percorso (*flag data F*) e una configurazione molecolare (*molcode data M*) rispettivamente, all'iterazione seguente i caratteri scorrono di una posizione verso la destra (*load molcode and flag*, regols 3). In questo scenario, sia l'indicatore *F* che la configurazione *M* saranno trasformati da caratteri mobili in carattere fissi. La funzione del carattere *F* sarà di definire una nuova connessione tra la posizione di memoria di sinistra e la molecola seguente nell'ordine di configurazione

della cellula (una delle quattro molecole vicine, secondo il valore di *F*, indicato nelle figure da una freccia). Da questo momento in poi, ogni carattere che si trovi nella posizione di memoria di sinistra verrà inviato alla molecola seguente all'iterazione successiva. La funzione del carattere *M* sarà invece di determinare la funzionalità della molecola.

All'iterazione $t = 12$, dodici caratteri sono memorizzati nelle dodici posizioni di memoria della cellula (Figura 11) e queste ultime sono collegate tra loro

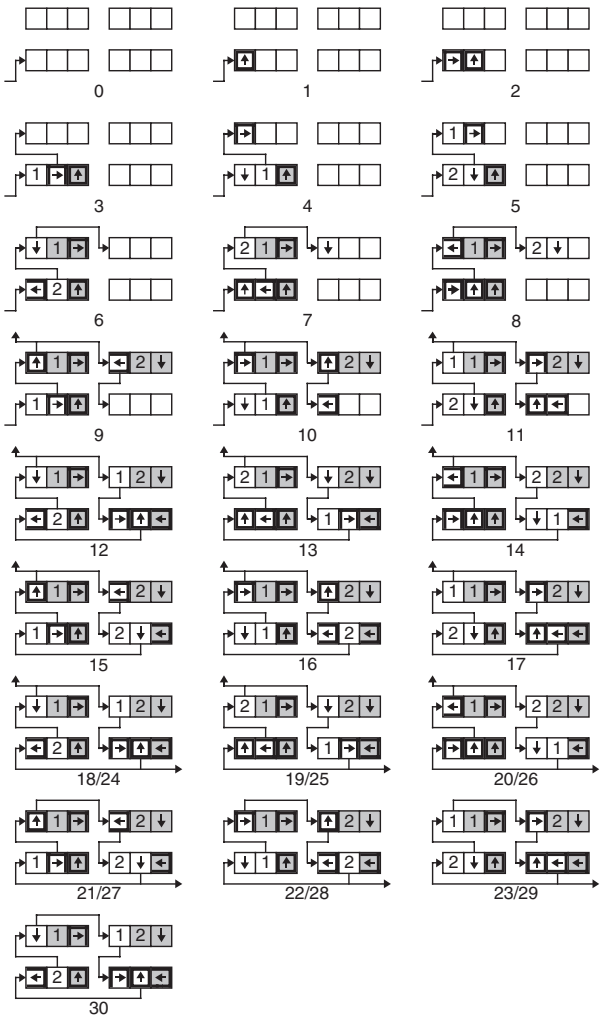


Figura 11 - Costruzione della cellula minima.

in modo tale da formare un circolo chiuso e internamente consistente. I caratteri rappresentano due copie complete della descrizione originale: una copia è fissa, mentre la seconda ruota in permanenza all'interno della cellula, seguendo le indicazioni di percorso definite dalla copia fissa (in questo senso, la copia fissa ricorda le briciole lasciate da Pollicino per ritrovare la strada, un'analogia che dà nome all'algoritmo).

In questa prima fase, l'algoritmo realizza sia la traduzione (la costruzione della cellula) che la trascrizione (la copia dell'informazione genetica e strutturale). Resta ad analizzare il meccanismo che permette a una cellula di costruirne un'altra. Come vedremo, la presenza di una copia mobile della descrizione all'interno di ogni cellula semplifica notevolmente la realizzazione di questo meccanismo.

4.3.3 La divisione cellulare

Per permettere la crescita di un organismo artificiale bidimensionale, la cellula madre deve essere capace di costruire due cellule figlie in verticale (verso nord) e in orizzontale (verso est).

All'iterazione $t = 8$ (Figura 11), si può osservare una sequenza di caratteri capace di lanciare la costruzione della cellula figlia a nord: la molecola nell'angolo in alto a sinistra è caratterizzata da due segnali specifici, vale a dire da un indicatore di biforcazione a nord (*north branch flag*, $F = E$) fisso e da un attivatore di biforcazione (*branch activation flag*, $F = 8$) pronto ad entrare nella posizione di memoria sinistra. Questa sequenza rappresenta la regola 4 del nostro universo artificiale.

All'iterazione $t = 17$, un'altra sequenza attiva a sua volta la costruzione della cellula figlia a est: la molecola in basso a destra contiene un indicatore di biforcazione a est (*east branch flag*, $F = D$) fisso e un attivatore di biforcazione (*branch activation flag*, $F = 8$) è pronto ad entrare nella sua posizione di memoria sinistra. Questa sequenza rappresenta la regola 5 del nostro universo artificiale.

4.3.4 Crescita di un organismo multicellulare

Per analizzare la crescita di un organismo, è utile osservare a livello macroscopico le interazioni delle connessioni create all'interno e all'esterno di ogni cellula.

Per le condizioni iniziali, si suppone che all'iterazione $t = -1$ una prima connessione sia costruita tra il registro a scorrimento esterno che contiene la descrizione originale (Figura 8) e la molecola in basso a sinistra nella

matrice. Ogni 6 iterazioni (per t = 5, 11, 17, …), la stessa sequenza di caratteri rilancia la creazione di questa connessione (Figura 12).

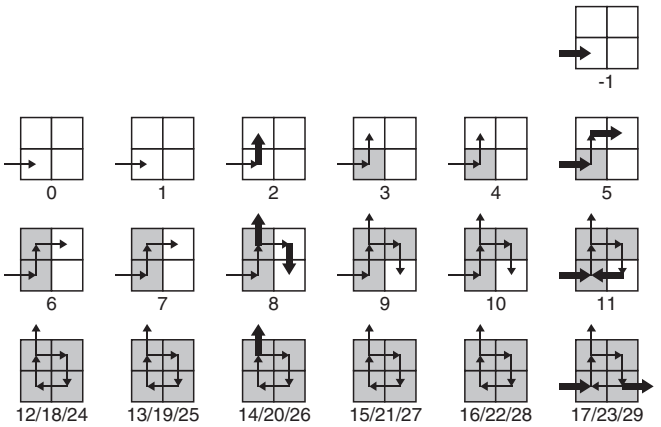


Figura 12: Attivazione delle connessioni all'interno di una cellula minima.

La costruzione della cellula è caratterizzata dall'attivazione successiva di quattro connessioni interne verso nord (t = 2), verso est (t = 5), verso sud (t = 8) e verso ovest (t = 11). Da notare la collisione tra due segnali all'iterazione t = 11, dove la priorità è data alla connessione interna.

Per effettuare la divisione della cellula madre in due cellule figlie vengono attivate una connessione esterna verso nord all'iterazione t = 8 e una connessione esterna verso est all'iterazione t = 17.

Una rappresentazione macroscopica della cellula (Figura 13) indica le diverse iterazioni nelle quali vengono attivate le connessioni interne ed esterne. Grazie a questa rappresentazione è possibile determinare il numero d'iterazioni Δtn trascorse dall'attivazione della connessione iniziale (ti = -1) all'attivazione della connessione esterna verso nord (tn = 8):

$$\Delta tn = tn - ti = 9$$

In modo analogo, è possibile determinare il numero d'iterazioni Δte trascorse dall'attivazione della connessione iniziale (ti = -1) all'attivazione della connessione esterna verso est (te = 17):

$$\Delta te = te - ti = 18$$

Infine, il numero d'iterazioni Δtc trascorse dall'attivazione della connessione

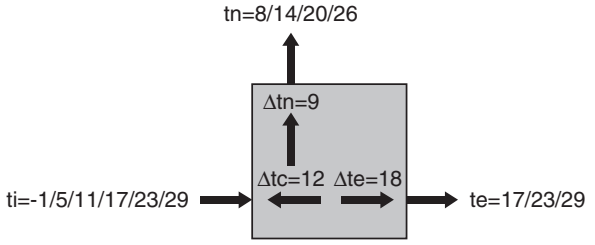


Figura 13 - Rappresentazione macroscopica di una cellula minima.

iniziale (ti = -1) all'attivazione dell'ultima connessione interna (tc = 11) è dato da:

$$\Delta tc = tc - ti = 12$$

Diviene a questo punto possibile derivare una rappresentazione macroscopica di un organismo multicellulare composto di 2x2 = 4 cellule (Figura 14) nella quale la tempistica d'attivazione di ogni connessione può essere determinata esattamente secondo le caratteristiche temporali di ogni singola cellula (Figura 15). L'inizio della costruzione di una cellula N avviene all'iterazione

$$ti(N) = \Delta tn \cdot Y + \Delta te \cdot X = 9 \cdot Y + 18 \cdot X$$

dove X e Y sono le coordinate cartesiane della cellula all'interno dell'organismo.

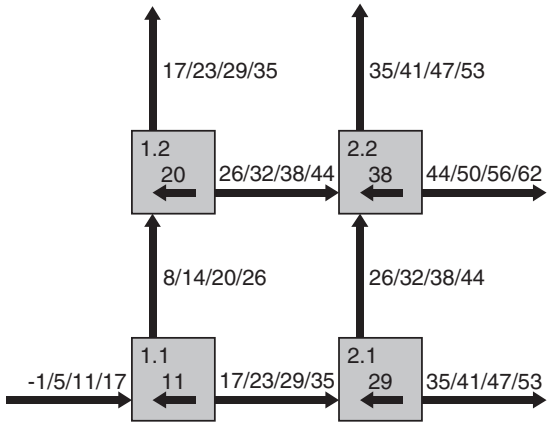


Figura 14 - Rappresentazione macroscopica di un organismo di 2x2 cellule minime.

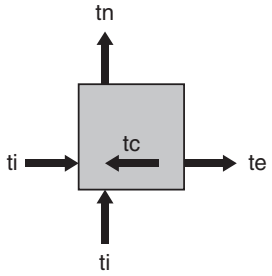


Figura 15 - Caratteristiche temporali macroscopiche della cellula minima.

In modo analogo, è possibile determinare l'iterazione esatta in cui la costruzione della cellula terminerà con la costruzione dell'ultima connessione interna:

$$tc(N) = ti(N) + \Delta tc = 9 \cdot Y + 18 \cdot X + 12$$

Come vedremo nella prossima sezione, queste formule possono essere facilmente adattate a cellule più grandi e complesse della cellula minima usata qui come esempio e permettono di determinare con esattezza la tempistica di crescita di un organismo artificiale.

4.3.5 Priorità di connessione

Per completare l'analisi della crescita di un organismo multicellulare, è necessario analizzare le priorità usate quando due o più connessioni verso la stessa molecola sono attivate simultaneamente. Per determinare tali priorità, abbiamo scelto tre modelli di crescita (Figura 14) che determinano le ultime 4 regole del nostro universo:

- Per le cellule della linea in basso (1.1 e 2.1), una collisione avviene all'iterazione $tc = ti + \Delta tc = ti + 12$ tra l'ultima connessione interna e la connessione esterna in provenienza da ovest. Come già menzionato, la priorità è data alla connessione interna (regola 6).
- Per le cellule della colonna di sinistra (1.2), ad eccezione della prima cellula d'origine, una collisione analoga avviene tra l'ultima connessione interna e la connessione esterna in provenienza da sud. Di nuovo, la priorità è data alla connessione interna (regola 7).
- Per le altre cellule (2.2), avvengono due collisioni distinte. La prima, al momento di creare la cellula all'iterazione *ti*, è tra le due connessioni esterne in provenienza da sud e da ovest (due cellule madri distinte

cercano di creare una figlia nella stessa posizione). Per questo tipo di collisioni, la connessione da sud ha priorità su quella da ovest (regola 8). La seconda collisione avviene tra l'ultima connessione interna e le due connessioni esterne. Anche qui, la connessione interna ha la priorità (regola 9).

In conclusione, le connessioni interne hanno sempre priorità su quelle esterne, permettendo ad ogni cellula di essere indipendente delle altre. Il secondo tipo di priorità assegna una direzione alla crescita dell'organismo, che si svilupperà dal basso verso l'alto (una scelta arbitraria che può essere cambiata a piacimento). È ora possibile ritornare ad una rappresentazione dettagliata di un organismo multicellulare composto di 2x2 = 4 cellule minime (Figura 16) e mostrare il suo sviluppo secondo la tempistica e l'ordine di priorità definito più sopra.

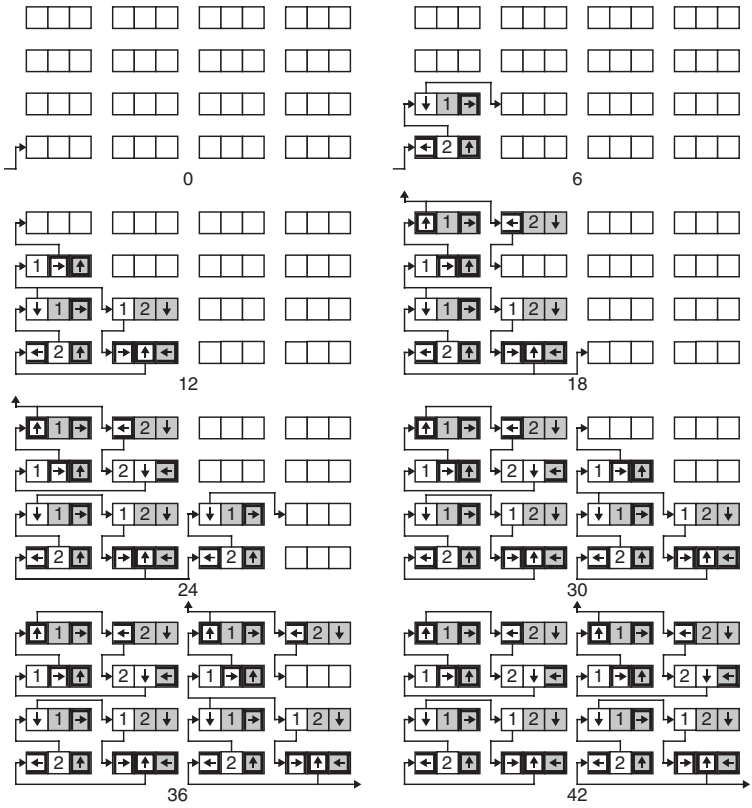


Figura 16 - Costruzione di un organismo multicellulare composto da 2x2=4 cellule minime.

5 La differenziazione cellulare in un organismo artificiale

Come per la divisione cellulare, anche nel contesto della differenziazione cellulare le differenze fondamentali tra i sistemi elettronici e i sistemi biologici necessitano una reinterpretazione dei meccanismi di base che governano questo processo in natura. In particolare, la differenziazione cellulare in biologia implica la creazione di cellule che sono strutturalmente diverse a seconda della loro funzione all'interno dell'organismo. Mentre un tale meccanismo resta possibile nel contesto dell'elettronica (la divisione cellulare in un FPGA può in teoria creare circuiti che sono strutturalmente diversi tra di loro), esso sarebbe molto complesso e probabilmente inutilizzabile all'atto pratico.

Di conseguenza, come descritto in precedenti lavori (per esempio, in [11][12]), abbiamo optato per una soluzione di compromesso, dove le cellule sono strutturalmente identiche, ma funzionalmente diverse. In questa sezione analizzeremo il nostro approccio alla differenziazione cellulare nel mondo dell'elettronica cominciando, come per la divisione, con un'analisi molto superficiale di questo processo in natura per identificare gli aspetti che abbiamo ritenuto più rilevanti per il nostro progetto. In seguito, dopo un breve sommario dell'approccio da noi scelto per realizzare la differenziazione in Embryonics, vedremo come l'algoritmo Tom Thumb può essere generalizzato a cellule di dimensioni arbitrarie. Concluderemo la sezione con un semplice esempio di differenziazione in questo contesto.

5.1 La divisione cellulare nell'universo della biologia

Il nematode *Caenorhabditis Elegans* è un piccolo verme che di recente è venuto ad occupare un posto di rilievo nella biologia molecolare dello sviluppo. Grazie alle sue caratteristiche (per esempio, è trasparente), è stato possibile ricostruire l'intera storia di ogni sua cellula durante il suo sviluppo da un uovo fertilizzato a un adulto multicellulare [26].

Due conclusioni sorprendenti sono emerse quando le informazioni provenienti da uno studio dettagliato di vermi vivi sono state combinate con studi anatomici più classici, ottenuti con l'analisi al microscopio elettronico di sezioni del verme a differenti stadi di sviluppo. Prima di tutto, ogni cellula del verme adulto è derivata dallo zigote (la prima cellula madre

dell'organismo) con una serie di divisioni cellulari (chiamata *cell lineage*) virtualmente invariabile. Inoltre, come diretta conseguenza della stabilità di questa serie di divisioni, i nematodi individuali sono copie virtualmente identiche l'uno dell'altro dal punto di vista anatomico. L'ermafrodito adulto consiste sempre di esattamente 945 cellule.

La sviluppo di un organismo può avvenire in due stili fondamentalmente diversi, *a mosaico* o *regolativo*, rivelando presumibilmente la presenza di due meccanismi di differenziazione fondamentalmente diversi. Nello sviluppo a mosaico, o sviluppo temporale (così chiamato perché l'organismo è messo insieme a partire da elementi indipendenti), la differenziazione di una cellula non dipende né dal comportamento né dalla presenza di cellule vicine: apparentemente, una serie di eventi interni alla cellula determinano le sue azioni. Tali eventi possono essere provocati dalla divisione cellulare stessa o da una specie di orologio interno azionato al momento della fertilizzazione. Nello sviluppo regolativo, o sviluppo spaziale, la differenziazione è invece parzialmente o totalmente dipendente dall'interazione tra cellule vicine.

Lo sviluppo a mosaico, governato da strette discendenze, è di gran lunga il tipo di sviluppo dominante nel *C. elegans*, mentre lo sviluppo regolativo è l'eccezione. Nella maggior parte degli altri organismi (incluso l'*Homo sapiens sapiens*), è quasi sicuramente vero il contrario. Il prezzo dello sviluppo a mosaico può essere una taglia limitata (forse solo un piccolo numero di divisioni cellulari può essere programmato così rigidamente) e una complessità modesta (le interazioni tra cellule sono forse necessarie per costruire un'anatomia più elaborata).

Dal momento che è la costruzione di organismi artificiali complessi ad interessarci, siamo stati portati a scegliere il modello di sviluppo regolativo come fonte d'ispirazione per il nostro progetto.

5.2 La differenziazione cellulare in Embryonics

Come abbiamo visto in precedenza, le cellule artificiali del nostro progetto sono piccoli processori che contengono al loro interno una copia del genoma dell'organismo sotto forma di un programma.

In biologia, tuttavia, nessuna cellula usa l'intero genoma, ma accede solo alla porzione necessaria per la propria funzione. In modo simile, in Embryonics nessun processore eseguirà tutte le istruzioni del programma,

ma userà la propria posizione all'interno della matrice per identificare quale parte del programma è rilevante per la sua funzione ed eseguirla.

Tenendo presente le necessità e la struttura degli organismi artificiali, possiamo ora definire le caratteristiche essenziali della nostra cellula artificiale. A livello hardware, tutte le cellule devono essere identiche: dal momento che i nostri organismi devono poter eseguire in modo efficace diverse applicazioni, non possiamo fissare a priori la funzionalità delle nostre cellule. Inoltre, devono poter memorizzare il programma genetico con un meccanismo di accesso dipendente dalle coordinate.

La struttura hardware della nostra cellula dovrà dunque poter eseguire una qualsiasi funzione, e sarà compito del genoma e del sistema di coordinate scegliere quale parte del circuito dovrà essere attivata a un dato momento.

Ovviamente, esistono molti possibili approcci alla definizione di una cellula artificiale, ma se dobbiamo mantenere l'analogia con la biologia, essa dovrà necessariamente contenere (Figura 4):

- Una memoria per il genoma, la cui taglia dipende dall'applicazione.
- Un sistema di coordinate [X,Y] che permetta alla cellula di identificare la propria posizione all'interno della matrice, e quindi la propria funzione.
- Un decodificatore per leggere ed eseguire il genoma.
- Un'unità funzionale per il trattamento dei dati. A seconda dell'applicazione, essa può contenere una varietà di elementi logici, da un semplice registro a un'unità aritmetica completa e oltre.
- Una serie di connessioni comandate da un'unità di scambio.

Appare ovvio che anche la più semplice delle cellule che possono corrispondere a questa descrizione sarà molto più complessa della cellula minima descritta per illustrare la divisione cellulare. È quindi fondamentale che il nostro algoritmo Tom Thumb per la divisione cellulare sia estensibile a cellule di grandezza arbitraria.

5.3 Generalizzazione delle cellule artificiali

Per rispondere alle necessità del progetto, la cellula minima presentata in precedenza (Figura 11) può in effetti essere direttamente estesa per ottenere cellule di taglia più grande. Per esempio, le cellule della Figura 17 rappresentano un'estensione a 8 e 16 molecole. Queste tre cellule,

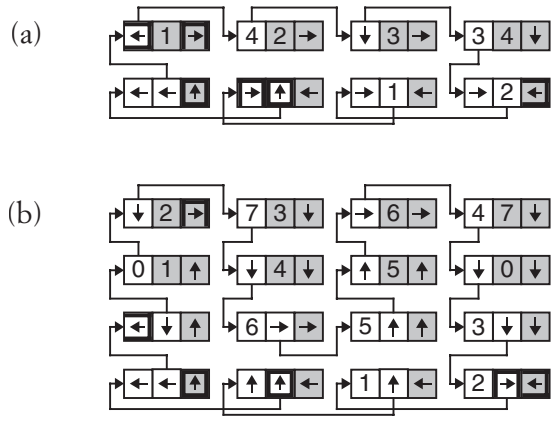


Figura 17 - Estensioni canoniche della cellula a (a) 8 e (b) 16 molecole.

inoltre, hanno la caratteristica di essere *canoniche*, dal momento che possono essere suddivise in due mezze cellule della stessa taglia, secondo l'organizzazione seguente:

- La metà superiore della cellula consiste di *molecole attive* che contengono un codice molecolare fisso insieme alle indicazioni di percorso. Il codice molecolare contiene le indicazioni sull'operazione della molecola e di conseguenza solo questa metà sarà effettivamente attiva all'interno della cellula. Come vedremo, questo codice molecolare può essere usato direttamente per rappresentare l'output della cellula (come nell'esempio più sotto) o indirettamente per configurare la funzionalità di un elemento di FPGA (come nei sistemi normalmente usati per il nostro progetto Embryonics [11][19]).
- La metà inferiore della cellula consiste di *molecole passive* che contengono unicamente delle indicazioni di percorso fisse, mentre il resto del registro di configurazione contiene solo dati mobili (la descrizione della cellula usata per la divisione cellulare).

Se C è il numero di colonne di molecole all'interno della cellula e R il numero di linee di molecole nella mezza cellula, è semplice derivare le seguenti relazioni:

- Il numero totale di molecole M in una cellula è:

$$M = 2 \cdot C \cdot R$$

- Nel caso generale (vale a dire nel caso in cui il registro di configurazione abbia una grandezza superiore alle 3 posizioni dell'esempio minimo),

una molecola attiva contiene un indicatore di percorso fisso nella posizione di destra, un carattere mobile nella posizione di sinistra e D codici molecolari tra le due. Di conseguenza la lunghezza H del suo registro di configurazione è pari a:

$$H = D + 2$$

- Il numero totale T di caratteri esadecimali in una cellula è quindi:

$$T = 2 \cdot C \cdot R \cdot (D + 2)$$

- La lunghezza L della descrizione della cellula è la metà di T, dal momento che ogni cellula contiene due copie della descrizione (una fissa e l'altra mobile):

$$L = C \cdot R \cdot (D + 2)$$

Il *periodo* della cellula, cioè il tempo necessario perché la copia mobile della descrizione compia un giro completo all'interno della cellula, è quindi di L iterazioni.

Un'analisi attenta del meccanismo di divisione cellulare permette di derivare le seguenti relazioni per Δtn, Δte e Δtc:

$$\Delta tn = L + R \cdot (D + 1) + R = (C + 1) \cdot R \cdot (D + 2)$$
$$\Delta te = 3 \cdot L = 6 \cdot R \cdot (D + 2) \qquad \text{se } C = 2$$
$$\Delta te = 2 \cdot L - (C - 2) \cdot (D + 1) \qquad \text{se } C > 2$$
$$\Delta tc = T = 2 \cdot L = 2 \cdot C \cdot R \cdot (D + 2)$$

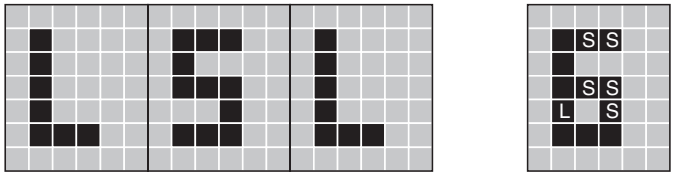
Queste generalizzazioni permettono di estendere le formule derivate nella sezione precedente al caso di cellule non minime, un'estensione necessaria per poter disporre di una funzionalità ragionevole per le nostre cellule artificiali. La tempistica di sviluppo di qualsiasi organismo può così essere determinata con precisione, a condizione naturalmente che la divisione cellulare avvenga in modo automatico fino a riempire lo spazio disponibile.

Per ottenere un meccanismo di divisione e differenziazione cellulare più versatile nei nostri sistemi artificiali, è infatti possibile introdurre nel nostro sistema un meccanismo di regolazione della divisione: piuttosto che replicare automaticamente le cellule fino a riempire lo spazio disponibile, un semplice meccanismo integrato alla cellula stessa può avviare o inibire la divisione e di conseguenza controllare lo sviluppo dell'organismo. In questo caso, sia lo sviluppo a mosaico sia quello regolativo diventano realizzabili, attivando la divisione unicamente in base alle istruzioni contenute nel genoma nel primo caso o in seguito a interazioni con le cellule vicine nel secondo.

5.4 Un esempio di differenziazione cellulare

Anche se lo scopo finale del nostro progetto è lo sviluppo di macchine complesse, per illustrare i meccanismi di base del nostro approccio alla differenziazione cellulare useremo un esempio estremamente semplificato: la funzione del nostro organismo sarà quella di visualizzare la sigla LSL del nostro Logic Systems Laboratory.

La macchina per visualizzare questa sigla può essere vista come un organismo artificiale unidimensionale composto di tre cellule (Figura 18a). Ogni cellula è identificata da una coordinata X tra 1 e 3 (tra 01 e 11 in codice binario). Quando $X = 1$ o $X = 3$ la cellula deve visualizzare il carattere L, mentre quando $X = 2$ la cellula deve visualizzare il carattere S. Una cellula totipotente, cioè capace di visualizzare tanto il carattere L quanto il carattere S, richiede 6x7 = 42 molecole (Figura 18b), di cui 36 sono invariabili, 5 sono usate solo per visualizzare il carattere S e una è usata solo per L. La cellula finale comprende inoltre un incrementatore (un circuito che incrementa di 1 il valore che riceve come input) modulo 3. L'operazione di questo circuito corrisponde alla tavola di verità della Figura 18c ed è realizzata dal circuito logico della Figura 18d. Come si può



Character	X	X1 X0	X+	X1+ X0+
L	1	0 1	2	1 0
S	2	1 0	3	1 1
L	3	1 1	1	0 1

(c)

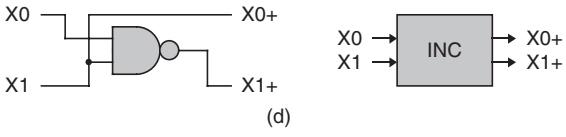


Figura 18 - (a) L'organismo completo; (b) La cellula totipotente; (c) La tavola di verità del circuito (d) che determina la differenziazione in funzione della coordinata X.

vedere nella tavola di verità, il valore della variabile binaria *X*0 è sufficiente per distinguere le cellule che devono visualizzare il carattere L (*X*0 = 1) da quelle che devono visualizzare il carattere S (*X*0 = 0).

Queste specifiche sono sufficienti per definire l'architettura finale della cellula totipotente (Figura 19). Secondo l'algoritmo Tom Thumb, la metà delle 42 molecole della cellula sono passive (*G*) e di conseguenza non hanno funzionalità a parte quella di conservare una copia della descrizione della cellula. Le altre molecole sono attive e possono essere divise in 6 categorie a seconda della loro funzione:

1. Propagazione orizzontale della coordinata *X*.
2. Incremento modulo 3.
3. Distribuzione verticale della variabile logica *X*0.
4. Visualizzazione permanente per entrambi i caratteri L e S.
5. Visualizzazione del carattere S.
6. Visualizzazione del carattere L.

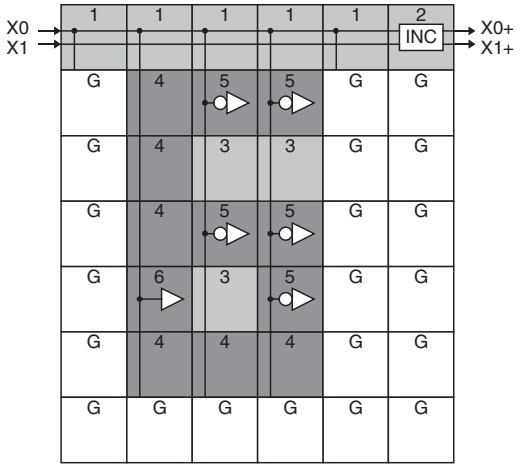


Figura 19 - Architettura della cellula totipotente.

Per quanto riguarda la configurazione e la divisione cellulare, la cellula finale consiste di 6x7 = 42 molecole connesse tra loro come mostrato nella Figura 20a: dal basso verso l'alto nelle colonne pari, dall'alto verso il basso nelle colonne dispari, mentre la linea in basso è usata per chiudere il circolo. A partire da questa struttura, è possibile definire tutti gli indicatori di percorso fissi nelle posizioni di destra delle memorie (in grigio nella

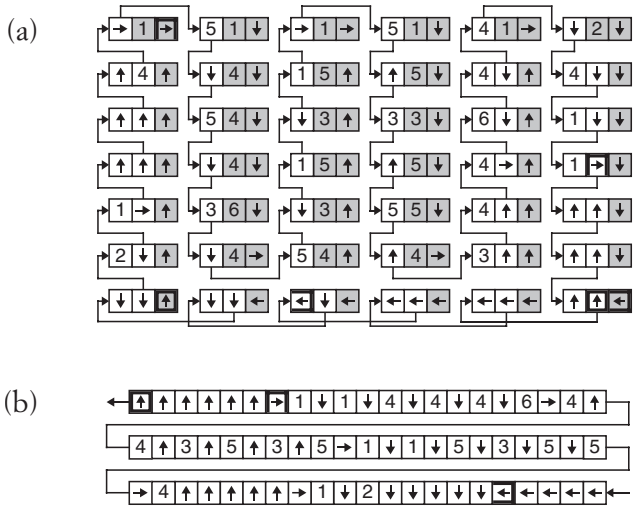


Figura 20 - (a) Configurazione e (b) descrizione della cellula.

figura), senza dimenticare l'attivatore di biforcazione nella molecola in basso nella prima colonna ($F = 8$), l'indicatore di biforcazione a nord nella molecola in alto nella prima colonna ($F = E$) e l'indicatore di biforcazione ad est nella molecola in basso nell'ultima colonna ($F = D$).

Secondo il nostro algoritmo, le 21 molecole attive (Figura 19) occupano una posizione fissa e predeterminata nella cellula totipotente (Figura 20a). Le 21 molecole passive sono usate per conservare e circolare una copia della descrizione (Figura 20b) della cellula. Per motivi di ordine pratico, abbiamo neutralizzato la visualizzazione delle cellule passive, le cui uscite sono dunque fissate al valore logico 0.

Per verificare il nostro sistema abbiamo inserito una molecola in ognuno dei 2,000 elementi programmabili del BioWall (Figura 21). Il comportamento del sistema corrisponde perfettamente al modello teorico e abbiamo potuto realizzare la crescita del nostro organismo artificiale, seguita dalla sua riproduzione in due dimensioni.

È opportuno rilevare che, in questo semplice esempio, non si può identificare un vero e proprio genoma: le cellule sono in questo caso tanto triviali da non necessitare la presenza di un programma e della logica necessaria per interpretare un set di istruzioni. L'algoritmo Tom Thumb, però, è perfettamente compatibile con *MuxTree* [11][19],

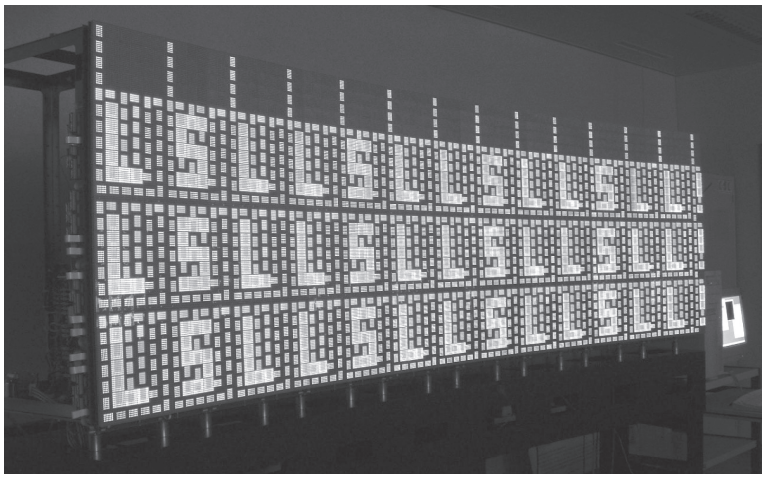


Figura 20 - (a) Configurazione e (b) descrizione della cellula.

l'FPGA sviluppato nel contesto del nostro progetto per realizzare la funzione di molecola nella gerarchia di complessità. Di conseguenza, tutte le applicazioni e le strutture cellulari sviluppate all'interno del progetto Embryonics possono essere immediatamente adattate al nuovo algoritmo per approfittare dell'aumentata versatilità della divisione cellulare.

6 Conclusione

L'introduzione di un meccanismo di divisione cellulare che permettesse alle nostre cellule di replicarsi in modo autonomo era uno degli obiettivi principali del progetto Embryonics. Nel nostro precedente articolo su questa rivista [18] abbiamo cercato di mostrare che le caratteristiche delle nostre cellule artificiali conferiscono loro una certa somiglianza con le cellule biologiche. Per esempio, abbiamo mostrato come il dogma centrale della biologia molecolare [4] e la trinità *genotipo-ribotipo-fenotipo* proposta da Barbieri [1] siano applicabili ai nostri sistemi, pur con le dovute alterazioni richieste dalle differenze fondamentali tra i due substrati [10].

Altri paralleli possono essere fatti tra le cellule di Embryonics e le cellule biologiche, come per esempio l'analogia tra il calcolo delle coordinate e

la funzione dei geni di coordinata [26] e dei geni di selezione [5] negli organismi. Tuttavia, come da noi precisato, molte di queste analogie erano rese meno verosimili dalla necessità di pilotare la divisione cellulare e quindi lo sviluppo dell'organismo tramite un controllore esterno al circuito.

L'algoritmo Tom Thumb è la soluzione che proponiamo al problema di introdurre una divisione cellulare controllata esclusivamente dalle cellule stesse. Questa soluzione è ovviamente molto diversa dai meccanismi di divisione nelle cellule biologiche. Tuttavia, in aggiunta a qualche somiglianza che si potrebbe definire "logica" (la presenza di una doppia descrizione della cellula al suo interno ricorda la doppia elica del DNA ed il suo sdoppiamento al momento della divisione è un meccanismo comune tra la biologia e l'elettronica), l'osservazione cruciale è che i due meccanismi hanno uno scopo e un modo di procedere assai simili. Per ottenere la crescita di un organismo, è necessario infatti che le sue cellule si dividano e che questa divisione non sia aleatoria, ma piuttosto finalizzata alla costruzione di strutture ben definite. Il nostro algoritmo permette alle nostre cellule artificiali di ottenere appunto questa funzionalità e la possibilità di controllare la crescita del sistema all'interno delle cellule stesse a sua volta permette al sistema di implementare modelli di sviluppo di arbitraria complessità e stile (a mosaico o regolativi).

Un certo numero di futuri sviluppi sono in corso intorno al progetto Embryonics. Da un lato, l'algoritmo Tom Thumb sarà modificato per migliorare la sua efficienza e per permettere la divisione cellulare su substrati difettosi, una necessità assoluta per circuiti di grande taglia. Dall'altro, stiamo sviluppando nuove versioni delle nostre molecole [23] e delle nostre cellule [22] per migliorare la loro capacità di eseguire calcoli complessi. Anche se questi miglioramenti non sono di per sé destinati ad aumentare la somiglianza tra i nostri sistemi e i sistemi biologici, essi sono tuttavia una tappa fondamentale per poter sviluppare organismi complessi e per eventualmente introdurre altri meccanismi informatici ispirati dalla biologia, quali ad esempio le reti neurali.

BIBLIOGRAFIA

[1] M. Barbieri [1985]. *La Teoria Semantica dell'Evoluzione*. Ed. Boringhieri, Torino, Italy, 1985. Pubblicato in inglese come *The Semantic Theory of Evolution*. Harwood Academic Publishers, Chur, Switzerland, 1985.

[2] S.D. Brown, R.J. Francis, J. Rose, Z.G. Vranesic [1992]. *Field-Programmable Gate Arrays*. Kluwer Academic Publishers, Boston, 1992.

[3] N.A. Campbell, J.B. Reece, L.G. Mitchell [1999]. *Biology, 5th Edition*. Benjamin/Cummings, Menlo Park, 1999.

[4] F.H.C. Crick [1958]. *On protein synthesis*. Symposia of the Society for Experimental Biology, 12:548-555, 1958.

[5] S.F. Gilbert [1991]. *Developmental Biology*. Sinauer Associates, Inc., MA, 3rd ed., 1991.

[6] J.P. Hayes [1993]. Introduction to Digital Logic Design. Addison-Wesley, Reading, MA, 1993.

[7] T. Higuchi, M. Iwata, I. Kajitani, H. Iba, Y. Hirao, T. Furuya, B. Manderick [1996]. "Evolvable Hardware and its Application to Pattern Recognition and Fault-Tolerant Systems". In E. Sanchez, M. Tomassini, eds., *Towards Evolvable Hardware*, LNCS Springer, Berlin, 1996, pp. 118-135.

[8] J.R. Koza, F.H. Bennett III, D. Andre, M.A. Keane [1996]. "Automated {WYWIWYG} Design of Both the Topology and Component Values of Electrical Circuits Using Genetic Programming". In *Genetic Programming 1996: Proceedings of the First Annual Conference*, The MIT Press, Cambridge, MA, 1996, pp.123-131.

[9] D. Mange, M. Goeke, D. Madon, A. Stauffer, G. Tempesti, S. Durand [1996]. "Embryonics: A New Family of Coarse-Grained Field-Programmable Gate Array with Self-Repair and Self-Reproducing Properties". In E. Sanchez, M. Tomassini, eds., *Towards Evolvable Hardware*, Lecture Notes in Computer Science, Springer, Berlin, 1996, pp. 197-220.

[10] D. Mange, M. Sipper [1998]. "Von Neumann's Quintessential Message: Genotype + Ribotype = Phenotype". *Artificial Life Journal*. Accepted.

[11] D. Mange, M. Tomassini, eds. [1998]. *Bio-inspired Computing Machines: Towards Novel Computational Architectures*. Presses Polytechniques et Universitaires Romandes, Lausanne, Switzerland, 1998.

[12] D. Mange, M. Sipper, A. Stauffer, G. Tempesti [2000]. "Toward Robust Integrated Circuits: The Embryonics Approach". In *Proceedings of the IEEE*, vol.88, n.4 (2000), pp.516-541.

[13] D. Mange, A. Stauffer, E. Petraglio, G. Tempesti [2004]. "A Macroscopic View of Self-Replication". In *Proceedings of the IEEE*, vol.92, n.12 (2004), pp.1929-1945.

[14] D. Mange, A. Stauffer, E. Petraglio, G. Tempesti [2004]. "Embryonic Machines that Divide and Differentiate" In *Proc. 1st Int. Workshop on Biologically-Inspired Approaches to Advanced Information Technology (Bio-ADIT 2004)*, LNCS 3141, Springer-Verlag, Berlin, 2004, pp.201-216.

[15] D. Mange, A. Stauffer, L. Peparolo, G. Tempesti [2004]. "Self-replicating loop with universal construction". In *Physica D*, v.191, n.1 (2004), pp.178-192.

[16] C. Maxfield [1995]. *Bebop to the Boolean Boogie*. HighText Publications, Solana beach, CA, 1995.

[17] E. Sanchez, D. Mange, M. Sipper, M. Tomassini, A. Perez-Uribe, A. Stauffer [1997]. "Phylogeny, Ontogeny, and Epigenesis: Three Sources of Biological Inspiration for Softening Hardware". In T. Higuchi, M. Iwata, W. Liu, eds., *Proc. 1st Int. Conference on Evolvable Systems: From Biology to Hardware (ICES96)*, Lecture Notes in Computer Science, vol. 1259, Springer-Verlag, Berlin, 1997, pp. 35-54.

[18] G. Tempesti, D. Mange, A. Stauffer [1998]. "Il progetto Embryonics: una macchina fatta di cellule artificiali". In *Systema Naturae, Annali di biologia teorica*, n.1 (1998), pp.41-82.

[19] G. Tempesti [1998]. *A Self-Repairing Multiplexer-Based FPGA Inspired by Biological Processes*. Ph.D. Thesis, Swiss Federal Institute of Technology, Lausanne, 1998.

[20] G. Tempesti, D. Mange, A. Stauffer, C. Teuscher [2002]. "The BioWall: an Electronic Tissue for Prototyping Bio-Inspired Systems". In *Proc. 2002 NASA/DoD Conf. on Evolvable Hardware (EH2002)*, IEEE Computer Society press, Los Alamitos, CA, 2002, pp.221-230.

[21] G. Tempesti, C. Teuscher [2003]. "Biology Goes Digital". In *Xcell Journal*, vol.47 (2003), pp.40-45.

[22] G. Tempesti [2004]. "Processor Architectures for Ontogenesis". In *Proc. 2004 NASA/DoD Conference on Evolvable Hardware (EH 2004)*, IEEE Computer Society Press, Los Alamitos, CA, pp.269-272

[23] Y. Thoma, G. Tempesti, E. Sanchez, J.M. Moreno Arostegui [2003]. "POEtic: An Electronic Tissue for Bio-Inspired Cellular Applications". In *Proc. 5th Int. Workshop on Information Proceesing in Cells and Tissues* (*IPCAT* 2003), pp.199-214

[24] A. Thompson [1996]. "Silicon Evolution". In *Genetic Programming 1996: Proceedings of the First Annual Conference*, The MIT Press, Cambridge, MA, 1996, pp. 444-452.

[25] S. Trimberger, ed. [1994]. *Field-Programmable Gate Array Technology*. Kluwer Academic Publishers, Boston, 1994.

[26] J.D. Watson, N.H. Hopkins, J.W. Roberts, J. Argetsinger Steitz, A.M. Weiner [1987]. *Molecular Biology of the Gene*. Benjamin/Cummings, Menlo Park, CA, 4th edition, 1987.