II.TECNICHE DI PROGRAMMAZIONE MULTILIVELLO

1.La programmazione

floating-gate prende il nome di programmazione. Nella programmazione standard a due livelli le operazioni di

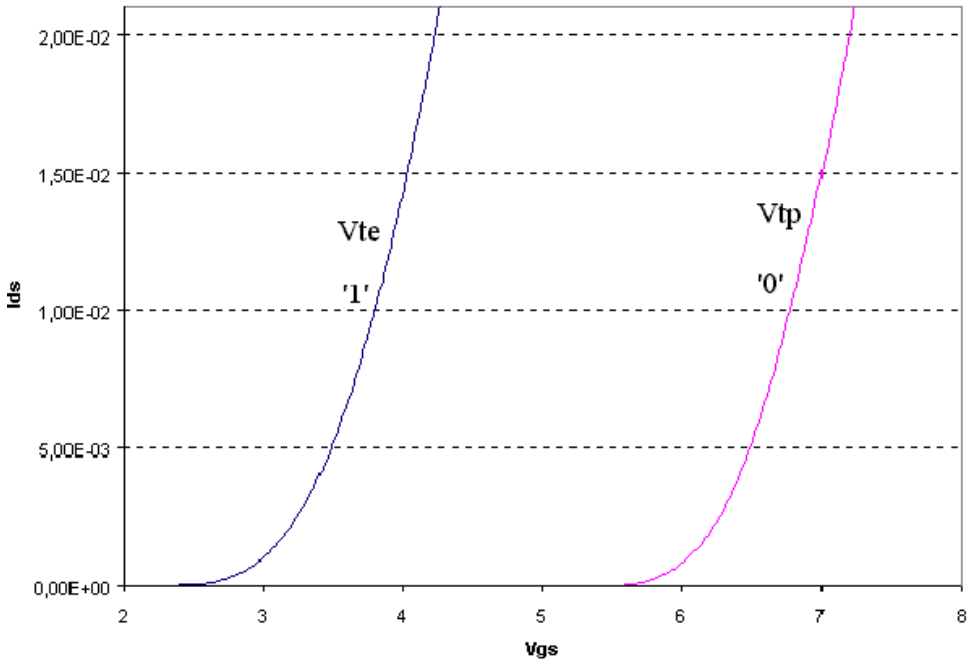

Fig.II.1.a.Caratteristica elettrica

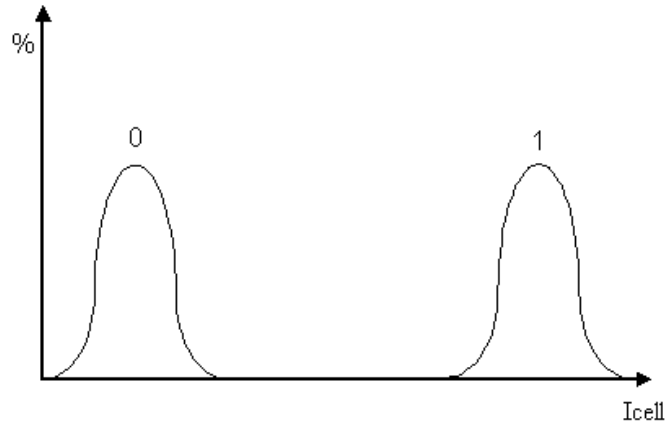

Fig.II.1.b. Distribuzioni delle correnti

programmazione e lettura risultano molto semplici; infatti la programmazione richiede soltanto di portare la distribuzione, corrispondente al livello logico '0'(Fig.II.1.a), delle correnti di drain il più possibile a zero. Per la lettura basta scegliere una tensione di gate tra i due valori di soglia relativi allo stato *erasato* e stato *programmato* della cella, questa tensione è scelta uguale a 5V(Fig.II.1.b).

Nella programmazione multilivello si hanno quattro differenti distribuzioni di corrente(Fig.II.2), per cui è necessario controllare con estrema precisione la carica iniettata nella *floating-gate,* perché quest'ultima determina lo stato di programmazione della cella. Per un corretto funzionamento del dispositivo, è indispensabile evitare la sovrapposizione delle distribuzioni delle correnti, perché nel tempo, a causa di disturbi e fenomeni indesiderati, la *floating-gate* può perdere carica provocando così lo spostamento delle distribuzioni. Inoltre, nell'operazione di lettura, occorre mantenere un certo margine tra le distribuzioni delle correnti, in modo tale che il *sense amplifier* possa distinguere le diverse correnti.

I metodi di programmazione sono sostanzialmente due:

- programmazione algoritmica;

- programmazione analogica.

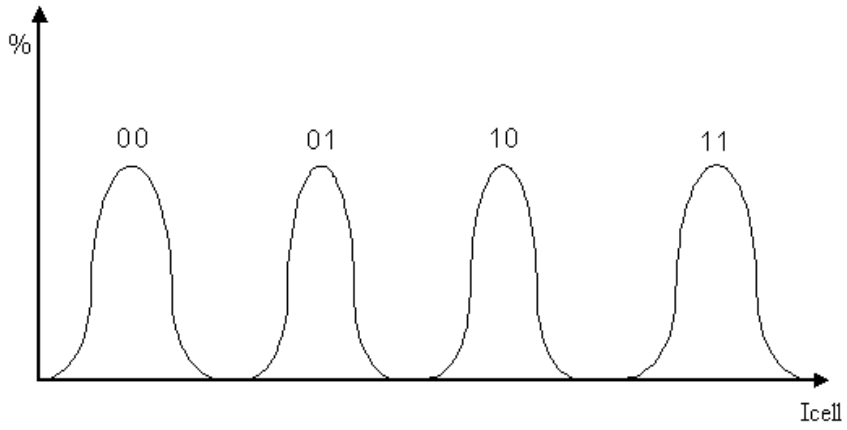

Fig.II.2.Distribuzioni di corrente per quattro livelli di programmazione

Entrambe le tecniche sono valide ai fini della programmazione e nel seguito verranno descritte in dettaglio.

2.Tecniche di programmazione algoritmica

L'obiettivo principale della programmazione di celle di memoria multi-livello è la capacità di controllare in modo molto preciso la carica immagazzinata nella *floating-gate.*

Gli algoritmi di programmazione mirano all'ottenimento di distribuzioni di tensioni di soglia strette e adeguatamente separate fra loro. A questo aspetto se ne aggiungono altri non meno importanti, specie in ambito multi-livello, quali la precisione e i tempi di programmazione.

La scelta del metodo di programmazione riveste una notevole importanza in tutti gli aspetti delle memorie multi-livello. La programmazione per *Channel Hot Electron* (CHE) presenta alcuni vantaggi rispetto alla programmazione per effetto

Tunnel Fowler-Nordheim (FN), perché è più veloce, più affidabile e meno sensibile ai parametri di processo. A favore, la programmazione per FN dal lato di una giunzione di drain o di source, offre il vantaggio che le elevate tensioni necessarie possono essere generate sul chip mediante pompe di carica in quanto, a differenza del meccanismo CHE, sono richieste basse correnti di programmazione per la singola cella. Per questo è possibile programmare in parallelo un numero sufficiente di celle.

Per avere il maggior numero di livelli per cella devono essere soddisfatti requisiti molto più stringenti rispetto a quelli richiesti per memorie convenzionali bilivello: in particolare la dispersione statistica dei livelli diventa sempre più critica, in quanto sono richieste distribuzioni molto strette.

Per una programmazione accurata delle celle conviene usare un *algoritmo adattativo* in cui ogni bit della parola sia *programmato* e *verificato* separatamente. Attualmente, la soluzione più utilizzata è la cosiddetta tecnica *Program & Verify*. L'approccio consiste nell'applicazione di opportune tensioni ai terminali della cella (fase di programmazione) e nella successiva lettura del contenuto (fase di verifica). La programmazione viene controllata applicando impulsi di tensione adeguatamente controllati a uno o più terminali. In particolare la regolazione della tensione sul *control-gate* è da preferire rispetto a quella sulla tensione di *drain*.

La programmazione con verifica comporta un'occupazione di area non trascurabile, per questo trova principalmente applicazione per memorie *stand-alone*, dove la matrice di memoria costituisce la parte più consistente dell'intero sistema. La costante crescita del mercato delle memorie *embedded*, cioè integrate all'interno di un sistema per applicazioni specifiche, ha motivato l'interesse per i sistemi di programmazione senza verifica. Gli schemi di programmazione non verificati

peggiorano le prestazioni di durata e producono distribuzioni più larghe rispetto a quelle con verifica.

Verranno di seguito illustrate alcune tecniche di programmazione algoritmiche attualmente più utilizzate per le memorie flash multi-livello.

Programmazione con tensione di gate

È stato dimostrato sperimentalmente, e giustificato teoricamente, che è possibile stabilire una relazione lineare fra la variazione della tensione di gate (Vg) e il salto della tensione di soglia (Vt) per fissati valori di Vd e Vs. Una tecnica per programmare una cella è pertanto, quella di incrementare gradualmente la tensione di gate, in modo che la tensione applicata in un impulso di programmazione differisca da quella dell'impulso precedente per una quantità costante delta Vg. La tensione applicata al *control gate* della cella assume la tipica forma a gradinata avente un passo costante. Il valore di soglia prefissato viene raggiunto anche dalle celle lente, applicando semplicemente alcuni passi in più di programmazione con tensione di *control gate* più elevata.

Il risultato, dell'uso dell'algoritmo della tensione di gate a gradinata, è che la larghezza della distribuzione delle tensioni di soglia ottenuta equivale idealmente alla variazione delta Vg della tensione applica al *control gate*, a meno dell'errore dovuto alla non idealità dei componenti del sistema, in particolare all'inaccuratezza della lettura.

Programmazione con tensione di drain

Un'altra tecnica di programmazione, per memorie multi-livello, consiste nell'applicazione di una tensione controllata al drain mantenendo fissa a un valore opportuno la tensione al *control gate* della cella.

3.Algoritmo di programmazione

In questo paragrafo viene presentato un algoritmo di programmazione con tensione di gate, nel caso di memoria multi-livello con quattro livelli di programmazione ('00' , '01' , '10' , '11').

La programmazione algoritmica (con tensione di gate) consiste nell'inviare una successione di impulsi di durata e ampiezza regolabili, alla cella di memoria da programmare. Dopo ciascun impulso è necessario verificare lo stato di programmazione della cella.

Nella programmazione standard a due livelli questa tecnica di programmazione risulta efficace e veloce, in quanto basta verificare soltanto se è stata superata una certa tensione di soglia.

La programmazione multilivello richiede, per i livelli intermedi '01' e '10', di verificare sia vincoli inferiori che superiori, perché è di estrema importanza conoscere con esattezza la quantità di carica immagazzinata nella *floating-gate.*

Per controllare con precisione la carica immagazzinata nella *floating-gate* durante la programmazione, è necessario avere impulsi brevi di programmazione e questo comporta un aumento del tempo complessivo per l'operazione di scrittura della cella di memoria.

L’operazione di verifica avviene necessariamente in successione ad ogni impulso di programmazione con tensioni di polarizzazione diverse- rispetto alla scrittura.

Un algoritmo di programmazione è mostrato in Fig.II.3:

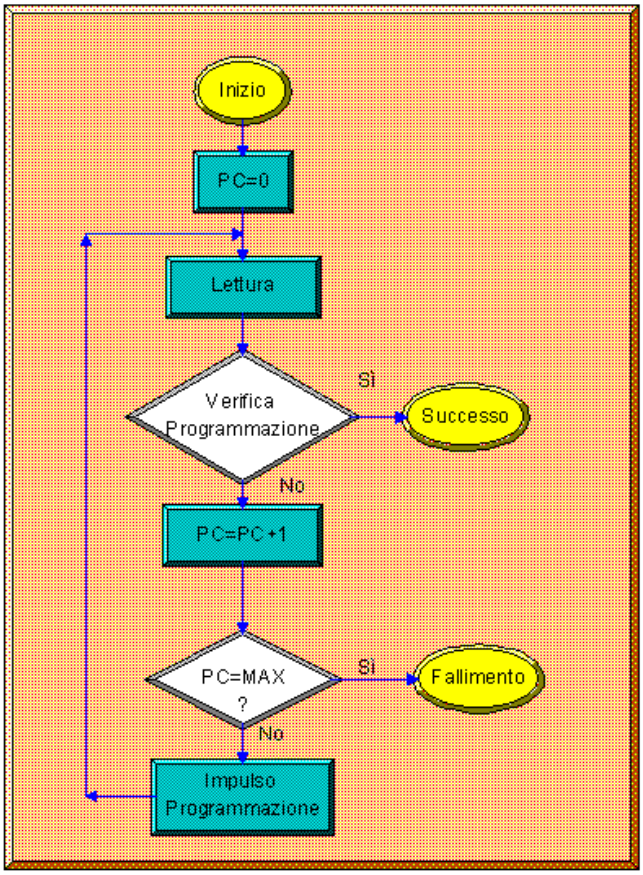

Fig.II.3.Algoritmo di programmazione

L’algoritmo è costituito dai seguenti passi fondamentali:

- azzeramento del contatore di impulsi;
- lettura della cella di memoria per controllare il livello di programmazione;
- verifica se il livello di programmazione è stato raggiunto. Se la condizione è vera la programmazione ha termine con esito positivo altrimenti va avanti;
- incrementa il contatore di impulsi;

- controlla se il contatore ha raggiunto il massimo. Se la condizione è vera la programmazione termina con esito negativo altrimenti va avanti;
- viene mandato un nuovo impulso di programmazione con tensione di control gate incrementata di delta VG e si torna a leggere la cella di memoria.

Dalla descrizione dell'algoritmo la tecnica di programmazione algoritmica risulta onerosa dal punto di vista del consumo di tempo, per questo è stata introdotta una nuova tecnica con l'intento di diminuire il tempo globale di programmazione e se possibile semplificare la complessità circuitale.

4.Programmazione analogica

verify

avvengono simultaneamente. Infatti durante l'operazione di scrittura il controllo

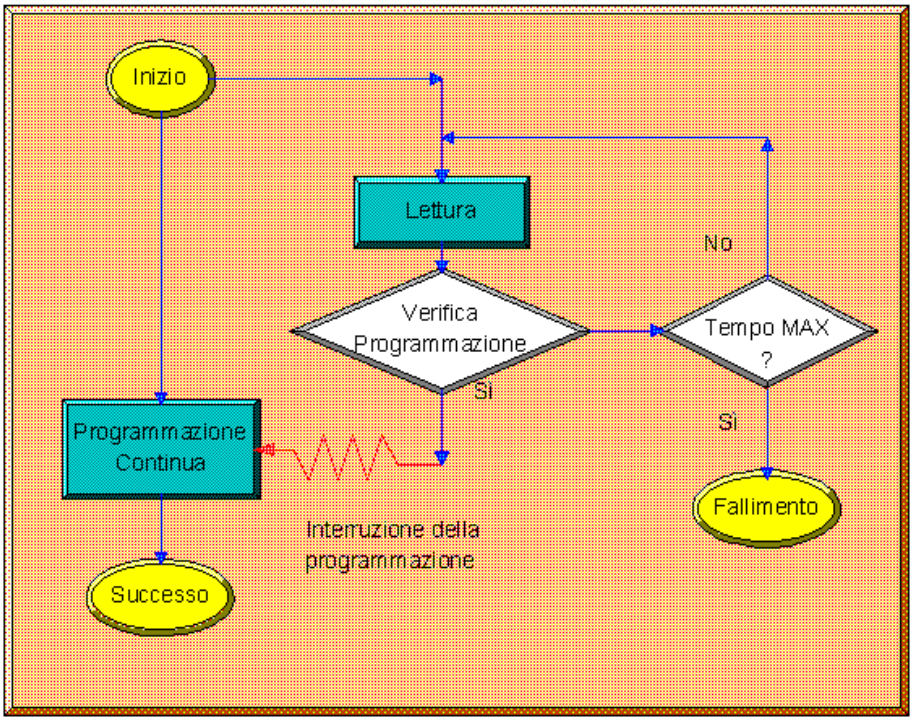

Fig.II.4.Diagramma di flusso

dello stato di programmazione si realizza mediante un apposito circuito di retroazione, il quale arresterà la programmazione quando la grandezza controllata raggiunge il valore di riferimento. Il riferimento viene precedentemente selezionato in base al dato da scrivere.

Con la programmazione analogica si ha un miglioramento delle prestazioni temporali rispetto alla programmazione algoritmica, ma c'e' il rischio di una minore precisione in quanto la grandezza controllata è una grandezza di programmazione. Nella Fig.II.4 è mostrato il diagramma di flusso del processo di scrittura di una cella di memoria col metodo analogico.

Il processo ha inizio con:

- la programmazione continua della cella di memoria;
- simultaneamente si legge il contenuto della cella e si verifica lo stato di programmazione, si decide se la programmazione deve essere fermata con successo, oppure continuare. Se la cella non raggiunge entro un tempo massimo lo stato di programmazione richiesto, la programmazione termina con una condizione di errore (fallimento).

Le grandezze che vengono scelte per controllare la programmazione della cella sono la tensione di *drain* (Vd) oppure la corrente di canale della cella di memoria (Ids). Prima di procedere nella descrizione di circuiti di programmazione analogica vediamo come queste grandezze variano durante la programmazione.

La Fig.II.5 mostra un circuito di polarizzazione della cella di memoria durante la scrittura, le tensioni applicate nella programmazione sono :

- tensione di *control gate* Vg=12V;
- tensione di alimentazione Vdd=6V.

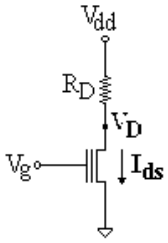

Fig.II.5.Circuito di polarizzazione

La resistenza R_D rappresenta il carico della cella e costituisce la rete di polarizzazione. Applicate le tensioni Vg e Vdd scorrerà corrente nel canale, e contemporaneamente si avrà iniezione di carica per *channel hot electrons* nella *floating-gate*. Questo comporta una diminuzione della corrente di cella Ids, e di

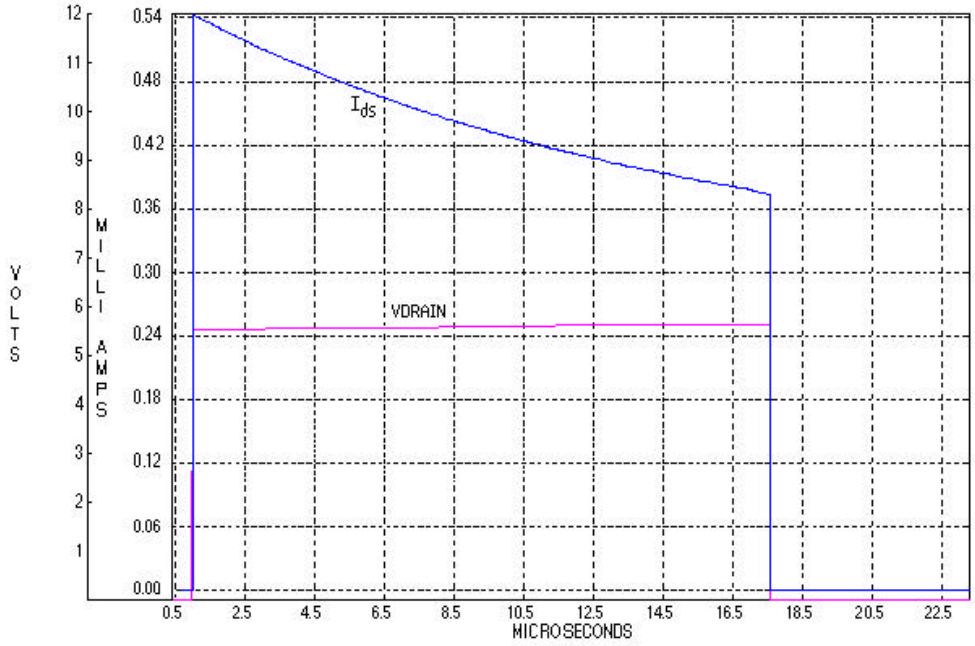

Fig.II.6.Andamenti della Ids e V_D durante la programmazione

conseguenza visto che durante la programmazione le tensioni applicate Vg e Vdd restano costanti si ha un innalzamento della tensione di drain V_D (Fig.II.6).

Le grandezze che variano durante la programmazione sono la corrente di cella Ids e la tensione di drain V_D per cui i loro valori possono essere controllati.

Nei seguenti paragrafi vengono descritti due circuiti di programmazione analogica, nel quale, in uno la grandezza controllata è la tensione di drain (V_D), nell'altro la corrente di cella (Ids). L'analisi e le simulazioni riportate sono state condotte dall'Ing. Graziano Mirichigni nella sua tesi di laurea svolta presso la Texas Istruments di Avezzano nell'anno 1998.

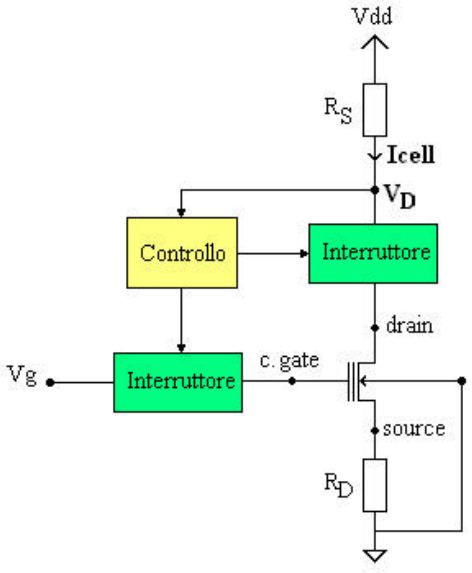

Fig.II.7.Schema di funzionamento della programmazione analogica

Nella Fig.II.7 è riportato lo schema generale di funzionamento per la programmazione analogica con i seguenti blocchi:

- blocco di controllo, il quale effettua il confronto tra la grandezza controllata e il valore di riferimento;

- blocchi di interruttore necessari per interrompere la programmazione, quando la grandezza controllata raggiunge il riferimento.

5.Circuito con amplificatore differenziale

Durante la programmazione la tensione di drain VD aumenta, per cui è possibile utilizzare un amplificatore differenziale il quale, non solo controlla quando la VD raggiunge un valore di riferimento Vref, ma genera anche il segnale di interruzione quando questa condizione si verifica. Il segnale di interruzione, generato dall'amplificatore differenziale, deve essere in grado di interrompere nel più breve tempo possibile la programmazione, portando i terminali di gate e di drain rapidamente a massa per evitare che l'operazione di programmazione continui oltre il limite voluto.

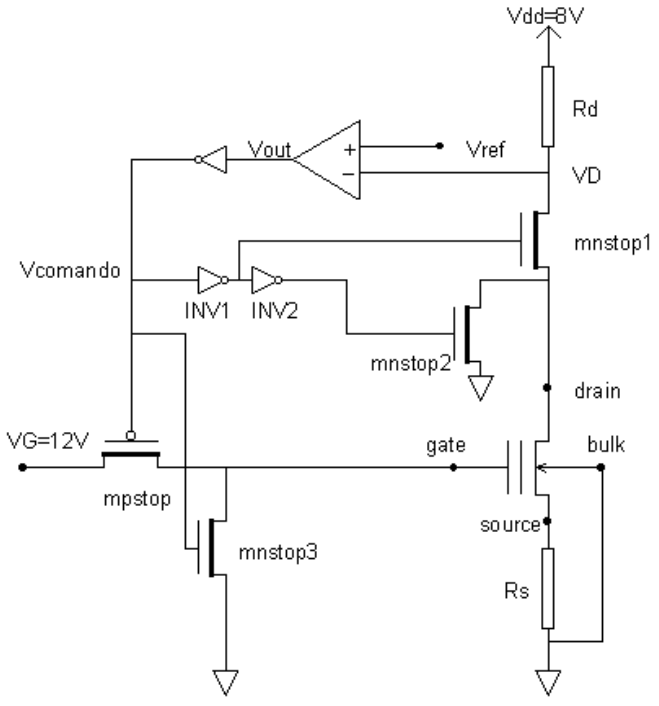

Fig.II.8.Circuito con amplificatore differenziale per la programmazione analogica.

La Fig.II.8 mostra il circuito che effettua la programmazione analogica con amplificatore differenziale.

L'amplificatore differenziale usato è schematizzato in Fig.II.9.

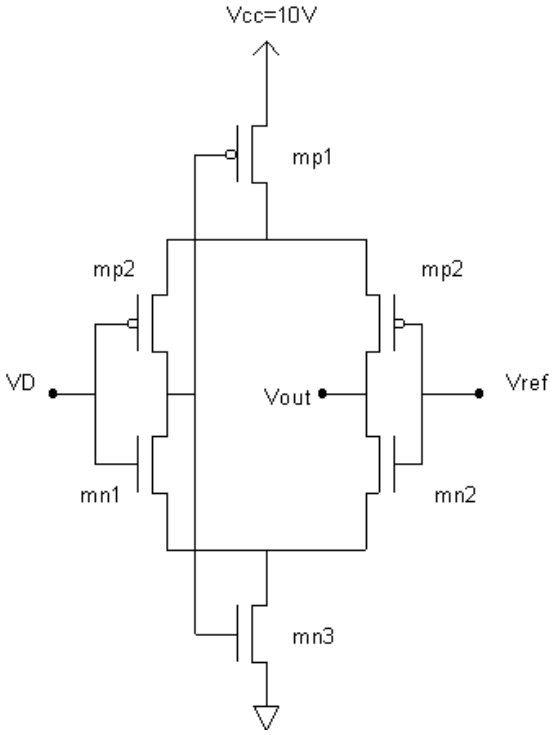

Fig.II.9.Amplificatore differenziale

Le caratteristiche dell'amplificatore sono alto guadagno, dissipazione di potenza statica nulla e transcaratteristica di tipo non invertente, cioè l'uscita è alta se l'ingresso è maggiore di Vref e bassa nel caso contrario.

Polarizzando la cella per la programmazione, con tensioni VG=12V e Vdd=6V, si ha iniezione di carica nella *floating-gate* la quale produce un aumento della tensione di soglia, con conseguente diminuzione della conduttanza di canale della cella di memoria. Questo comporta una diminuzione della corrente Ids e parallelamente un aumento di VD, al limite si ha Icell=0A e VD=Vdd.

Nell'istante in cui la VD raggiunge il valore di Vref l'amplificatore differenziale commuta basso, l'inverter all'uscita del differenziale trasforma il segnale Vout in un segnale digitale di comando Vcomando il cui livello alto vale 12V. La seguente figura mostra i segnali Vref, VD, e Vcomando.

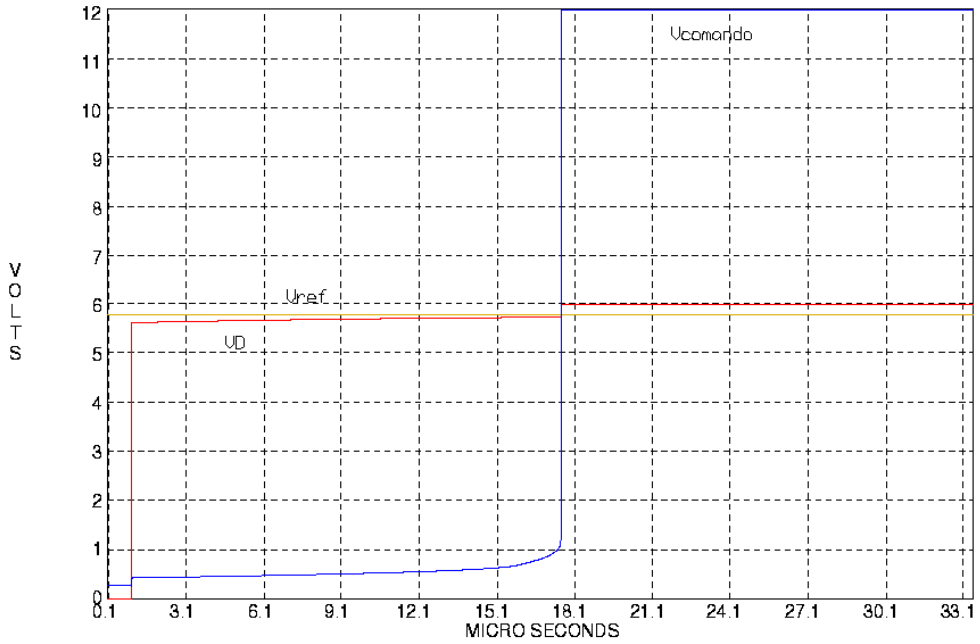

Fig.II.10.Comportamento di Vref, VD e Vcomando durante la programmazione.

Il *drain* e la *gate* vengono portati rapidamente a massa da dispositivi che lavorano da interruttore e pilotati dal segnale Vcomando=12V. In particolare Vcomando porta la *gate* da VG=12V a massa, spegnendo **mpstop** e accendendo **mnstop3**. Il segnale Vcomando viene trasformato dalla coppia di inverters INV1 e INV2 in segnale digitale con livello logico alto a 8V. L'uscita dell'inverter INV1 va a spegnere **mnstop1** in modo da fissare Vcomando=12V, mentre l'uscita dell'inverter INV2 va ad accendere **mnstop2**. In questo modo VD commuta alto verso Vdd ed il drain

viene portato a massa. Portando la gate a massa la programmazione si interrompe e la corrente di cella Ids va rapidamente a zero (Fig.II.11).

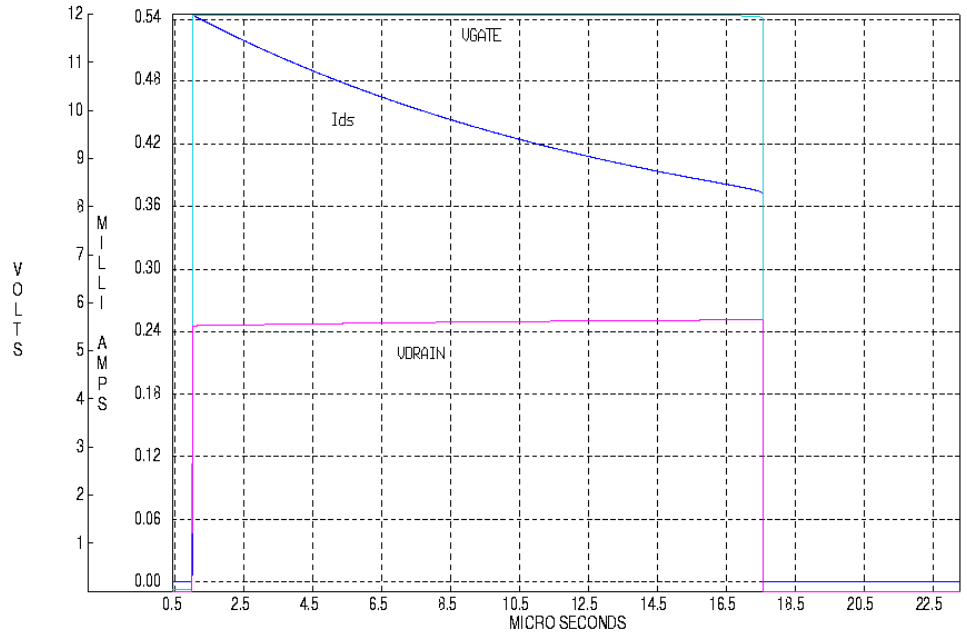

Fig.II.11.Andamento della Ids, delle tensioni di drain e di gate

La Fig.II.12 mostra invece l'andamento temporale della corrente di canale Ids della cella di memoria e delle tensioni VD e Vref.

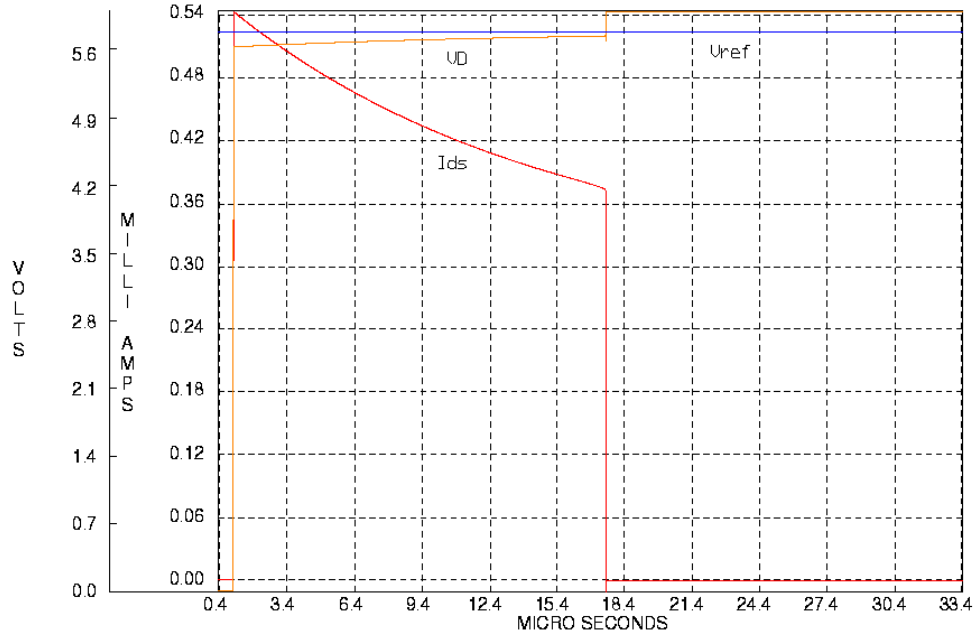

Fig.II.12.Andamento della corrente Ids e delle tensioni VD e Vref

E’ importante sottolineare la necessita’ del transistor **mnstop1**, perché quando il **mnstop2** porta a massa il drain, la VD scenderebbe sotto il valore Vref innescando di nuovo la programmazione.

Per consentire l’immagazzinamento di quantità di carica diverse nella *floating-gate*, e quindi di spostare l’istante di arresto della programmazione, si può spostare il valore del riferimento. L'analisi del differenziale ha messo in risalto che, al variare del riferimento Vref, il suo *offset* varia, questo ridimensiona almeno in parte l’utilizzo di questo circuito per la programmazione.

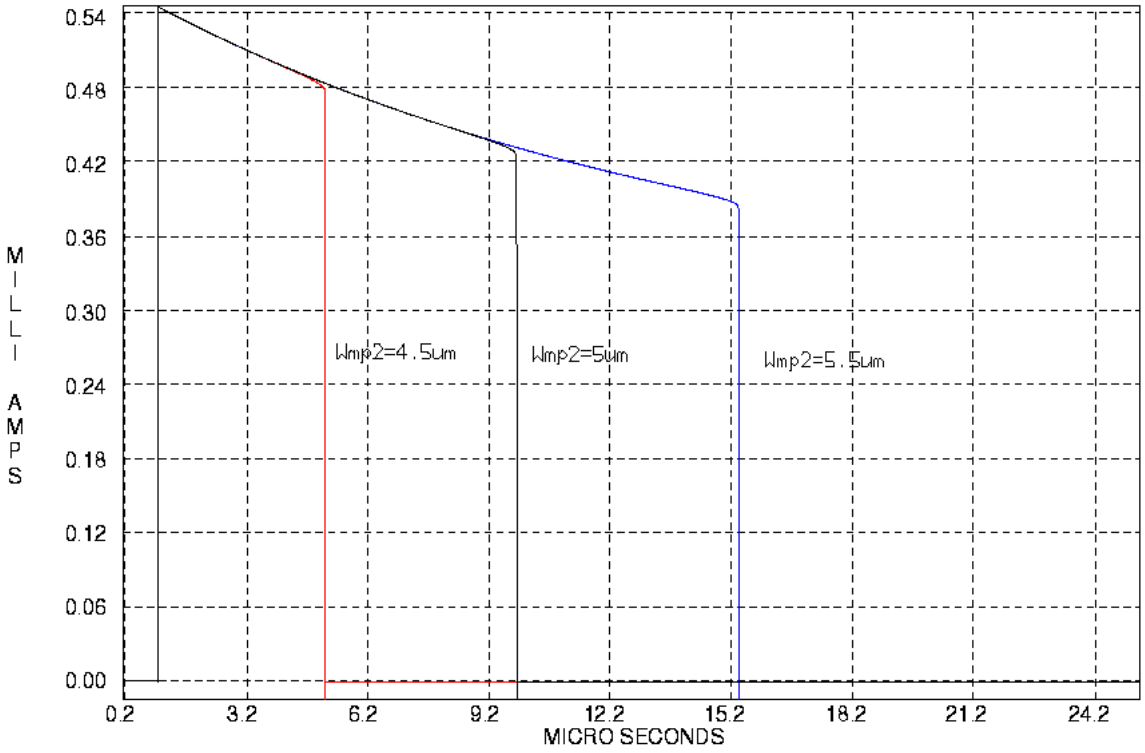

Fig.II.13.Interruzione della programmazione per differenti valori di Rd.

Un’alternativa, che consente di utilizzare il differenziale come blocco di controllo, è di mantenere costante il riferimento Vref e di variare la resistenza sul drain Rd, ciò comporta una traslazione ed una variazione della forma di VD. La Fig.II.13 mostra

l’andamento della corrente di cella al variare di Rd., è possibile notare i diversi punti di interruzione della programmazione per differenti valori di Rd.

L’aumento di Rd ritarda la salita di VD e quindi l’istante in cui raggiunge Vref, questo permette di programmare differentemente la cella.

6.Circuito con specchi di corrente

Il circuito che verrà descritto nel seguito è costituito da due specchi di corrente, in grado di effettuare il confronto di due correnti (Fig.II.14).

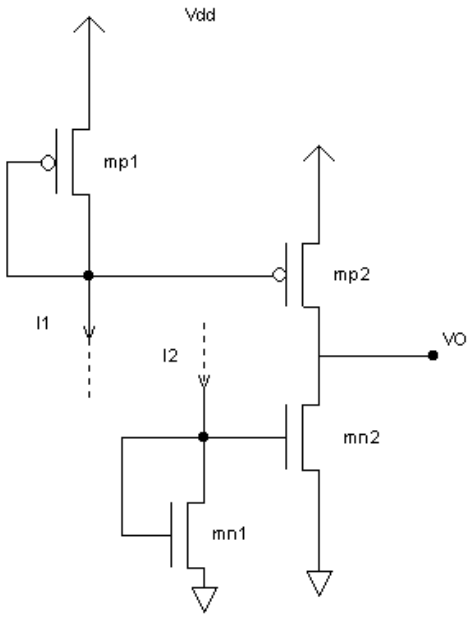

Fig.II.14.Specchi per il confronto di due correnti

Le correnti I_1 e I_2 risulteranno uguali se il rapporto di specchio è unitario, e l’uscita VO si stabilizzerebbe ad un valore di riferimento VO* pari a Vdd/2.

Mantenendo unitario il rapporto di specchio e supponendo le correnti $I_1=kI$ e $I_2=I$, si avrebbe nello specchio PMOS una corrente maggiore dello specchio NMOS, il quale porterebbe l'uscita VO ad un valore maggiore di VO* e il transistor **mp2** uscirebbe dalla zona di saturazione.

È necessario intervenire sul rapporto di specchio per riportare l'uscita VO a VO*, valore che aveva quando $I_1=I_2$. Affinché questa condizione sia di nuovo verificata è necessario che il rapporto di specchio, definito dalla relazione (Wmp2/Lmp2)/(Wmp1/Lmp1) sia uguale a 1/k.

Questo modo di procedere permette di confrontare correnti molto diverse tra loro, sfruttando l'informazione relativa alla variazione della tensione di uscita VO rispetto al riferimento VO*.

Durante la programmazione la corrente di canale Ids parte da valori di corrente molto elevati (circa 400uA) per arrivare ad un valore minimo quando l'operazione è a regime.

Per controllare la programmazione della cella di memoria si procede nel modo seguente: si divide la corrente Ids per un fattore k opportuno e si confronta con un valore di corrente di riferimento Iref. La tensione VO che si ottiene dal confronto viene utilizzata come segnale di comando per interrompere la programmazione. Tale segnale deve essere in grado di portare a massa, il più rapidamente possibile, i terminali di gate e di drain della cella di memoria.

L'uscita VO assume il valore di riferimento VO* quando la corrente di cella scalata raggiunge il riferimento Iref. Inizialmente il suo valore si troverà ad un livello alto e maggiore di VO*, quando si ha l'uguaglianza tra le correnti il segnale commuta basso.

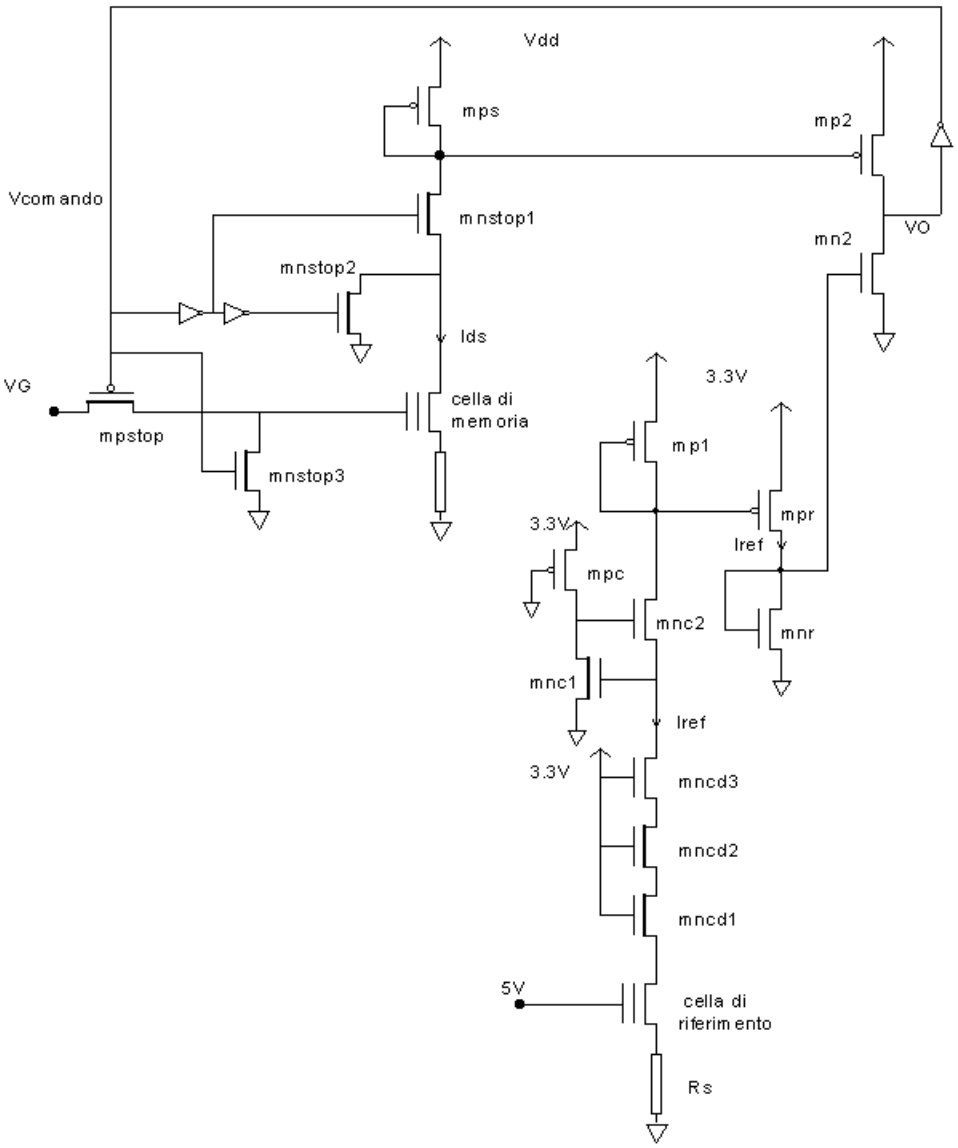

Fig.II.15.Circuito con specchi di corrente

Il circuito per la programmazione analogica con specchi di corrente è mostrato in Fig.II.15, lo schema comprende anche il circuito che genera il riferimento di corrente.

Le tensioni applicate sono VG=12V e Vdd=8V, con queste polarizzazioni la cella inizia a programmarsi fin quando la corrente di canale Ids, scalata per un determinato fattore k, risulta maggiore del riferimento Iref. Infatti nell'istante in cui k*Ids raggiunge il riferimento Iref, la tensione VO in uscita al circuito di confronto, che durante la programmazione si trova al livello alto, commuta basso. Questo segnale viene trasformato da un inverter in un segnale digitale di comando con valore alto pari a 12V; tale segnale è in grado di interrompere la programmazione portando a massa i terminali di drain e di gate della cella di memoria.

Il terminale di gate è collegato a Vcomando attraverso gli interruttori **mpstop** e **mnstop3**; quando il segnale di comando va a 12V il primo di questi interruttori si chiude, mentre il secondo si apre portando a massa la gate della cella.

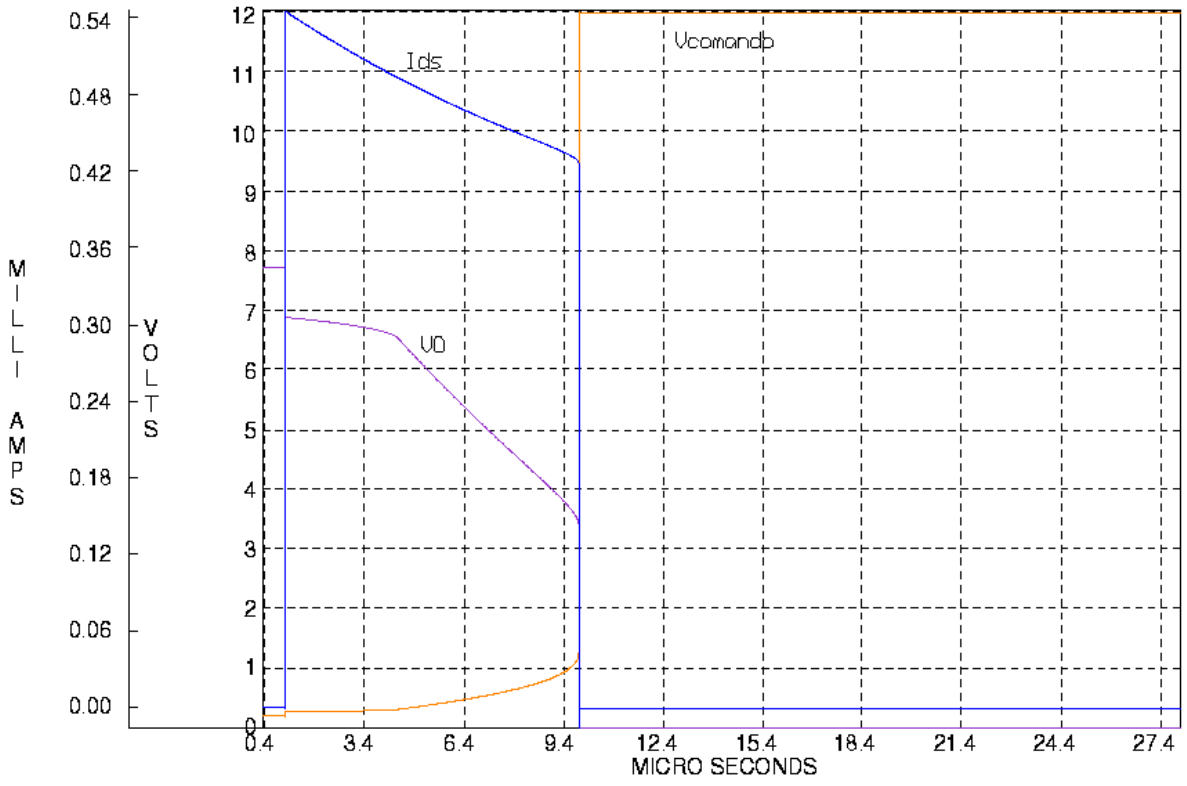

Fig.II.16.Risultati della simulazione del circuito con specchi di corrente

La coppia di inverters presente nello schema circuitale trasforma Vcomando in un segnale digitale di valore alto 8V. L'uscita del primo inverter spegne **mnstop1** fissando Vcomando a 12V, mentre il secondo inverter chiude **mnstop2** portando il drain della cella a massa.

La Fig.II.16 mostra l'andamento della corrente di canale della cella di memoria, del segnale d'uscita VO e del segnale Vcomando che interrompe la programmazione.

È possibile anticipare o ritardare l'istante di commutazione dell'uscita VO del circuito, che esegue il confronto tra la corrente di canale della cella di memoria scalata e la corrente di riferimento, variando il rapporto di specchio dei transistors PMOS.

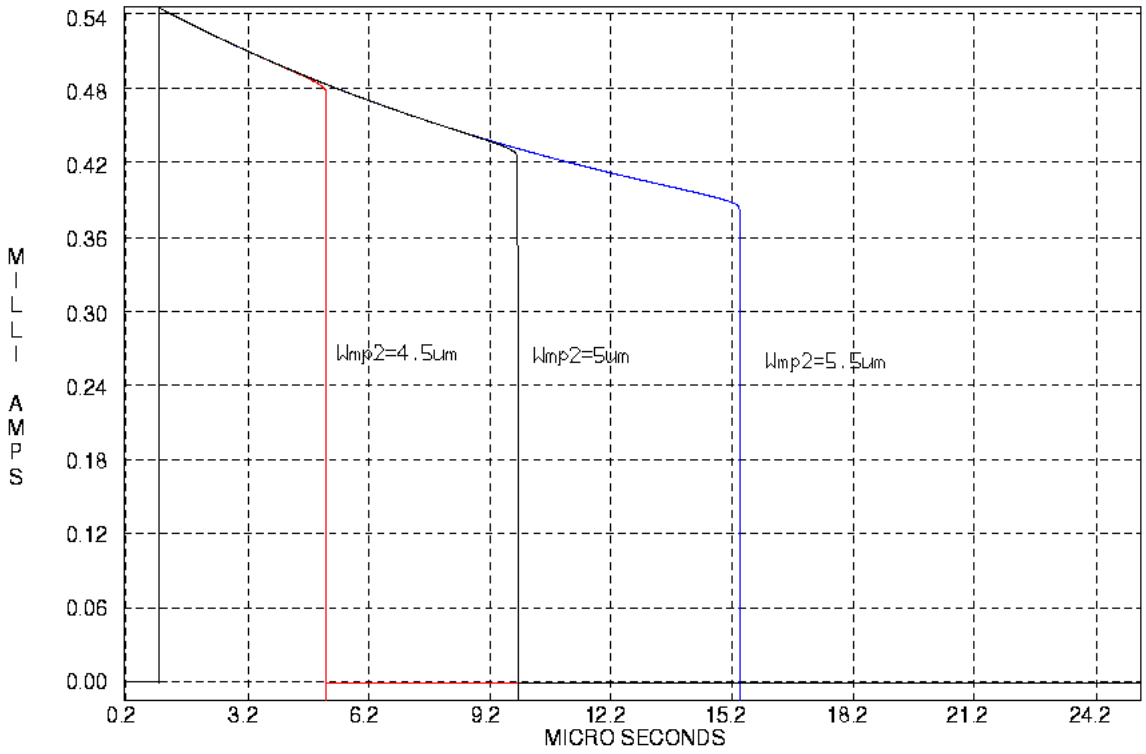

Fig.II.17.Andamentodella corrente di canale Ids per diversi valori del rapporto di specchio

La Fig.II.17 mostra i risultati delle simulazioni per diversi valori del parametro Wmp2, e mostra anche come a parità di polarizzazione della cella è possibile variare l'istante di interruzione della programmazione, consentendo di immagazzinare diverse quantità di carica nella *floating-gate*

7.Affidabilità delle memorie multi-livello

L'immagazzinamento di più di due bit per cella non introduce, in generale, nuovi problemi fisici di rilievo per l'affidabilità rispetto al caso di memorie Flash convenzionali, ma sicuramente rende più critici tutti gli aspetti legati alla ritenzione dei dati a causa della ridotta separazione tra un livello e l'altro. Mentre in una memoria a due livelli i margini di affidabilità sono tipicamente superiori al volt, nel caso di una memoria a quattro livelli essi si riducono a poche centinaia di mV. I due problemi principali sono costituiti dalla perdita di elettroni nella *floating-gate* per le celle che si trovano nei livelli a soglia più alta e quello, complementare, della introduzione di elettroni nella *floating-gate* per le celle a soglia più bassa. Per il primo meccanismo la driving force è data dal campo elettrico generato dalla carica negativa nella *floating-gate* ed è attivo anche a dispositivo spento. Il secondo meccanismo, noto come disturbo in lettura, è attivo per quelle celle che si trovano sulla stessa word-line di una cella selezionata in lettura e che hanno quindi una tensione positiva applicata sul *control gate*. Il caso peggiore da gestire è quello della lettura continuata di celle sulla stessa word-line.

I dati di affidabilità raccolti su memorie Flash convenzionali, e prove specifiche su dispositivi di test, indicano che è possibile soddisfare i requisiti di affidabilità di

memorie a quattro livelli, almeno nel caso di dispositivi non sottoposti a cicli di programmazione e cancellazione. Sta però emergendo l'evidenza di una degradazione delle qualità dell'ossido di tunnel, in seguito a ripetuti cicli di programmazione e cancellazione, che può dar luogo a problemi di disturbi in lettura e di perdita di elettroni dalla *floating-gate*. I dati sperimentali mostrano che questa degradazione aumenta notevolmente al diminuire dello spessore dell'ossido. Da questo punto di vista le memorie programmabili per CHE hanno un punto a favore, in quanto la velocità di programmazione non dipende significativamente dallo spessore dell'ossido. Al contrario la programmazione per FN richiede tensioni proporzionali allo spessore dell'ossido per avere pari velocità di programmazione.

Nel caso di memorie multi-livello, ripetuti cicli di programmazione, possono dar luogo a variazioni di soglia non compatibili con i ridotti margini tra i livelli. Volendo quindi garantire sia la possibilità di effettuare un numero elevato di cicli di *write/erase* , sia la ritenzione dei dati per tempi dell'ordine di anni, l'unica possibilità è di utilizzare nelle memorie multi-livello opportuni codici per la correzione degli errori. I vantaggi di queste memorie, in termini di riduzione di area del chip, vengono quindi in parte ridimensionati dalla necessità di introdurre circuiti addizionali per la correzione dell'errore, rendendo questa soluzione effettivamente vantaggiosa solo