Progetto Verilog A.A. 2016-17 – MMU.

Nome: Andrea **Cognome**: Tosti **Matricola**: 518111

Obiettivo:

Va realizzata un'unità che implementi l'algoritmo di traduzione degli indirizzi logici in indirizzi fisici nella MMU. L'unità riceve un indirizzo logico da 32 bit con IPL da 20 bit e OFFSET da 12 bit e lo traduce nel corrispondente indirizzo fisico utilizzando una cache completamente associativa da 8 posizioni. Va gestito il meccanismo di rimpiazzamento di un'entry della cache in caso di fault, che si avverrà di un'interfaccia standard verso il sottosistema di memoria in grado di richiedere operazioni di lettura e di un registro dedicato TABRIL contenente la base della tabella di rilocazione per il processo in esecuzione. La politica di rimpiazzamento nella cache può essere implementata come politica round robin.

...

Per quanto riguarda Verilog, tra le difficoltà incontrate, quella maggiore è stata risolvere il non-determinismo incontrato in fase di scrittura/lettura del contenuto dei registri a fine ciclo di clock. Un punto a favore è sicuramente il Debugging, che mi ha permesso di scoprire tale non-determinismo, nel mio caso ho risolto con un `assegnamento Non-Blocking`.

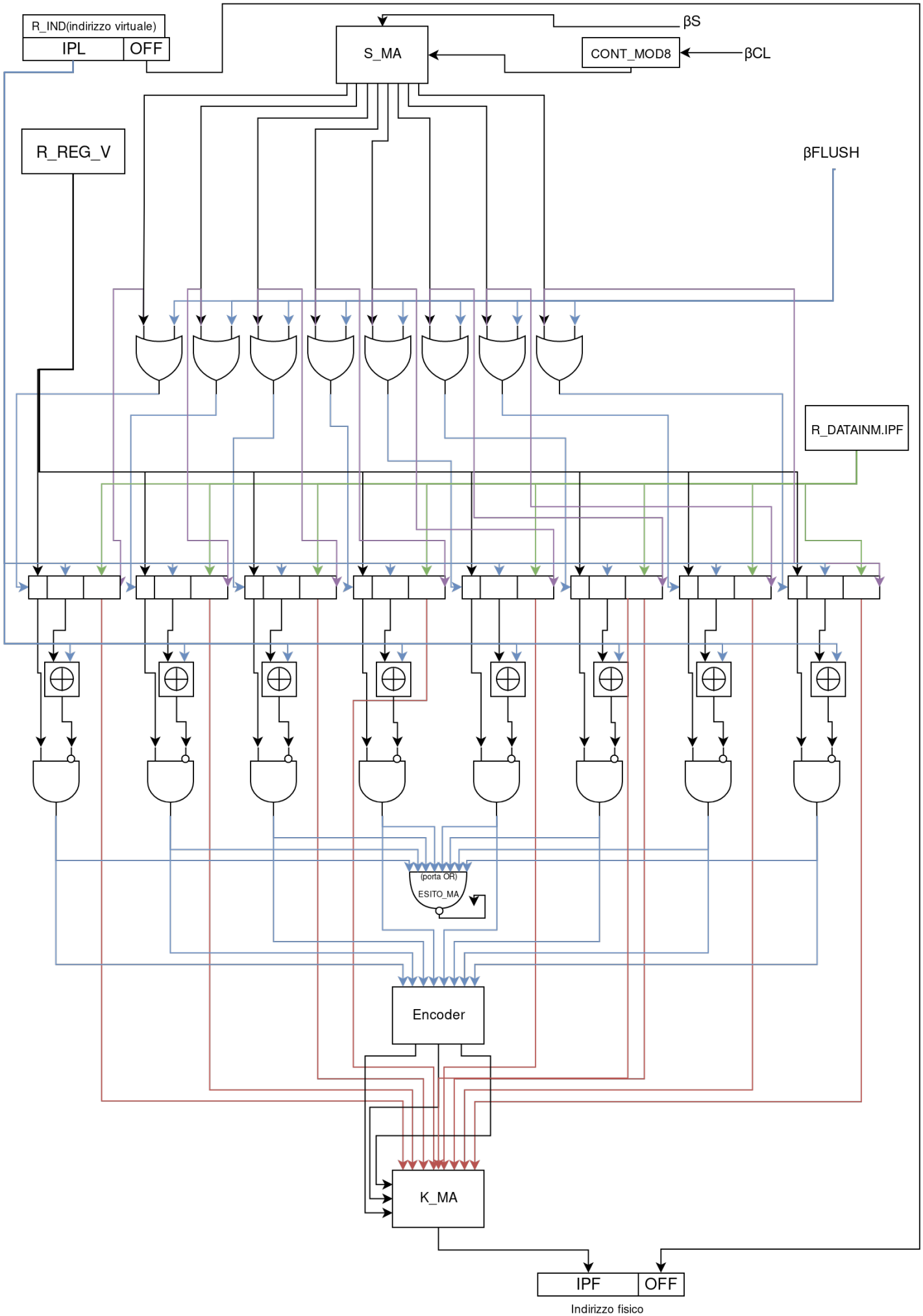
```
always @(posedge clock) //registro

    begin
        if(beta == 1)
            stato <= in; //Non-blocking
    end
```

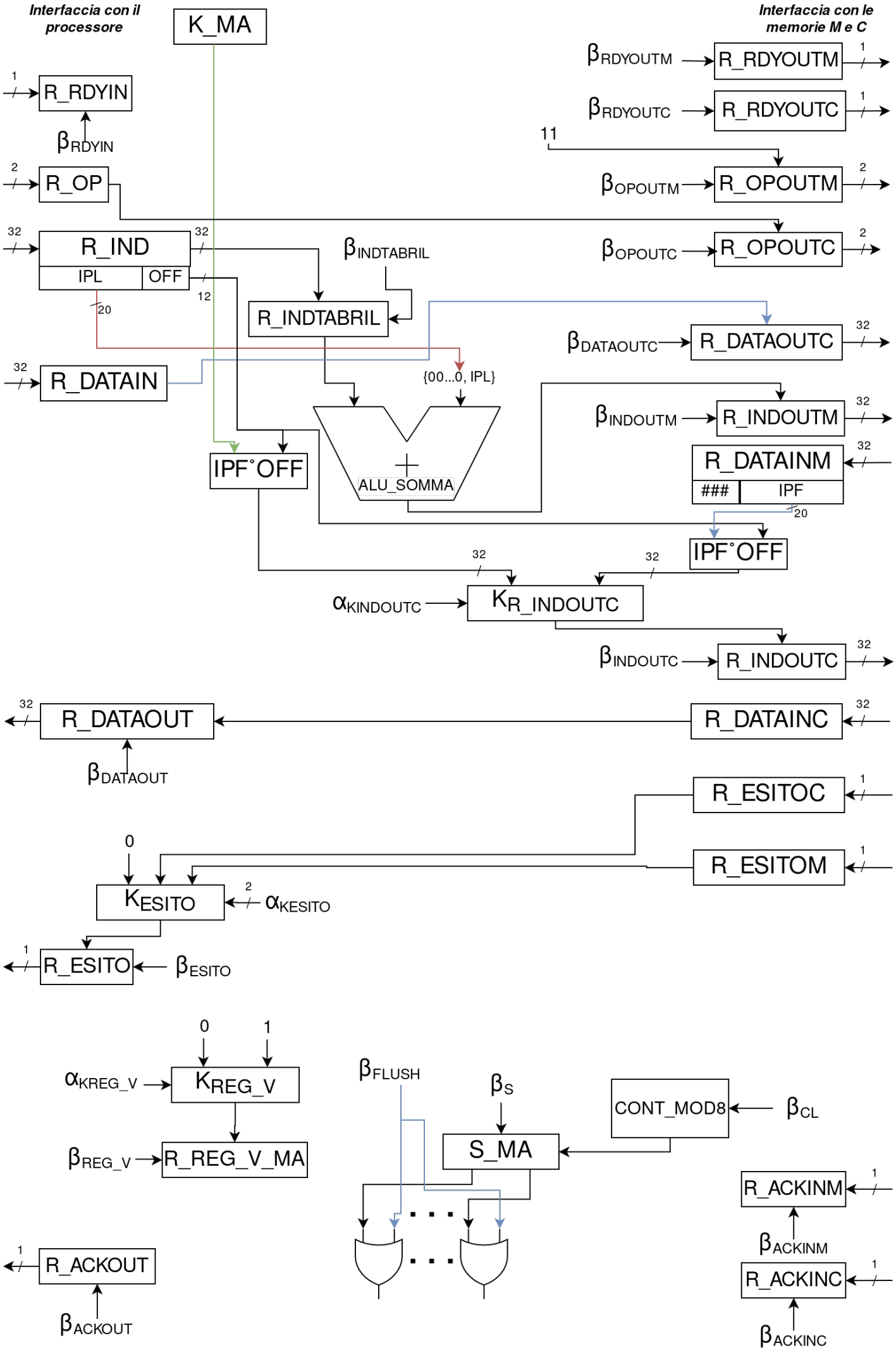
Testcase1 - File necessari per produrre testcase1.vcd :

testcase1_MMU.vl MMU.vl PC.vl PO.vl OmegaPC.vl SigmaPC.vl registro.vl rdyack_in.vl rdyack_out.vl comparatore.vl commutatore_2vie_nbit.vl commutatore_3vie_nbit.vl alu_somma.vl contatore_mod8.vl selezionatore_8vie_nbit.vl or_tp.vl confrontatore_nbit.vl confrontatore_1bit.vl and_tp.vl or_tp_8vie_neg.vl encoder.vl commutatore_8vie_nbit.vl

Ho preso spunto soprattutto dai moduli scritti dal Prof. Danelutto.

Nelle pagine a seguire ci sono un'inquadratura generale dello schema di riferimento, la Parte Operativa, Tabella di Verità (di cui viene fornito anche un ulteriore file tab_verita.pdf che ha orientamento orizzontale), MicroProgramma, assunzioni fatte sulle componenti, tra cui i tempi di stabilizzazione. Infine c'è un modulo di test che simula l'interazione tra Processore e MMU, tra MMU e Cache e tra MMU e Memoria, a seguire la spiegazione dettagliata di cosa succede durante la simulazione. Nel codice di testcase1_MMU.vl è possibile aggiungere facilmente nuove operazioni stando attenti ad alternare le linee RDYIN e ACKIN (0 -> 1 e 1 -> 0). Prima di risolvere il non-determinismo spiegato sopra, bisognava impostare i tempi ad Hoc, ma dopo la modifica ciò non è più necessario.


R_IND(indirizzo virtuale)
IPL
OFF
S_MA
βS
CONT_MOD8
βCL
R_REG_V
βFLUSH
R_DATAINM.IPF
(porta OR)
ESITO_MA
Encoder
K_MA
IPF
OFF
Indirizzo fisico


Interfaccia con il processore
K_MA
Interfaccia con le memorie M e C
βRDYOUTM
R_RDYOUTM
βRDYOUTC
R_RDYOUTC
11
βOPOUTM
R_OPOUTM
βOPOUTC
R_OPOUTC
R_RDYIN
βRDYIN
R_OP
R_IND
IPL
OFF
βINDTABRIL
R_INDTABRIL
βDATAOUTC
R_DATAOUTC
R_DATAIN
{00...0, IPL}
ALU_SOMMA
βINDOUTM
R_INDOUTM
R_DATAINM
###
IPF
IPF˚OFF
αKINDOUTC
KR_INDOUTC
βINDOUTC
R_INDOUTC
R_DATAOUT
βDATAOUT
R_DATAINC
R_ESITOC
R_ESITOM
KESITO
αKESITO
R_ESITO
βESITO
αKREG_V
KREG_V
βREG_V
R_REG_V_MA
βFLUSH
βS
S_MA
CONT_MOD8
βCL
R_ACKOUT
βACKOUT
R_ACKINM
βACKINM
R_ACKINC
βACKINC

Inquadratura generale

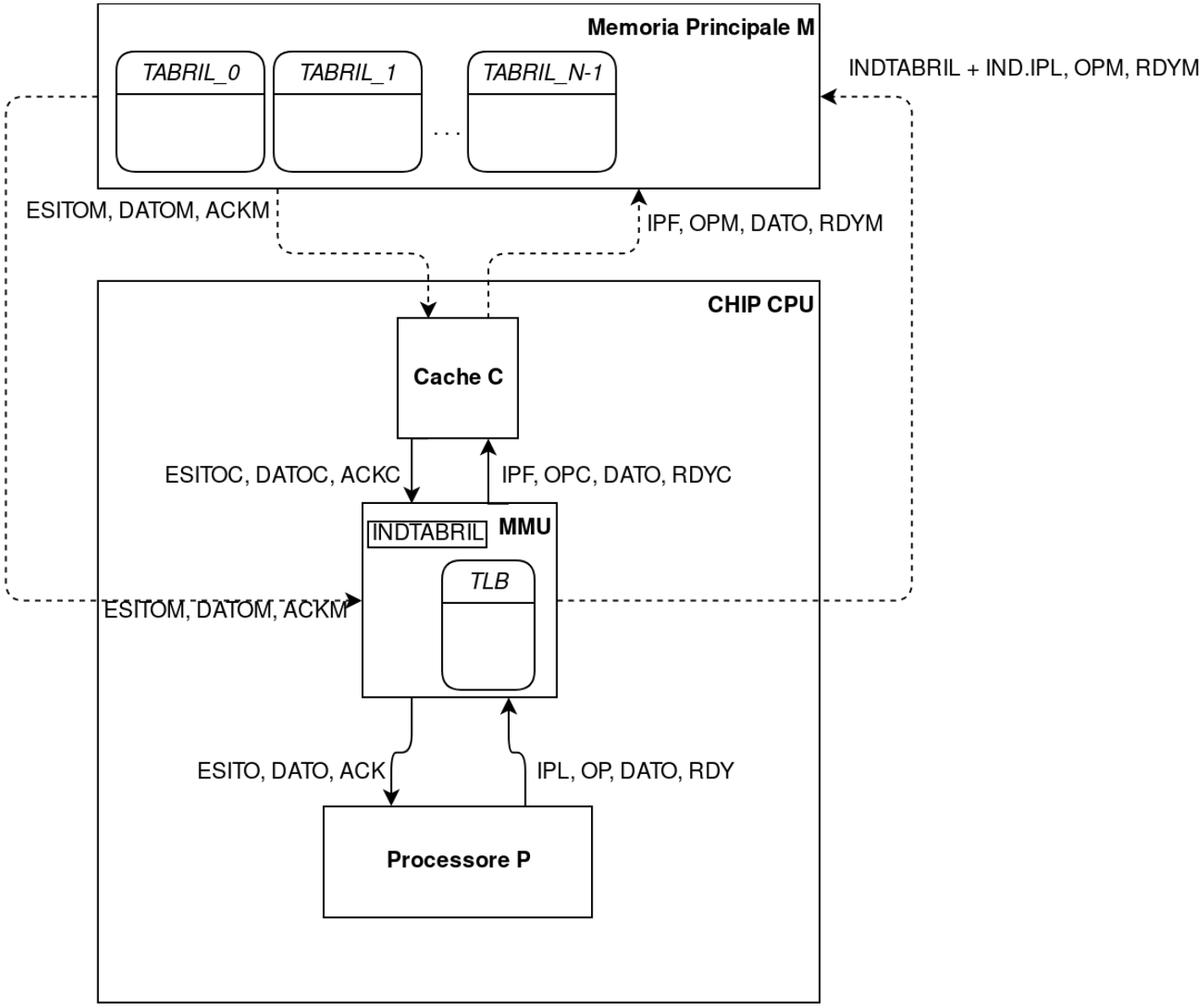


Tabella di verita' (vedere file tab_verita.pdf per la versione A4 landscape)

															$\hookrightarrow_{PC}$																							σ_{PC}	
S_1	S_0	RDYIN	OP_1	OP_0	MA.ESITO	ACKINC	$OPOUTC_1$	$OPOUTC_0$	ESITOC	ACKINM	$OPOUTM_1$	$OPOUTM_0$	ESITOM	$\beta_{INDOUTC}$	β_{OPOUTC}	$\beta_{RDYOUTC}$	β_{ACKINC}	$\beta_{DATAOUTC}$	$\beta_{INDTABRIL}$	β_{ESITO}	β_{RDYIN}	β_{ACKOUT}	β_{FLUSH}	β_{OPOUTM}	$\beta_{INDOUTM}$	$\beta_{RDYOUTM}$	β_{ACKINM}	β_{REG_V}	$\beta_{DATAOUT}$	β_{CL}	β_S	$\alpha_{KINDOUTC}$	α_{KESITO_1}	α_{KESITO_0}	α_{KREG_V}	S_1'	S_0'
0	0	0	-	-	-	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	-	0	0
0	0	1	0	0	0	-	-	-	-	-	-	-	-	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	0	1
0	0	1	0	1	0	-	-	-	-	-	-	-	-	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	0	1
0	0	1	1	0	-	-	-	-	-	-	-	-	-	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	-	0	0	-	0	0
0	0	1	0	0	1	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	-	-	-	1	1	0
0	0	1	0	1	1	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	-	-	-	1	1	0
0	1	-	-	-	0	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	0	1
0	1	-	-	-	1	0	0	0	-	-	-	-	-	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	-	0	0	-	0	0
0	1	-	-	-	1	0	1	0	-	-	-	-	-	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	-	0	0	-	0	0
0	1	-	-	-	1	-	-	1	-	-	-	-	-	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	-	0	1	-	0	0
1	0	-	-	-	-	-	-	-	0	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	-	1	0
1	0	-	0	0	-	-	-	-	-	1	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	-	-	0	0	1
1	0	-	0	1	-	-	-	-	-	1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	1	1	-	-	0	0	1
1	0	-	-	-	-	-	-	-	-	1	-	-	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	-	1	0	0	0	0

Microprogramma MMU

0. (RDYIN, OP, MA(IND.IPL).ESITO = 0 - -) nop, 0.

// Ho ricevuto un indirizzo logico dal processore, TLB-HIT, mando l'indirizzo fisico e l'operazione richiesta dal processore alla Cache C (lettura parola);
(= 1 00 0) (MA(IND.IPL).IPF°IND.OFFSET) → INDOUTC, OP → OPOUTC, set RDYOUTC, reset ACKINC, 1.

//Ho ricevuto un indirizzo logico dal processore, TLB-HIT, mando l'indirizzo fisico e l'operazione richiesta dal processore alla Cache C (scrittura parola);
(= 1 01 0) (MA(IND.IPL).IPF°IND.OFFSET) → INDOUTC, OP → OPOUTC, DATAIN → DATAOUTC, set RDYOUTC, reset ACKINC, 1.

//Ho ricevuto un indirizzo fisico dal processore, eseguo il flush della TLB mettendo i bit di validita' di tutte le linee a zero, memorizzo l'indirizzo fisico di TABRIL e mando l'esito al processore;
(= 1 10 -) IND → INDTABRIL, 0 → ESITO, reset RDYIN, set ACKOUT, 0.

//Ho ricevuto un indirizzo logico dal processore, TLB-MISS, chiedo alla memoria M la linea di cache;
(= 1 00 1 , 1 01 1) 11 → OPOUTM, (INDTABRIL + IND.IPL) → INDOUTM, set RDYOUTM, reset ACKINM, 1 → REG_V, 2.

//Nel caso in cui la Cache C faccia Hit, mandera' esito ed eventuale dato alla MMU;
//Nel caso in cui la Cache C faccia Miss, la pagina sara' in memoria M, la Cache C comunichera' con la //memoria M, la memoria M restituira' il dato e l'esito alla Cache, la Cache memorizzera' il dato e //passera' l'esito e l'eventuale dato letto alla MMU;

1. (ACKINC, OPOUTC, ESITOC = 0 - -) nop, 1.

//Ho ricevuto il dato dalla cache, lo mando al processore
(= 1 00 0) DATAINC → DATAOUT, 0 → ESITO, set ACKOUT, reset RDYIN, 0.

//Ho ricevuto la conferma di scrittura dalla cache, mando l'esito al processore
(= 1 01 0) 0 → ESITO, set ACKOUT, reset RDYIN, 0.

//Errore, mando l'esito al processore
(= 1 --- 1) ESITOC → ESITO, set ACKOUT, reset RDYIN, 0.

2. (ACKINM, OP, OPOUTM, ESITOM = 0 – – –) nop, 2.

//La memoria M manda la linea di cache trovata nella TABRIL del processo con bit di presenza = 1, //provvedo a sovrascrivere la linea nella TLB puntata dal registro CL (current line) e incremento tale //registro (politica Round Robin), dopodiche' mi appresto a chiedere di nuovo alla Cache il dato

(= 1 00 11 0) (1˚IND.IPL˚DATAINM.IPF) -> MA[CL], CL+1 -> CL, 0 → REG_V, DATAINM.IPF°IND.OFFSET → INDOUTC, OP → OPOUTC, set RDYOUTC, reset ACKINC, 1.
(= 1 01 11 0) (1˚IND.IPL˚DATAINM.IPF) -> MA[CL], CL+1 -> CL, 0 → REG_V, DATAINM.IPF°IND.OFFSET → INDOUTC, OP → OPOUTC, set RDYOUTC, reset ACKINC, DATAIN → DATAOUTC, 1.

//Ricevo una risposta dalla memoria principale, si tratta di un Page Fault, invio tale esito al processore;
(= 1 – – 1) ESITOM -> ESITO, set ACKOUT, reset RDYIN, 0 → REG_V, 0.

Assunzioni:

- Il ritardo di una porta logica con al massimo 8 ingressi è di 1tp
- Il ritardo di una ALU è di 5tp

Tempi di stabilizzazione

Parte Controllo

3 microistruzioni, 14 frasi, 12 variabili di condizionamento di cui al massimo 6 testate contemporaneamente. Quindi gli ingressi delle porte AND saranno al massimo 8 (2 bit di stato + 6 variabili di condizionamento) e servirà un livello di porte AND. Per 14 frasi serve 1 livello di porte OR. Pertanto, $\mathbf{T\omega_{PC} = T\sigma_{PC} = 2\ tp}$

Parte Operativa

Per la ω_{PO} ci sono le variabili di condizionamento rdyin, op1, op0, ackinc, opoutc1, opoutc0, esitoc, ackinm, opoutm1, opoutm0, esitom che costano 0tp, mentre la variabile esito_ma passa per 1 livello OR, 1 livello AND, per un confrontatore che ha in ingresso 20 bit , dove si hanno 20 confrontatori da un bit (che lavorano in parallelo, 1 livello OR e 1 livello AND), le cui uscite vanno in 3 porte OR (che lavorano in parallelo, 1 livello OR), le cui uscite vanno in 1 porta OR (1 livello OR). Pertanto, $\mathbf{T\omega_{PO} = 6\ tp}$.

Per la σ_{PO} si hanno:

$\mathbf{T\sigma_{PO/RDYOUTM} = T\sigma_{PO/RDYOUTC} = T\sigma_{PO/ACKINM} = T\sigma_{PO/ACKINC} = T\sigma_{PO/DATAOUTC}} = \mathbf{T\sigma_{PO/INDTABRIL}} = \mathbf{T\sigma_{PO/DATAOUT}} = \mathbf{T\sigma_{PO/OPOUTM} = T\sigma_{PO/OPOUTC} = T\sigma_{PO/ACKOUT}} = \mathbf{T\sigma_{PO/DATAINM}} = 0$ tp

$\mathbf{T\sigma_{PO/ESITO}}$ = Tk_esito = 2 tp

$\mathbf{T\sigma_{PO/REG\_V}}$ = Tk_reg_v = 2 tp

$\mathbf{T\sigma_{PO/CONTATORE\_MOD8}}$ = 2 tp (con Tab. verità e somma di prodotti ho 1 livello OR e 1 livello AND)

$\mathbf{T\sigma_{PO/INDOUTM}}$ = Talu = 5 tp

$\mathbf{T\sigma_{PO/INDOUTC}}$ = Tk_indoutc (2 tp) + Tk_ma (2 tp) + Tencoder (2tp) + Livello AND (1tp) + Tconfrontatore (6tp) = 13 tp

$\mathbf{T\sigma_{PO/LINEA\_TLB\_IPL\_IPF}}$ = Ts_ma (1 tp) + Livello OR (1 tp) = 2 tp

$\mathbf{T\sigma_{PO/LINEA\_TLB\_Validità}}$ = Ts_ma (1 tp) = 1 tp

Pertanto, si ha $\mathbf{T\sigma_{PO}} = \mathbf{T\sigma_{PO/INDOUTC}}$ = 13 tp

$\boldsymbol{\tau_{MMU}} = \mathbf{T\omega_{PO}} + \max(\mathbf{T\omega_{PC} + T\sigma_{PO}, T\sigma_{PC}}) + \boldsymbol{\delta}$ = 6 tp + max(2 tp + 13 tp, 2 tp) + 1 tp = 6 tp + 15 tp + 1 tp = 22 tp

Le operazioni (OP) che la MMU è in grado di riconoscere sono

00 : READ (lettura di un’istruzione o di un dato, da inoltrare alla Cache C)

01 : WRITE (scrittura di un dato ad un certo indirizzo, da inoltrare alla Cache C)

10 : START_PROCESS (commutazione di contesto, quindi flush della TLB)

Al fine di simulare le interazioni tra processore e MMU, tra MMU e cache C, tra MMU e memoria M, vengono usate, nei testcase Verilog, le costanti TAU, TAU_C e TAU_M. Vengono assunti dei ritardi poco realistici al fine di rendere piu’ comprensibile la simulazione.

Inoltre, per utilizzare il protocollo di comunicazione a domanda e risposta, vengono alternate le linee d’ingresso ACKIN e RDYIN (0 -> 1 e da 1 -> 0).

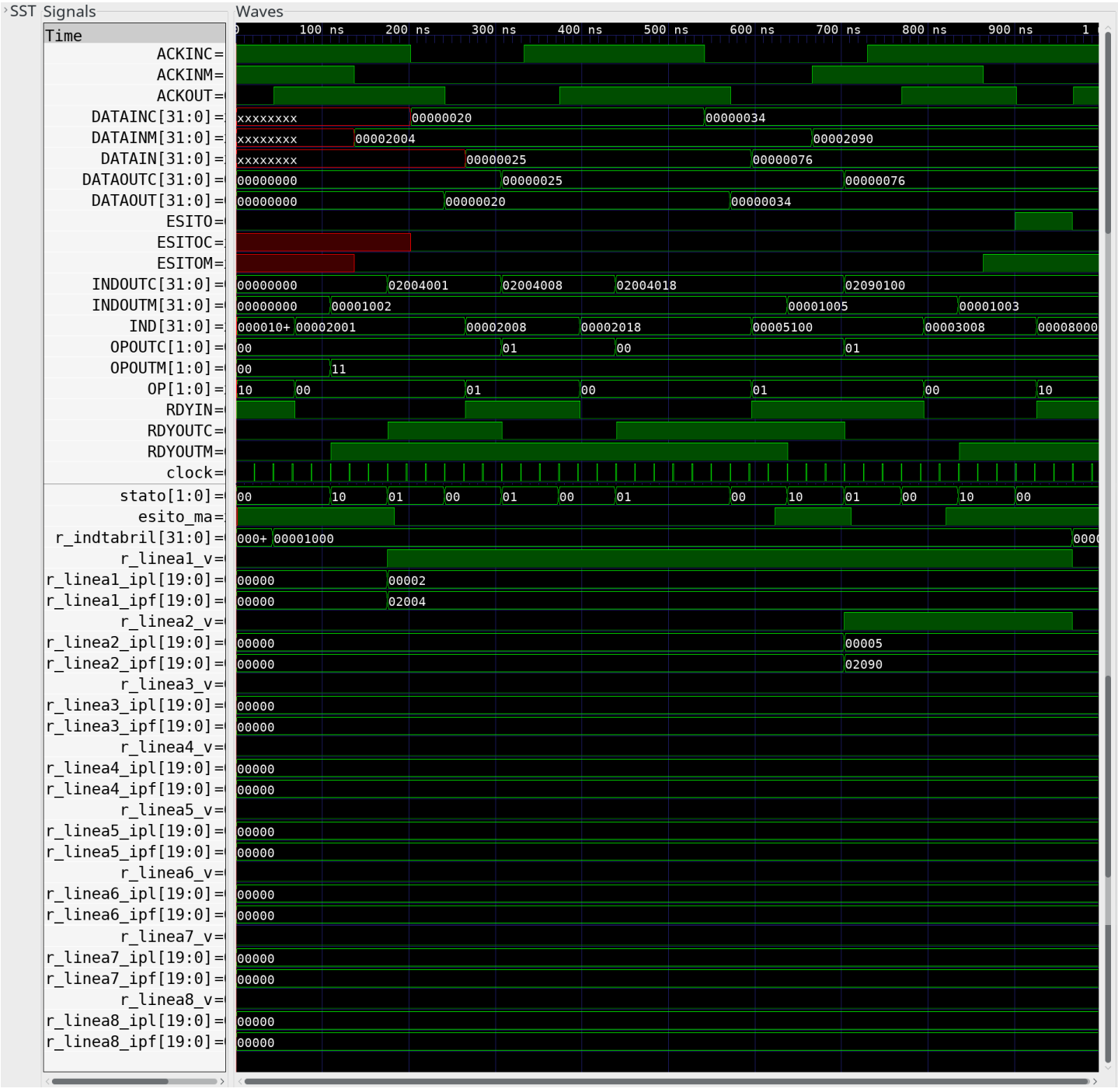
La politica di rimpiazzamento della TLB è Round Robin, quindi la linea viene scelta ad ogni rimpiazzamento a seconda del valore del contatore modulo 8. Le linee di cache sono implementate come registri da 40 bit (20 bit per TAG e 20 bit per IPF). Vengono scritti i bit di validità, 0 in tutti e otto i registri nel caso di TLB Flush, 1 per scrivere in uno degli otto registri puntati dal ContatoreModulo2 che a sua volta è un registro.

Ecco come si presenta la simulazione per intero, i dettagli di ogni passo del Testcase sono piu' avanti. Per ottenere la schermata seguente ho impostato Gtkwave in questo modo:

Su Gtkwave: View → Show filled High Values

> Oltre ai segnali che propone Gtkwave di default (sezione testcase1_MMU), ne sono stati aggiunti alcuni della sezione parteoperativa e sono esito_ma, r_indtabril, r_lineav#, r_linea#_ipl e r_linea#_ipf (# = 1, 2, ..., 8), della sezione partecontrollo, stato

Su Linux con KDE Plasma: Schermi → Scala lo schermo → Scala: 2 → OK → avvio Gtkwave



Gli indirizzi sono rappresentati in Esadecimale, le operazioni in Binario.

t = 1ns : La prima operazione richiesta dal processore e' il Flush della TLB, quindi OP = 10, viene passato l'indirizzo fisico 1000 della TABRIL del processo.

t = 43ns : Il registro TABRIL ora contiene l'indirizzo fisico 1000

t = 44ns : La MMU manda l'esito positivo di Flush TLB al processore

t = 68ns : Il processore richiede la lettura di un dato, quindi OP = 00, all'indirizzo logico 2001, si verifica Fault di TLB (esito_ma = 1)

t = 109ns : La MMU chiede la linea di cache TLB alla memoria M, quindi OPOUTM = 11,

INDOUTM = 1002 (che deriva dalla somma dell'indirizzo fisico di TABRIL 1000 e l'indirizzo pagina logica 2)

t = 137ns : La memoria M risponde alla richiesta della linea di tabril con esito=0 e indirizzo fisico presente su DATAINM che e' 2004

t = 176ns : La MMU manda la richiesta di lettura del dato alla cache C all'indirizzo fisico 2004001 (IPF = 2004 e OFFSET = 1) e al contempo scrive in linea di TLB l'indirizzo di pagina logica 2 e l'indirizzo di pagina fisica 2004

t = 202ns : La Cache C risponde alla MMU con il dato che e' presente in cache, quindi

DATAINC = 20

t = 241ns : La MMU manda il dato e l'esito al processore

t = 265ns : Il processore richiede la scrittura del dato 25, quindi OP = 01,

all'indirizzo logico 2008

t = 307ns : TLB-Hit, la MMU manda la richiesta alla cache C, quindi OPOUTC = 01,

INDOUTC = 2004008, DATAOUTC = 25

t = 333ns : La cache risponde alla MMU con l'esito di avvenuta scrittura

t = 374ns : La MMU manda l'esito positivo al processore

t = 398ns : Il processore manda la richiesta di lettura di un dato, quindi OP = 00, all'indirizzo logico 2018

t = 440ns : TLB-Hit, la MMU manda la richiesta alla cache C, quindi OPOUTC = 00,

INDOUTC = 2004018

t = 466ns : (non visibile, viene simulato solo il tempo impiegato a trattare il Fault di Cache), quindi Fault di cache C, la cache C manda la richiesta alla memoria M, la memoria M risponde alla cache C, il dato viene memorizzato in cache C

t = 542ns : La cache C risponde alla MMU con il dato, quindi DATAINC = 34

t = 572ns : La MMU manda il dato e l'esito al processore, quindi DATAOUT = 34

t = 596ns : Il processore manda la richiesta di scrittura del dato 76, quindi OP = 01, all'indirizzo logico 5100 (IPL=5, OFF=100), si verifica Fault di TLB (esito_ma = 1)

t = 638ns : La MMU chiede la linea di cache TLB alla memoria M, quindi OPOUTM = 11

t = 666ns : La memoria M risponde alla richiesta della linea di tabril con esito=0 e indirizzo fisico presente su DATAINM che e' 2090

t = 704ns : La MMU manda la richiesta di scrittura del dato alla cache C all'indirizzo fisico 2090100 (IPF = 2090 e OFFSET = 100) e al contempo scrive in linea di TLB l'indirizzo di pagina logica 5 e l'indirizzo di pagina fisica 2090

t = 730ns : La cache risponde alla MMU con esito di scrittura positivo

t = 771ns : La MMU manda l'esito al processore

t = 795ns : Il processore manda la richiesta di lettura di un dato, quindi OP = 00, all'indirizzo logico 3008, si verifica Fault di TLB (esito_ma = 1)

t = 836ns : La MMU chiede la linea di cache TLB alla memoria M, quindi OPOUTM = 11,

INDOUTM = 1003

t = 864ns : La memoria M risponde con un esito = 1, quindi Fault di Pagina

t = 902ns : La MMU passa l'esito al processore

t = 926ns : Il processore richiede il Flush della TLB, quindi OP = 10, viene passato l'indirizzo fisico 8000 della TABRIL del processo

t = 968ns : La MMU manda l'esito al processore e il registro INDTABRIL e' aggiornato con il nuovo indirizzo

t = 1us : Fine simulazione