

TP : Réalisation d'une machine algorithmique

5.3 Modification de l'algorithme

1.

```
r = a
while (r >= n) {
    r = r - n;
}
```

Pour calculer le modulo d'un entier en n'utilisant que des soustractions, l'idée est de retirer répétitivement la valeur n de l'entier a jusqu'à ce que le résultat soit strictement inférieur à n. Autrement dit, si l'on souhaite calculer $r = a \bmod n$, on peut procéder de la manière suivante :

Cette méthode garantit que le résultat final r satisfait $0 \leq r < n$.

2.

```
#include <stdio.h>
#include <stdlib.h>

void main(int argc, char * argv[])
{
    unsigned long int m, exp, n;
    unsigned long int code, temp;

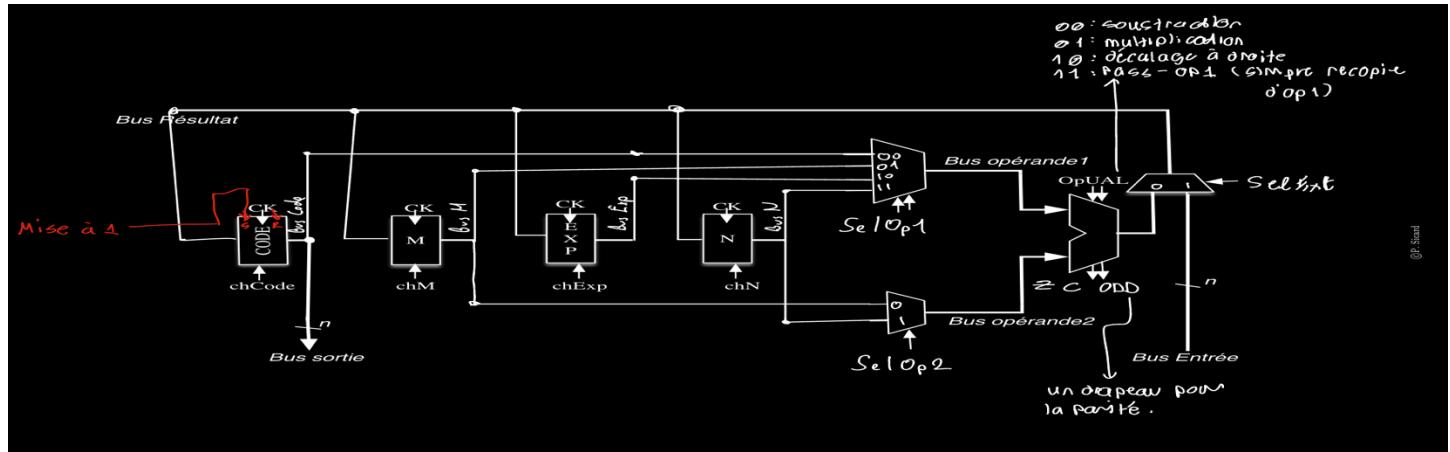
    if (argc != 4) exit(0);
    sscanf(argv[1], "%ld", &m);
    sscanf(argv[2], "%ld", &exp);
    sscanf(argv[3], "%ld", &n);

    code = 1;
    while (exp > 0) // exp != 0 donc flag (not Z)
    {
        if ((exp & 1) > 0) // (LSB == 1) Last Significant Bit
        {
            // code = (code * m) % n, calculé par soustractions répétées
            code = code * m;
            while (code >= n) //while C
                code = code - n;
        }

        exp = exp >> 1; // maj flag Z
        // m = (m * m) % n, calculé par soustractions répétées
        m = m * m;
        while (m >= n) //while C
            m = m - n;
    }
    printf("Resultat chiffrement %ld\n", code);
}
```

5.4 La partie opérative

2.



3.

```
-- Instruction |Op[1] |Op[2] |Opération |Résultat chargé dans
-- code < code * m |CODE |M |MUL |CODE
-- m < m * m |M |M |MUL |M
-- code < code < n |CODE |N |SUB |CODE
-- exp < exp >>1 |EXP | |SHR1 |EXP

-- Il nous faut donc quatre traitements dans l'UAL: PASS[OP1], SUB, MUL, SHR1 (shift logique d'un bit vers la droite).

-- Codage des commandes opUAL
-- On garde un codage compact sur 2 bits:

-- opUAL[1:0]|Signification |Drapeaux générés
-- 00 | SUB(Op[1] - Op[2]) | Z, C (emprunt), N
-- 01 | MUL(Op[1] x Op[2]) | Z
-- 10 | SHR1 (décalage à droite de Op[1]) | Z, C (bit perdu)
-- 11 | PASS[OP1] (simple recopie d'Op[1]) | Z (=0?), Odd
```

```
-- Définition d'un soustracteur n bits
include "../multv6base2.lus"
include "../entierv6.lus"
include "../UALCryptoV6.lus"

function subnbits<<const n: int>>(A, B: bool^n) returns (S: bool^n; Borrow: bool);
var
    B_out: bool^n;
let
    -- Complément à 2 de B pour la soustraction (A - B = A + (-B + 1))
    B_out = map<<not, n>>(B); -- Complément à 1 en inversant les bits
    (S, Borrow) = add_n_bits<<n>>(A, B_out, true); -- +1 pour le complément à 2
tel;

-- Notre définition de l'UAL
function UAL<<const n: int>>(
    OpUAL: bool^2;          -- Commande de l'UAL (2 bits)
    Op1, Op2: bool^n;       -- Opérandes (n bits)
)
returns (
    C, Z, Odd: bool;        -- Drapeaux de statut
    Result: bool^n           -- Résultat de l'UAL (n bits)
);
var
    Mult_Result: bool^(2*n);  -- Résultat de la multiplication (2*n bits)
    Sub_Result: bool^n;       -- Résultat de la soustraction
    Borrow: bool;             -- Retenue de la soustraction
    Shift_Result: bool^n;     -- Résultat du décalage
    Exp_Is_Odd: bool;         -- EXP & 1 = 1 ?
let
    -- Multiplication (code * m)
    -- On évite de faire le calcul selon OpUAL
    Mult_Result = multnbits<<n>>(Op1, Op2);
    -- Soustraction (pour le modulo)
    (Sub_Result, Borrow) = subnbits<<n>>(Op1, Op2);
    -- Décalage à droite (exp >> 1)
    Shift_Result = shiftl<<n>>(Op1, false);

    -- Résultat final (selon OpUAL)
    Result = map <<mux4, n>>(
        OpUAL[1]^n, OpUAL[0]^n,
        Sub_Result, Mult_Result[0..n-1], Shift_Result, Op1
    );
    -- Calcul des flags
    Z = (Result = false^n);   -- Z: EXP = 0 ?
    Exp_Is_Odd = Result[0];      -- EXP & 1 = 1 ?

    C = not Borrow;            -- Emprunt de la soustraction
    Odd = Exp_Is_Odd;          -- EXP & 1 = 1 ?
```

```
| tel

-- Exemple d'utilisation de l'UAL comme l'exemple donné dans le UALcryptoV6
function ualaff_2<<const n:int>>( A , B : bool^n ; OpUAL : bool^2 )
returns ( C,Z,Odd : bool ; T : bool^n; AE,BE,R:int );
let
(C,Z,Odd,T )=UAL<<n>>( OpUAL, A , B );

R=entiernat<<n>>(T);
AE=entiernat<<n>>(A);
BE=entiernat<<n>>(B);

tel;
node ualinstr_2=ualaff<<a>>;
```

4.

```
-- Version sans CK
-----
include "../memoire.lus"

node RegN <<const n : int>>
  (ent : bool^n;      -- bus de données à charger
  char : bool;        -- signal de chargement commun
  reset : bool;       -- reset synchrone (met tous les bits à 0)
  set : bool)         -- set asynchrone (met tous les bits à 1)
returns
  (sort      : bool^n); -- bus de sortie (état du registre)
let
  -- map applique n fois le noeud bascule,
  -- en dupliquant char, reset et set en vecteurs de taille n
  sort = map <<bascule; n>>
    (ent,
     | char^n,      -- ent[i]
     | reset^n,    -- reset pour chaque bit
     | set^n,      -- set pour chaque bit
     );
tel
-- Pour juste 4 bits on instancie :
node Reg4 = RegN<<4>>

-- Version avec CK
-----
-- Registre n bits avec chargement conditionnel
node registre_n_bits<<const n: int>>
  CK: bool;          -- Horloge globale du circuit --On n'utilise pas dans le code juste pour la clarté
  chM: bool;         -- Signal de chargement (connecté à CHAR des bascules)
  reset: bool;       -- Reset global (actif à 1)
  set: bool;         -- Set global (actif à 1)
  D: bool^n          -- Donnée d'entrée
)
returns
  O: bool^n          -- Sortie du registre
);
let
  -- Génération des n bascules en parallèle
  Q = map<<bascule; n>>(D, chM^n, reset^n, set^n);

  -- Connexion implicite de l'horloge :
  -- Le signal CK est implicite dans l'opérateur -> et pre de la bascule
tel;

--exemple
-- Registre M de 8 bits
node registre_M(
  Clock: bool;      -- Horloge --On n'utilise pas dans le code juste pour la clarté
  load_M: bool;      -- Commande de chargement
  rst: bool;         -- Reset
  data_in: bool^8   -- Bus d'entrée
)
returns
  M_out: bool^8     -- Sortie
);
let
  M_out = registre_n_bits<<8>>(Clock, load_M, rst, false, data_in);
  -- set mis à false car on n'initialise pas le registre M avec 1
tel;
```

5.

```
-- 1) On inclut d'abord tout ce qu'il faut :  
--     * Les bascules les registres et UAL à n bits  
include ".../memoire.lus"  
include "3.lus"  
include "4.lus"  
include ".../entierV6.lus"  
include ".../multv6base2.lus"  
include ".../UALCryptoV6.lus"  
  
-----  
-- 2) Node PO générique à n bits -----  
  
node PO <>const n:int>(<  
    Clock : bool;      -- horloge  
    Reset : bool;       -- reset synchrone =1  
    selExt : bool;      -- 1 = charger depuis busIn (initialisation), 0 = recirculer UAL  
    busIn : bool^n;     -- bus d'entrée externe (M, EXP ou N)  
    chCode : bool;      -- load CODE  
    chM : bool;         -- load M  
    chExp : bool;       -- load EXP  
    chN : bool;         -- load N  
    selOp1 : bool^2;    -- code 00..11 pour choisir CODE/M/EXP/N en op1  
    selOp2 : bool;      -- code 0 1 pour choisir M/N en op2  
    opUAL : bool^2;     -- commande UAL (00=sub, 01=mul, 10=shrl, 11=pass)  
    mise_1 : bool        -- mise à 1 de CODE (pour le premier tour)  
>)  
returns (<  
    busOut : bool^n;   -- seule sortie visible : CODE  
    Z, C, Odd : bool;   -- flags Zero et Carry (emprunt)  
    CodeInt, MInt, NInt, ExpInt : int -- pour le debug  
>);  
var  
    CODE, M, EXP, Nreg : bool^n; -- registres internes  
    op1, op2 : bool^n; -- bus opérandes vers l'UAL  
    resUAL : bool^n; -- résultat de l'UAL  
    Borrow, Zflag : bool; -- emprunt, zero renvoyés par l'UAL  
    setVecCode: bool^n; -- vecteur de set pour CODE  
let  
  
    setVecCode = [true] | false^(n-1); -- vecteur à 1 seulement en position 0  
    -- CODE = if mise_1 then [true] | registre_n_bits<>(Clock, chCode, Reset, false,  
    --           if selExt then busIn else resUAL)  
    --           else registre_n_bits<>(Clock, chCode, Reset, false,  
    --           if selExt then busIn else resUAL);  
  
    -- Instanciation "à la main" de CODE via map<>bascule;n>> pour passer setVecCode  
    CODE = map <>bascule; n>>(<  
        -- ent : si phase init (selExt) on prend busIn, sinon resUAL  
        if selExt then busIn else resUAL,  
        -- char : chCode répliqué  
        chCode^n,  
        -- reset : Reset répliqué  
        Reset^n,  
        -- set : notre setVecCode  
        if mise_1 then setVecCode else false^n  
>);  
  
    -- On charge CODE en premier, puis les autres registres  
    -- peut être false au lieu de Reset and not mise_1  
    M = registre_n_bits<>(Clock, chM, Reset and not mise_1, false,
```

```
-- On charge CODE en premier, puis les autres registres
--peut etre false au lieu de Reset and not mise_1
M    = registre_n_bits<<n>>(Clock, chM, Reset and not mise_1, false,
|     if selExt then busIn else resUAL);
EXP = registre_n_bits<<n>>(Clock, chExp, Reset and not mise_1, false,
|     if selExt then busIn else resUAL);
Nreg = registre_n_bits<<n>>(Clock, chN, Reset and not mise_1, false,
|     if selExt then busIn else resUAL);

-----
-- 4) Multiplexeurs 4:1 et 2:1 pour selectionner les deux operandes
op1 = map <<mux4; n>>(
|   selOp1[1]^n, selOp1[0]^n,
|   CODE, M, EXP, Nreg);

op2 = map <<mux1; n>>(
|   selOp2^n,
|   Nreg, M);

-----
-- 5) Appel de l'UAL (issue de 3.lus) :
--     UAL(opUAL, op1, op2) returns (C, Z, Odd, Result)
(Borrow, Zflag, Odd, resUAL) = UAL<<n>>(opUAL, op1, op2);

-----
-- 6) Bus de sortie et drapeaux
busOut = CODE; -- on n'exporte que CODE
C      = not Borrow; -- emprunt de la soustraction
Z      = Zflag; -- resultat nul ?

--pour le voir mieux --debug
CodeInt = entiernat<<n>>(CODE);
MInt = entiernat<<n>>(M);
NInt = entiernat<<n>>(Nreg);
ExpInt = entiernat<<n>>(EXP);

tel

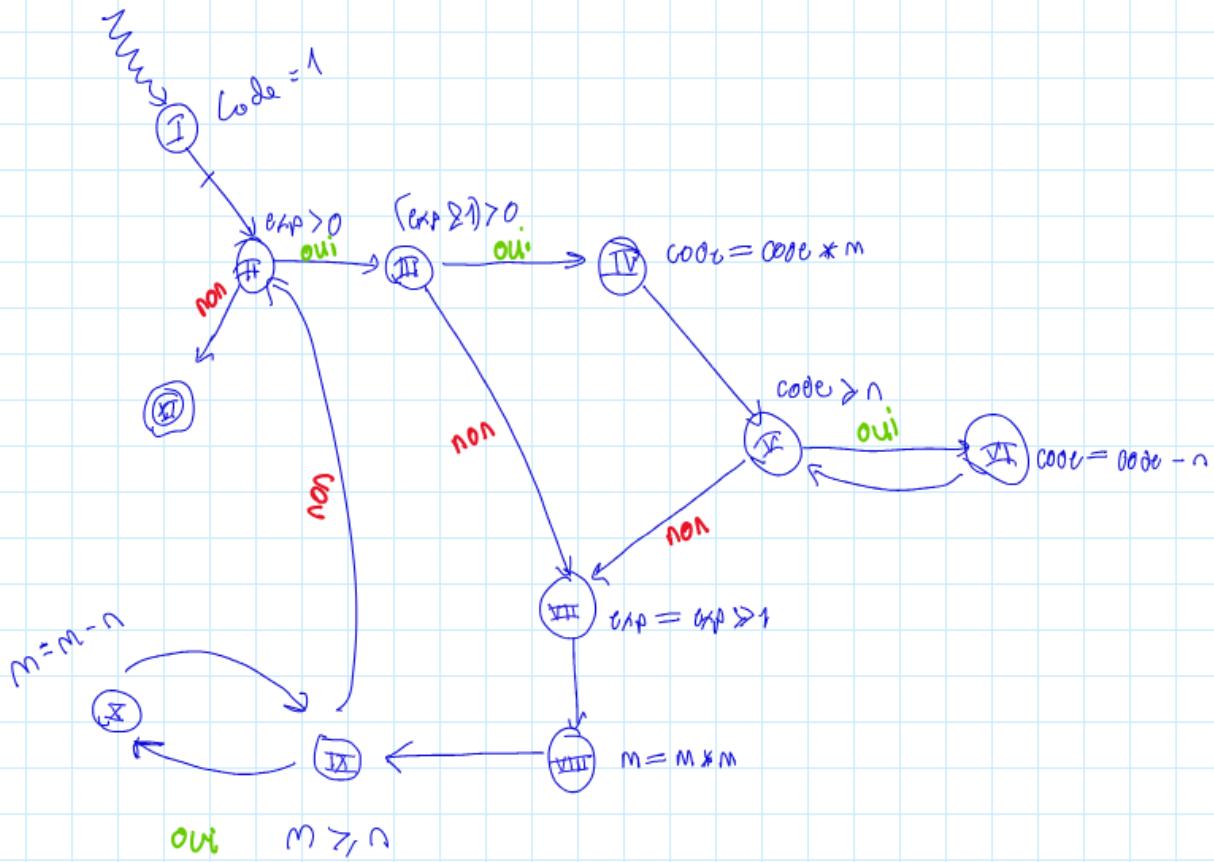
-- Instanciation pour n=8 bits, par exemple
node P08 = P0<<8>>;
```

5.5 La partie contrôlé

1.

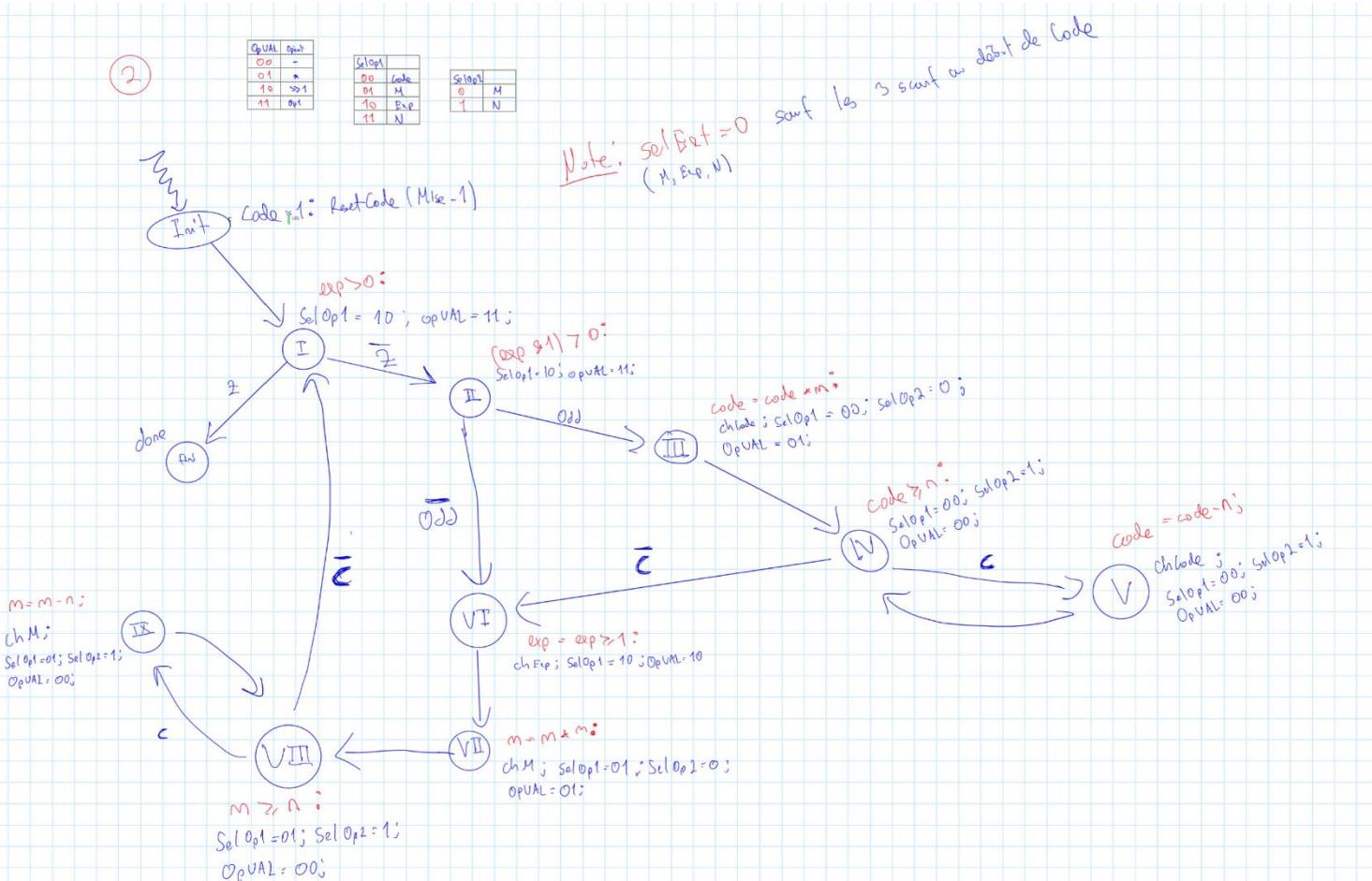
L'automate PC

①



2.

N.B : On n'indique pas explicitement les valeurs qui ne sont pas utilisées dans l'état en question.



3.

```
-----  
include "../memoire.lus"  
  
node PC(  
    CLK : bool;      -- Horloge -- ce n'est pas utilisé dans le code , juste pour la simulation  
    RESET : bool;     -- Reset synchrone (actif haut)  
    Zexp : bool;      -- Flag EXP = 0  
    C : bool;         -- Flag Carry (pour modulo)  
    Odd : bool        -- Flag EXP impair (LSB)  
)  
returns (  
    -- Signaux de chargement  
    chExp, chCode, chN, chM : bool;  
    -- Sélection entrée  
    selExt : bool;  
    -- Contrôle multiplexeurs  
    selOp1 : bool^2;  
    selOp2 : bool;  
    -- Commande UAL  
    opUAL : bool^2;  
    -- Indicateur de fin  
    done : bool;  
    -- signale de mise à 1  
    mise_1 : bool;  
    -- Signaux de débogage  
    etat_courant : bool^11  
)  
var  
    st, st_next : bool^11;    -- vecteurs un-among-11  
  
let  
    st[0] = bascule(st_next[0], true, false, RESET); --initial;  
    st[1..10] = map <<bascule;10>>(  
        st_next[1..10], -- Entrée D  
        true^10,          -- CHAR toujours actif  
        RESET^10,          -- Reset synchrone  
        false^10           -- Set initial  
)  
  
-----  
-- 3) Machine à états (version optimisée)  
st_next = [  
    -- État 0 (initCode) → État 1 (loopTest)  
    RESET,  
  
    -- État 1 : exp>0  
    st[0] or (st[8] and not C),  
  
    -- État 2 : (exp & 1) > 0  
    st[1] and not Zexp ,  
  
    -- État 3: code = code * m  
    st[2] and Odd,  
  
    -- État 4 : code >= n  
    st[3] or st[5],  
  
    -- État 5: cpde = code - n;  
    st[4] and C,  
  
    -- État 6: exp = exp >> 1  
    (st[2] and not Odd) or (st[4] and not C),  
];
```

```
-- État 6: exp = exp >> 1
(st[2] and not Odd) or (st[4] and not C),
-- État 7 : m = m*n
st[6],
-- État 8 : m>= n
st[7] or st[9],
-- État 9 : m = m - n
st[8] and C,
-- État 10 : m = m >> 1
st[1] and Zexp
];

-----
-- 4) Génération des signaux de contrôle (version clarifiée)
mise_1 = st[0];
-- Chargements initiaux
chCode = st[0] or st[3] or st[5] ;
chExp = st[6];
chM = st[7] or st[9];

-- On n'utilise pas selExt, chN comme nous ne faisons que les affectations au début de trois scanf
selExt = st[0];
chN = false;

-- Sélection entrée externe pendant l'init
--selExt = st[0];

-- Contrôle des multiplexeurs (table de vérité optimisée)
selOp1 = [
    -- Bit 0: 1 pour M
    st[7] or st[8] or st[9],
    -- Bit 1: 1 pour Exp
    st[1] or st[2] or st[6]
];
-- selOp2 = 1 pour N dans multiply(5)/shift(6)/square(7)
selOp2 = st[4] or st[5] or st[8] or st[9];

-- Commande UAL (nouveau codage plus lisible)
opUAL = [
    st[1] or st[2] or st[3] or st[7],
    st[1] or st[2] or st[6]
];
-- Signal de fin
done = st[10];

-- Nouveau: visualisation de l'état courant pour débogage
etat_courant = st;
tel
```

L'initialisation du circuit repose sur un **reset synchrone actif-haut** partagé par la partie contrôle (PC) et la partie opérative (PO). Au premier front montant de l'horloge après assertion de **RESET=1**, la bascule st[0] reçoit sa ligne **set** et passe à 1, tandis que toutes les autres bascules st[1..10] sont simultanément remises à 0 grâce à leur ligne **reset**. Cet unique "1" dans le vecteur d'état identifie l'**état 0** de la machine de contrôle et, via le décodage de cet état, active automatiquement les signaux **mise_1 = 1**, **selExt = 1**, **chCode = 1** (ainsi que, si nécessaire, **chM**, **chExp** et **chN**).

Dans la PO, **mise_1=1** alimente la ligne **set** du registre CODE avec un vecteur [1,0,...,0], ce qui lui donne la valeur 1, tandis que **selExt=1** et **chCode=1** ordonnent à tous les registres (CODE, M, EXP, N) de charger depuis le bus d'entrée leurs valeurs initiales. Dès que **RESET** retombe à 0, la machine conserve ce vecteur d'état "1 000...0" un cycle puis passe à l'état 1, lançant ainsi l'exécution normale de l'algorithme de chiffrement.

5. La réunion de deux parties PC et PO : top.lus

```
-- top.lus
-- dépendances : PC.lus, PO.lus (votre S.lus renommé), multv6base2.lus, UALCryptoV6.lus
include "PC.lus"
include "../PO/S.lus"

node top_debug <>const n:int>>(
begin_ : bool;    -- checkbox "begin"
chM : bool;    -- checkbox "chM"
chEXP : bool;   -- checkbox "chEXP"
chN : bool;    -- checkbox "chN"
input : int      -- slider "input" (valeur entière à charger)
)
returns (
m : int;    -- affichage registre M
exp : int;   -- affichage registre EXP
nreg : int;   -- affichage registre N
end_ : bool;  -- flag "end_"
q : int;     -- copie de "code" si vous voulez un label supplémentaire
-- Signaux de débogage
etat_courant : bool?1;
Sig_chExp : bool;
Sig_chCode : bool;
Sig_chN: bool;
Sig_chM : bool;
| mise_1 : bool;
selOp1 : bool^2;  -- commande UAL (issue de PC)
selOp2 : bool;
opUAL : bool^2;  -- commande UAL (issue de PC)
Sig_C : bool;    -- flag Carry
Sig_Z : bool;    -- flag Zero
Sig_Odd : bool;  -- flag Odd
);
var
Z, C, Odd : bool;    -- flags retour UAL
--selOp1 : bool^2;  -- commande UAL (issue de PC)
--selOp2 : bool;
--opUAL : bool^2;
-- vecteur de bits issu du slider
busIn : bool^n;
-- bus interne de sortie PO
code_bits : bool^n;
reset : bool;    -- reset PO

code : int; -- code de sortie
--signale de mise à 1

Sig_chM_interne : bool;
Sig_chExp_interne : bool;
Sig_chN_interne : bool;
Sig_chCode_interne : bool;

-- vecteurs de bits issus des registres
selExt : bool;  -- 1 = charger depuis busIn (initialisation), 0 = recirculer UAL
let
```

```
64 let
65   -- conversion de l'entier 'input' en bus de n bits (LSB à l'indice 0)
66   busIn = natier16(input);
67
68   -- Partie opérative (PO) : on charge manuellement via 'begin',chM,chEXP,chN
69   ( code_bits, Z, C, Odd ,code, m, nreg, exp ) =
70     PO<<n>>(false,
71       | reset,      -- reset PO = false (on ne reset pas ici)
72       | not begin,  -- selExt = !begin
73       | busIn,      -- bus d'entrée
74       | Sig_chCode_interne, -- chCode = begin (on force reset à 1 pour initialiser CODE=1)
75       | Sig_chM_interne,  -- chM    = checkbox chM
76       | Sig_chExp_interne, -- chExp   = checkbox chEXP
77       | Sig_chN_interne,  -- chN    = checkbox chN
78       | selOp1,      -- commande alu issue de PC
79       | selOp2,
80       | opUAL,
81       | mise_1
82     );
83
84   Sig_chExp_interne = mux1(begin, Sig_chExp, chEXP);
85   Sig_chN_interne = mux1(begin, Sig_chN, chN);
86   Sig_chM_interne = mux1(begin, Sig_chM, chM);
87   Sig_chCode_interne = mux1(begin, Sig_chCode, false);
88
89   reset = (not pre begin) and begin;
90
91
92   -- Partie contrôle (PC) : on reset PC quand begin=0→1, on récupère
93   -- uniquement les commandes sorties; on ignore ici selExt/chCode/ch*
94   ( Sig_chExp, Sig_chCode, Sig_chN, Sig_chM,
95     selExt, selOp1, selOp2,
96     opUAL, end_, mise_1, etat_courant ) =
97     PC(false,
98       | not begin, -- RESET_PC = 1 quand begin passe à 0→1 (synchronous)
99       | Z, C, Odd );
100
101
102   -- sorties finales
103   q   = entiernat<<n>>(code_bits); -- alias si besoin
104   Sig_C = C;
105   Sig_Z = Z;
106   Sig_Odd = Odd;
107
108   tel;
109
110   ----
111   --utilisation top_debug
112   -- inst_top16[lus
113
114   -- On fixe n = 16
115   node top16(
116     begin  : bool;
117     chM   : bool;
118     chEXP : bool;
119     chN   : bool;
120     input  : int
121   )
122   returns (
123     m      : int;
```

```
123 |     input  : int
124 | }
125 | returns (
126 |     m      : int;
127 |     exp   : int;
128 |     n      : int;
129 |     end_   : bool;
130 |     q      : int;
131 |     -- Signaux de débogage
132 |     etat_courant : bool^11;
133 |     Sig_chExp : bool;
134 |     Sig_chCode : bool;
135 |     Sig_chN: bool;
136 |     Sig_chM : bool;
137 |     | mise_1 : bool;
138 |     selOp1 : bool^2;    -- commande UAL (issue de PC)
139 |     selOp2 : bool;
140 |     opUAL : bool^2;    -- commande UAL (issue de PC)
141 |     Sig_C : bool;      -- flag Carry
142 |     Sig_Z : bool;      -- flag Zero
143 |     Sig_Odd : bool;    -- flag Odd
144 );
145 let
146 -- On déplie top_debug avec n=8
147 | (m, exp, n, end_, q, etat_courant,Sig_chExp,Sig_chCode,Sig_chN,Sig_chM, mise_1, selOp1, selOp2, opUAL, Sig_C, Sig_Z, Sig_Odd) =
148 | top_debug<<16>>(begin, chM, chEXP, chN, input);
149 tel;
150
151
152
153 -- natier :
154
155 +-- natier8.lus
156 -- Convertit un entier i en vecteur de 8 bits (LSB en position 0)
157 function natier8(i:int) returns (b: bool^8);
158 let
159     b[0] = (i mod 2) = 1;
160     b[1] = ((i / 2) mod 2) = 1;
161     b[2] = ((i / 4) mod 2) = 1;
162     b[3] = ((i / 8) mod 2) = 1;
163     b[4] = ((i / 16) mod 2) = 1;
164     b[5] = ((i / 32) mod 2) = 1;
165     b[6] = ((i / 64) mod 2) = 1;
166     b[7] = ((i / 128) mod 2) = 1;
167 tel;
168
169 -- Convertit un entier i en vecteur de 16 bits (LSB en position 0)
170 function natier16(i:int) returns (b: bool^16);
171 let
172     b[0] = (i mod 2) = 1;
173     b[1] = ((i / 2) mod 2) = 1;
174     b[2] = ((i / 4) mod 2) = 1;
175     b[3] = ((i / 8) mod 2) = 1;
176     b[4] = ((i / 16) mod 2) = 1;
177     b[5] = ((i / 32) mod 2) = 1;
178     b[6] = ((i / 64) mod 2) = 1;
179     b[7] = ((i / 128) mod 2) = 1;
180
181     b[8] = ((i / 256) mod 2) = 1;
182     b[9] = ((i / 512) mod 2) = 1;
183     b[10] = ((i / 1024) mod 2) = 1;
184     b[11] = ((i / 2048) mod 2) = 1;
185     b[12] = ((i / 4096) mod 2) = 1;
186     b[13] = ((i / 8192) mod 2) = 1;
187     b[14] = ((i / 16384) mod 2) = 1;
188     b[15] = ((i / 32768) mod 2) = 1;
189 tel;
190
```

Jeux de Tests

Avec $p=3$ et $q=11$: (13,33), (17,33) et $m = 31$:

```
[~/Desktop/repertoire/ArchMat/Real_mach_alg]
base arhis > ./chiffrement 31 13 33
Resultat chiffrement 25
◆
[~/Desktop/repertoire/ArchMat/Real_mach_alg]
base arhis >
```

Luciole 1.75 - top16

Files Options Clocks Tools Inputs

<input type="checkbox"/> begin	m	exp	n	<input type="checkbox"/> end_	q
<input type="checkbox"/> chM	?	?	?		?
<input type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2	etat_courant_3	etat_courant_4
<input type="checkbox"/> chN	etat_courant_5	etat_courant_6	etat_courant_7	etat_courant_8	etat_courant_9
input	etat_courant_10	Sig_chExp	Sig_chCode	Sig_chN	Sig_chM
0	mise_1	selOp1_0	selOp1_1	selOp2	opUAL_0
	opUAL_1	Sig_C	Sig_Z	Sig_Odd	
		Step			

Luciole 1.75 - top16

Files Options Clocks Tools Inputs

<input type="checkbox"/> begin	m	exp	n	<input type="checkbox"/> end_	q
<input checked="" type="checkbox"/> chM	0	0	0		0
<input type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2	etat_courant_3	etat_courant_4
<input type="checkbox"/> chN	etat_courant_5	etat_courant_6	etat_courant_7	etat_courant_8	etat_courant_9
input	etat_courant_10	Sig_chExp	Sig_chCode	Sig_chN	Sig_chM
31	mise_1	selOp1_0	selOp1_1	selOp2	opUAL_0
	opUAL_1	Sig_C	Sig_Z	Sig_Odd	
		Step			

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 2

5

<input type="checkbox"/> begin	m	exp	n	end_	q
<input checked="" type="checkbox"/> chM	31	65535	65535		65535
<input type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2	etat_courant_3	etat_courant_4
<input type="checkbox"/> chN	etat_courant_5	etat_courant_6	etat_courant_7	etat_courant_8	etat_courant_9
input	etat_courant_10	Sig_chExp	Sig_chCode	Sig_chN	Sig_chM
31	mise_1	selOp1_0	selOp1_1	selOp2	opUAL_0
		opUAL_1	Sig_C	Sig_Z	Sig_Odd

6 Step

7 - ((i / 4) mod 2) = 1:

8 Luciole 1.75 - top16

9

10 Files Options Clocks Tools Inputs 5

11

<input type="checkbox"/> begin	m	exp	n	end_	q
<input type="checkbox"/> chM	31	13	65535		65535
<input checked="" type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2	etat_courant_3	etat_courant_4
<input type="checkbox"/> chN	etat_courant_5	etat_courant_6	etat_courant_7	etat_courant_8	etat_courant_9
input	etat_courant_10	Sig_chExp	Sig_chCode	Sig_chN	Sig_chM
13	mise_1	selOp1_0	selOp1_1	selOp2	opUAL_0
	opUAL_1	Sig_C	Sig_Z	Sig_Odd	

12 Step

13 - ((i / 4) mod 2) = 1:

14 Luciole 1.75 - top16

15

16 Files Options Clocks Tools Inputs 7

17

<input type="checkbox"/> begin	m	exp	n	end_	q
<input type="checkbox"/> chM	31	13	33		65535
<input type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2	etat_courant_3	etat_courant_4
<input checked="" type="checkbox"/> chN	etat_courant_5	etat_courant_6	etat_courant_7	etat_courant_8	etat_courant_9
input	etat_courant_10	Sig_chExp	Sig_chCode	Sig_chN	Sig_chM
33	mise_1	selOp1_0	selOp1_1	selOp2	opUAL_0
	opUAL_1	Sig_C	Sig_Z	Sig_Odd	

18 Step

19 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 8
5
6 begin m exp n end_ q
7  chM 31 13 33 1
8  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
12 opUAL_1 Sig_C Sig_Z Sig_Odd
13 Step
14
15 -((i / 4) mod 2) = 1:
16 Luciole 1.75 - top16
17
18 Files Options Clocks Tools Inputs 9
19
20 begin m exp n end_ q
21  chM 31 13 33 1
22  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
23  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
24 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
25 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
26 opUAL_1 Sig_C Sig_Z Sig_Odd
27 Step
28
29 -((i / 4) mod 2) = 1:
30 Luciole 1.75 - top16
31
32 Files Options Clocks Tools Inputs 10
33
34 begin m exp n end_ q
35  chM 31 13 33 1
36  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
37  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
38 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
39 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
40 opUAL_1 Sig_C Sig_Z Sig_Odd
41 Step
42
43 -((i / 4) mod 2) = 1:
```

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 11

5

6 begin m exp n end_ q

7 chM 31 13 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1 :

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 12

18

19 begin m exp n end_ q

20 chM 31 13 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1 :

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 13

31

32 begin m exp n end_ q

33 chM 31 6 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1 :

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 14
5
6 begin m exp n end_ q
7 chM 961 6 33 31
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13
11 selOp1_0 selOp1_1 selOp2 opUAL_0
12 Sig_C Sig_Z Sig_Odd
13 Step
14
15
16
17
18
19
20
21 - ((i / 4) mod 2) = 1:
```

The screenshot shows three vertically stacked windows of the Luciole 1.75 software interface, each titled "Luciole 1.75 - top16". The windows are numbered 14, 15, and 16 from top to bottom. Each window has a menu bar with "Files", "Options", "Clocks", "Tools", and "Inputs". A status bar at the bottom of each window displays the number 14, 15, or 16 respectively. The main area contains a table with columns for "begin", "m", "exp", "n", "end_", and "q". The "begin" row has a checked checkbox for "begin" and an unchecked checkbox for "chM". The "m" row contains the value "961". The "exp" row contains the value "6". The "n" row contains the value "33". The "end_" row contains the value "31". The "q" row contains the value "q". Below the table, there is a list of state names: "etat_courant_0", "etat_courant_1", "etat_courant_2", "etat_courant_3", "etat_courant_4", "etat_courant_5", "etat_courant_6", "etat_courant_7", "etat_courant_8", "etat_courant_9", "etat_courant_10". To the right of these state names are several signal names: "Sig_chExp", "Sig_chCode", "Sig_chN", "Sig_chM", "Sig_C", "Sig_Z", "Sig_Odd", "opUAL_0", "selOp1_0", "selOp1_1", "selOp2", "opUAL_1", and "mise_1". The "SelOp" and "OpUAL" rows are highlighted in red, while the others are white. The bottom of each window has a "Step" button.

2

3 Luciole 1.75 - top16

4

5 Files Options Clocks Tools Inputs 17

6 begin m exp n end_ q

7 chM 928 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 18

18 begin m exp n end_ q

19 chM 895 6 33 31

20 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

21 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

22 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

23 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

24 opUAL_1 Sig_C Sig_Z Sig_Odd

25 Step

26 - ((i / 4) mod 2) = 1:

27 Luciole 1.75 - top16

28

29 Files Options Clocks Tools Inputs 19

30 begin m exp n end_ q

31 chM 895 6 33 31

32 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

33 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

34 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

35 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

36 opUAL_1 Sig_C Sig_Z Sig_Odd

37 Step

38 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 20

5

6 begin m exp n end_ q

7 chM 862 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 21

18

19 begin m exp n end_ q

20 chM 862 6 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 22

31

32 begin m exp n end_ q

33 chM 829 6 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 23

5

6 begin m exp n end_ q

7 chM 829 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 24

18

19 begin m exp n end_ q

20 chM 796 6 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 25

31

32 begin m exp n end_ q

33 chM 796 6 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 26

5

6 begin m exp n end_ q

7 chM 763 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 27

18

19 begin m exp n end_ q

20 chM 763 6 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 28

31

32 begin m exp n end_ q

33 chM 730 6 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 29
5
6 begin m exp n end_ q
7 chM 730 6 33 31
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13
11 selOp1_0 selOp1_1 selOp2
12 opUAL_1 Sig_C Sig_Z Sig_Odd
13
14 Step
15
16 -((i / 4) mod 2) = 1:
17
18 Luciole 1.75 - top16
19
20 Files Options Clocks Tools Inputs 30
21
22 begin m exp n end_ q
23 chM 697 6 33 31
24 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
25 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
26 input 13
27 selOp1_0 selOp1_1 selOp2
28 opUAL_1 Sig_C Sig_Z Sig_Odd
29
30 Step
31
32 -((i / 4) mod 2) = 1:
33
34 Luciole 1.75 - top16
35
36 Files Options Clocks Tools Inputs 31
37
38 begin m exp n end_ q
39 chM 697 6 33 31
40 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
41 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
42 input 13
43 selOp1_0 selOp1_1 selOp2
44 opUAL_1 Sig_C Sig_Z Sig_Odd
45
46 Step
47
48 -((i / 4) mod 2) = 1:
```

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 32
5
6 begin m exp n end_ q
7  chM 664 6 33 31
8  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
11 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
12  opUAL_1 SelOp2 Sig_C Sig_Z Sig_Odd
13 Step
14
15 -((i/4) mod 2) = 1:
16 Luciole 1.75 - top16
17
18 Files Options Clocks Tools Inputs 33
19
20 begin m exp n end_ q
21  chM 664 6 33 31
22  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
23  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
24 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
25 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
26  opUAL_1 SelOp2 Sig_C Sig_Z Sig_Odd
27 Step
28
29 -((i/4) mod 2) = 1:
30 Luciole 1.75 - top16
31
32 Files Options Clocks Tools Inputs 34
33
34 begin m exp n end_ q
35  chM 631 6 33 31
36  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
37  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
38 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
39 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
40  opUAL_1 SelOp2 Sig_C Sig_Z Sig_Odd
41 Step
42
43 -((i/4) mod 2) = 1:
```

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 35

5

6 begin m exp n end_ q

7 chM 631 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 36

18

19 begin m exp n end_ q

20 chM 598 6 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 37

31

32 begin m exp n end_ q

33 chM 598 6 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 38
5
6 begin m exp n end_ q
7 chM 565 6 33 31
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
11 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
12 13 opUAL_1 Sig_C Sig_Z Sig_Odd
13 Step
14
15 -((i / 4) mod 2) = 1:
16 Luciole 1.75 - top16
17
18 Files Options Clocks Tools Inputs 72
19
20 begin m exp n end_ q
21 chM 4 6 33 31
22 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
23 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
24 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
25 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
26 13 opUAL_1 Sig_C Sig_Z Sig_Odd
27 Step
28
29 -((i / 4) mod 2) = 1:
30 Luciole 1.75 - top16
31
32 Files Options Clocks Tools Inputs 73
33
34 begin m exp n end_ q
35 chM 4 6 33 31
36 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
37 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
38 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
39 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
40 13 opUAL_1 Sig_C Sig_Z Sig_Odd
41 Step
42
43 -((i / 4) mod 2) = 1:
```

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 74

5

6 begin m exp n end_ q

7 chM 4 6 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 75

18

19 begin m exp n end_ q

20 chM 4 6 33 31

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 76

31

32 begin m exp n end_ q

33 chM 4 3 33 31

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 77
5
6 begin m exp n end_ q
7 chM 16 3 33 31
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
11  opUAL_1 Sig_C Sig_Z Sig_Odd
12 Step
13 -((i / 4) mod 2) = 1:
14 Luciole 1.75 - top16
15
16 Files Options Clocks Tools Inputs 78
17 begin m exp n end_ q
18 chM 16 3 33 31
19 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
20 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
21 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
22  opUAL_1 Sig_C Sig_Z Sig_Odd
23 Step
24 -((i / 4) mod 2) = 1:
25 Luciole 1.75 - top16
26
27 Files Options Clocks Tools Inputs 79
28 begin m exp n end_ q
29 chM 16 3 33 31
30 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
31 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
32 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
33  opUAL_1 Sig_C Sig_Z Sig_Odd
34 Step
35 -((i / 4) mod 2) = 1:
```

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 80

5

6 begin m exp n end_ q

7 chM 16 3 33 31

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 13 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 81

18

19 begin m exp n end_ q

20 chM 16 3 33 496

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 13 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 82

31

32 begin m exp n end_ q

33 chM 16 3 33 496

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 13 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 83

5

6 begin m exp n end_ q

7 chM 16 3 33 463 etat_courant_4

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

9 chN mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

10 input 13 opUAL_1 Sig_C SelOp2 Sig_Z Sig_Odd

11 Step

12 - ((i / 4) mod 2) = 1:

13 Luciole 1.75 - top16

14

15 Files Options Clocks Tools Inputs 84

16

17 begin m exp n end_ q

18 chM 16 3 33 463 etat_courant_4

19 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

20 chN mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

21 input 13 opUAL_1 Sig_C SelOp2 Sig_Z Sig_Odd

22 Step

23 - ((i / 4) mod 2) = 1:

24 Luciole 1.75 - top16

25

26 Files Options Clocks Tools Inputs 85

27

28 begin m exp n end_ q

29 chM 16 3 33 430 etat_courant_4

30 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

31 chN mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

32 input 13 opUAL_1 Sig_C SelOp2 Sig_Z Sig_Odd

33 Step

34 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 86

5

6 begin m exp n end_ q

7 chM 16 3 33 430

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 87

18

19 begin m exp n end_ q

20 chM 16 3 33 397

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 89

31

32 begin m exp n end_ q

33 chM 16 3 33 364

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 90

5

6 begin m exp n end_ q

7 chM 16 3 33 364

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 91

18

19 begin m exp n end_ q

20 chM 16 3 33 331

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 109

31

32 begin m exp n end_ q

33 chM 16 3 33 34

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 110

5

6 begin m exp n end_ q

7 chM 16 3 33 34

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 111

18

19 begin m exp n end_ q

20 chM 16 3 33 1

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 112

31

32 begin m exp n end_ q

33 chM 16 3 33 1

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

40 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 113
5
6 begin m exp n end_ q
7  chM 16 1 33 1
8  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
11  13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
12  13 opUAL_1 Sig_C Sig_Z Sig_Odd
13 Step
14
15 -((i/4) mod 2) = 1:
16 Luciole 1.75 - top16
17
18 Files Options Clocks Tools Inputs 114
19
20 begin m exp n end_ q
21  chM 256 1 33 1
22  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
23  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
24 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
25  13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
26  13 opUAL_1 Sig_C Sig_Z Sig_Odd
27 Step
28
29 -((i/4) mod 2) = 1:
30 Luciole 1.75 - top16
31
32 Files Options Clocks Tools Inputs 127
33
34 begin m exp n end_ q
35  chM 58 1 33 1
36  chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
37  chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
38 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
39  13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
40  13 opUAL_1 Sig_C Sig_Z Sig_Odd
41 Step
42
43 -((i/4) mod 2) = 1:
```

```
2 Luciole 1.75 - top16
3
4 Files Options Clocks Tools Inputs 128
5
6 begin m exp n end_ q
7 chM 25 1 33 1
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
11 opUAL_1 Sig_C Sig_Z Sig_Odd
12 Step
13 -((i/.4) mod_2)=1:
14 Luciole 1.75 - top16
15
16 Files Options Clocks Tools Inputs 129
17 begin m exp n end_ q
18 chM 25 1 33 1
19 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
20 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
21 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
22 opUAL_1 Sig_C Sig_Z Sig_Odd
23 Step
24 -((i/.4) mod_2)=1:
25 Luciole 1.75 - top16
26
27 Files Options Clocks Tools Inputs 132
28 begin m exp n end_ q
29 chM 25 1 33 25
30 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
31 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
32 input 13 mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0
33 opUAL_1 Sig_C Sig_Z Sig_Odd
34 Step
35 -((i/.4) mod_2)=1:
```

2 Luciole 1.75 - top16

3

4 Files Options Clocks Tools Inputs 133

5

6 begin m exp n end_ q

7 chM 25 1 33 25

8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

10 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

11 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

12 13 opUAL_1 Sig_C Sig_Z Sig_Odd

13 Step

14 - ((i / 4) mod 2) = 1:

15 Luciole 1.75 - top16

16

17 Files Options Clocks Tools Inputs 134

18

19 begin m exp n end_ q

20 chM 25 0 33 25

21 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

22 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

23 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

24 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

25 13 opUAL_1 Sig_C Sig_Z Sig_Odd

26 Step

27 - ((i / 4) mod 2) = 1:

28 Luciole 1.75 - top16

29

30 Files Options Clocks Tools Inputs 171

31

32 begin m exp n end_ q

33 chM 31 0 33 25

34 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4

35 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9

36 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM

37 input mise_1 selOp1_0 selOp1_1 selOp2 opUAL_0

38 13 opUAL_1 Sig_C Sig_Z Sig_Odd

39 Step

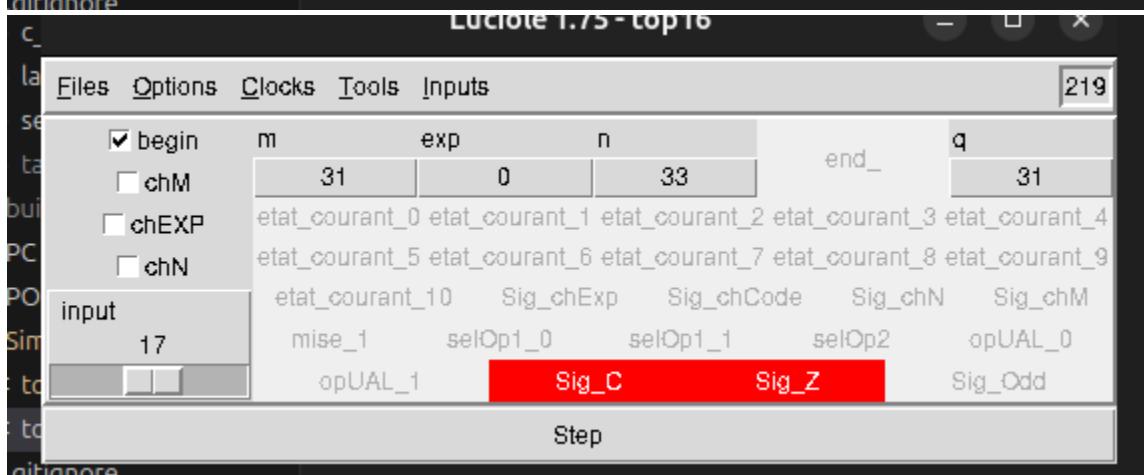
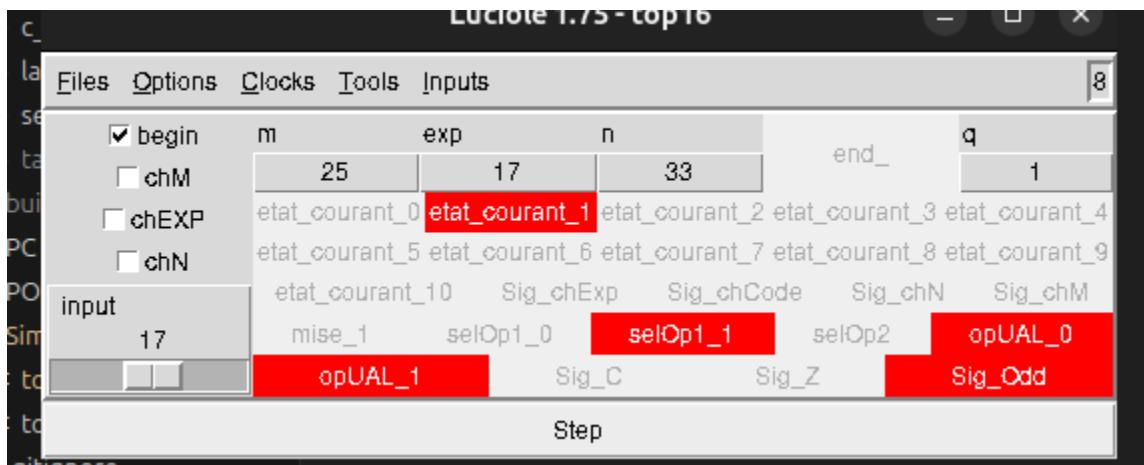
40 - ((i / 4) mod 2) = 1:

```
2 Luciole 1.75 - top16
3
4 Efiles Options Clocks Tools Inputs 172
5
6 begin m exp n end_ q
7 chM 31 0 33 25
8 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
9 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
10 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
11
12
13
14
15 Step
16
17 -((i/.4) mod 2) = 1:
18 Luciole 1.75 - top16
19
20 Efiles Options Clocks Tools Inputs 173
21
22 begin m exp n end_ q
23 chM 31 0 33 25
24 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
25 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
26 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
27
28
29
30
31 Step
32
33 -((i/.4) mod 2) = 1:
34 Luciole 1.75 - top16
35
36 Efiles Options Clocks Tools Inputs 174
37
38 begin m exp n end_ q
39 chM 31 0 33 25
40 chEXP etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4
41 chN etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9
42 input 13 etat_courant_10 Sig_chExp Sig_chCode Sig_chN Sig_chM
43
44
45
46
47 Step
48
49 -((i/.4) mod 2) = 1:
```

Pour les tests suivants, seuls les résultats et les états initiaux vous seront présentés.

```
[~/Desktop/repertoire/ArchMat/Real_mach_alg]
base ➤ arhis ➤ arhan-master ➤ ./chiffrement 31 13 33
Resultat chiffrement 25

[~/Desktop/repertoire/ArchMat/Real_mach_alg]
base ➤ arhis ➤ arhan-master ➤ ./chiffrement 25 17 33
Resultat chiffrement 31
```



```
[~/Desktop/repertoire/ArchMat/Real_mach_alg]
● base ➤ arhis ➤ arhan-master ➤ ./chiffrement 35 13 77
Resultat chiffrement 63

[~/Desktop/repertoire/ArchMat/Real_mach_alg]
● base ➤ arhis ➤ arhan-master ➤ ./chiffrement 63 37 77
Resultat chiffrement 35
```

Inputs

<input checked="" type="checkbox"/> begin	m	exp	n	end_	q
<input type="checkbox"/> chM	35	13	77		1
<input type="checkbox"/> chEXP	etat_courant_0	etat_courant_1	etat_courant_2 etat_courant_3 etat_courant_4		
<input type="checkbox"/> chN	etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9	etat_courant_10	Sig_chExp Sig_chCode Sig_chN Sig_chM		
input	77	mise_1	selOp1_0 selOp1_1	selOp2	opUAL_0
		opUAL_1	Sig_C	Sig_Z	Sig_Odd

Step

```
[~/Desktop/repertoire/ArchMat/Real_mach_alg]
● base ➤ arhis ➤ arhan-master ➤ ./chiffrement 42 0 77
Resultat chiffrement 63
```

Inputs

<input checked="" type="checkbox"/> begin	m	exp	n	end_	q
<input type="checkbox"/> chM	42	0	77		63
<input type="checkbox"/> chEXP	etat_courant_0 etat_courant_1 etat_courant_2 etat_courant_3 etat_courant_4	etat_courant_10	Sig_chExp Sig_chCode Sig_chN Sig_chM		
<input type="checkbox"/> chN	etat_courant_5 etat_courant_6 etat_courant_7 etat_courant_8 etat_courant_9	mise_1	selOp1_0 selOp1_1	selOp2	opUAL_0
input	77	opUAL_1	Sig_C	Sig_Z	Sig_Odd

Step

