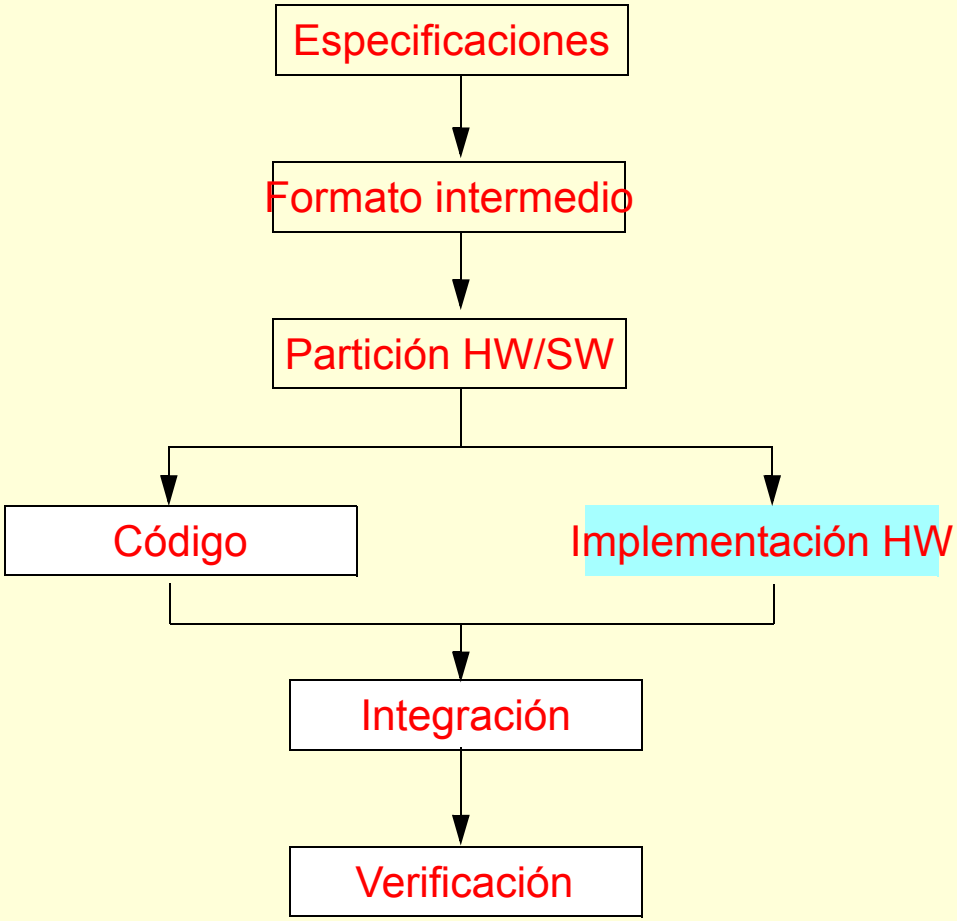
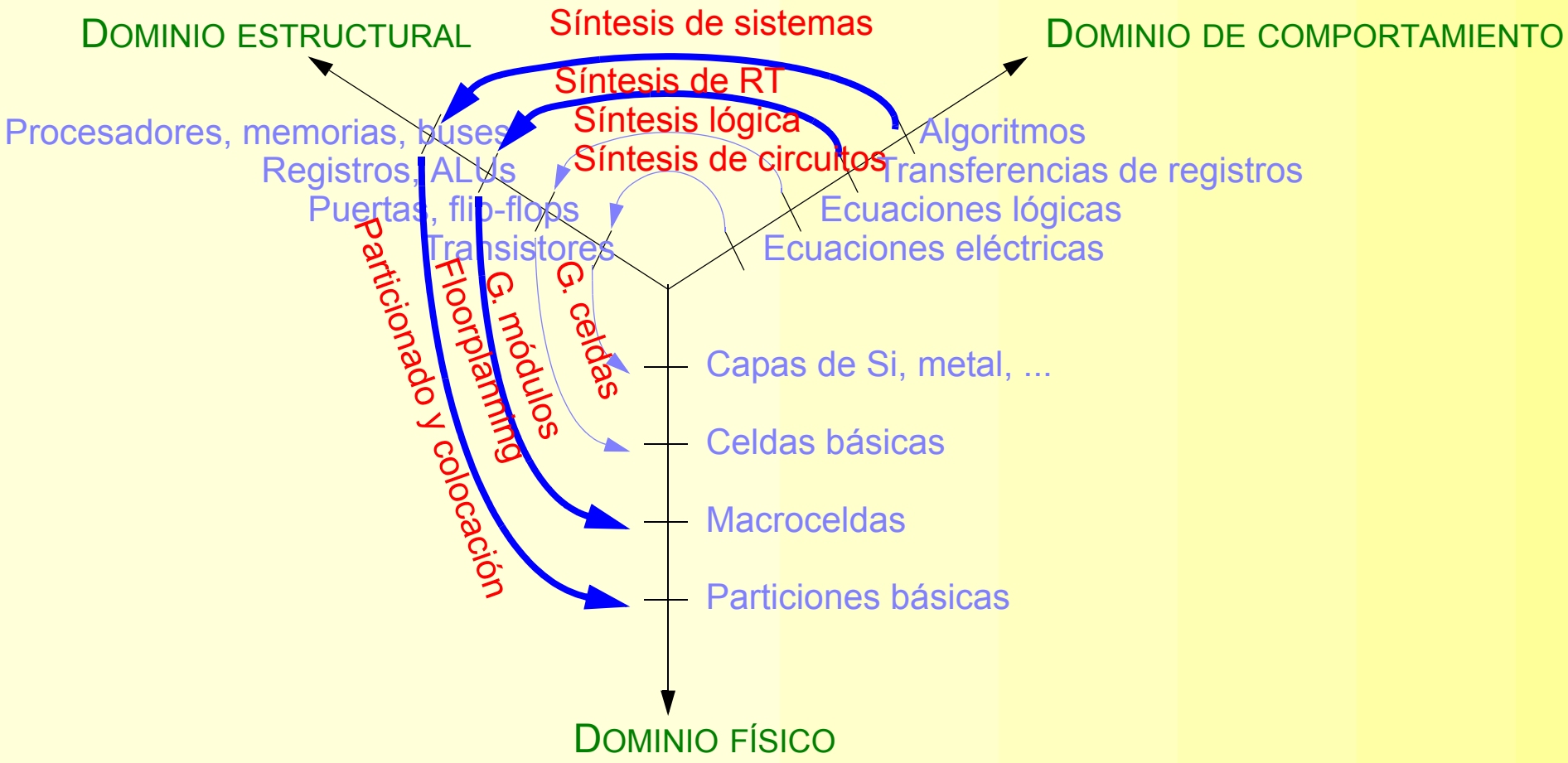
TEMA IV: SÍNTESIS HARDWARE



LA SÍNTESIS HARDWARE ES LA TAREA DE PASAR DE UN DOMINIO DE ABSTRACCIÓN A OTRO
COMPORTAMIENTO -> ESTRUCTURA

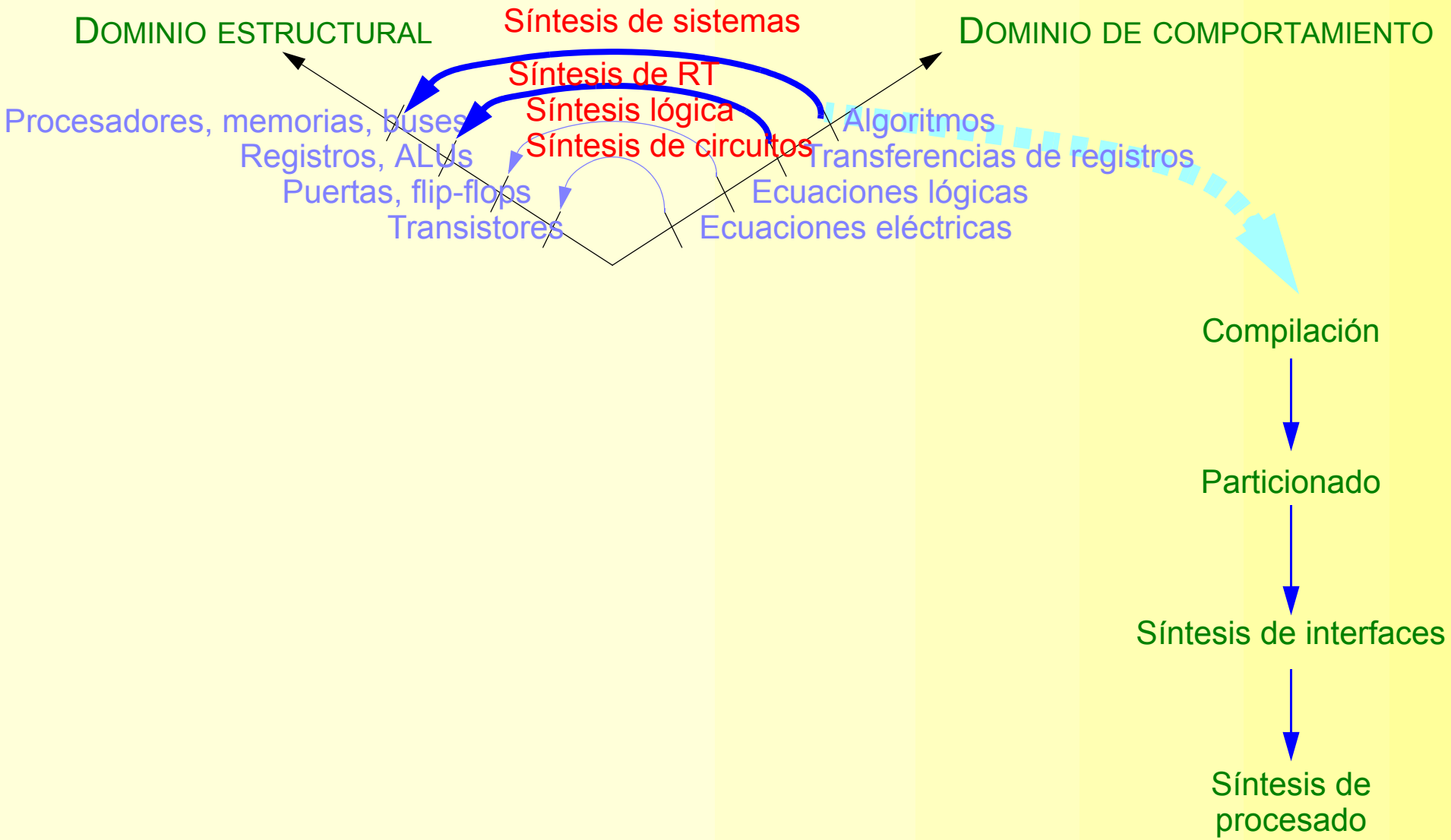
SÍNTESIS HARDWARE



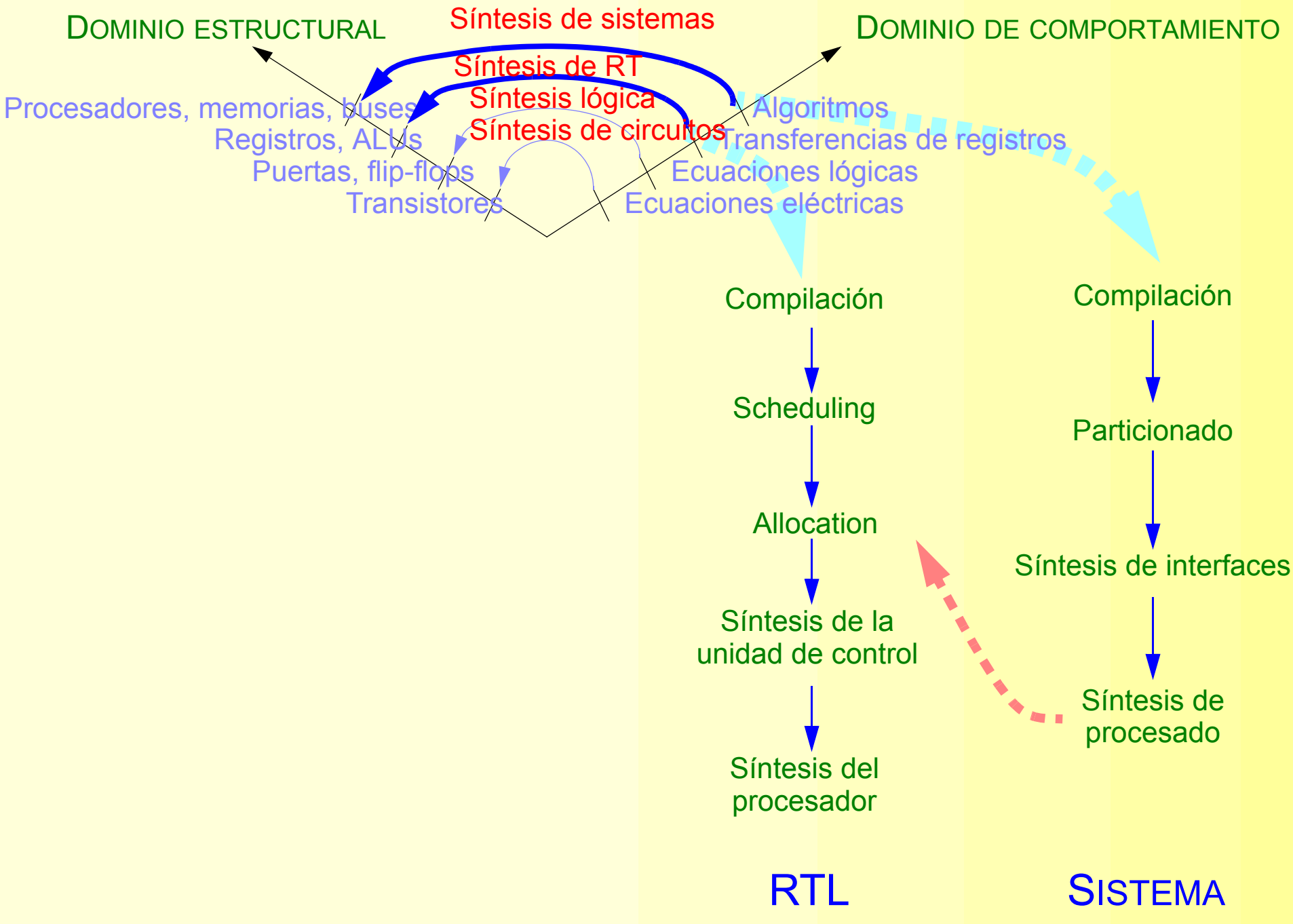
SÍNTESIS DE ALTO NIVEL
(Reducción del tiempo de diseño)

- Complejidad del diseño VLSI
- Madurez de la tecnología VLSI
- Mejor exploración del espacio de diseño

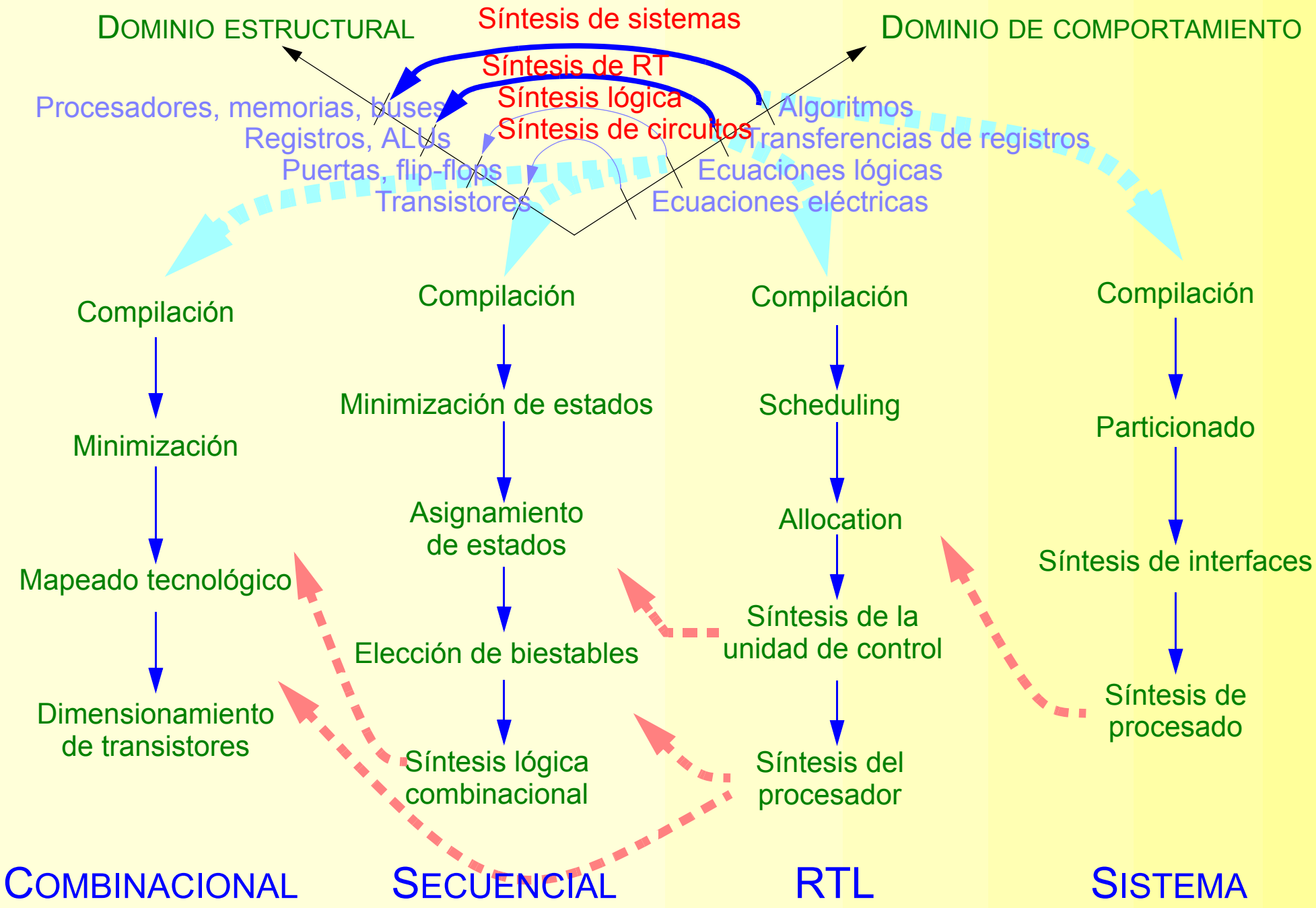
Procesos de diseño



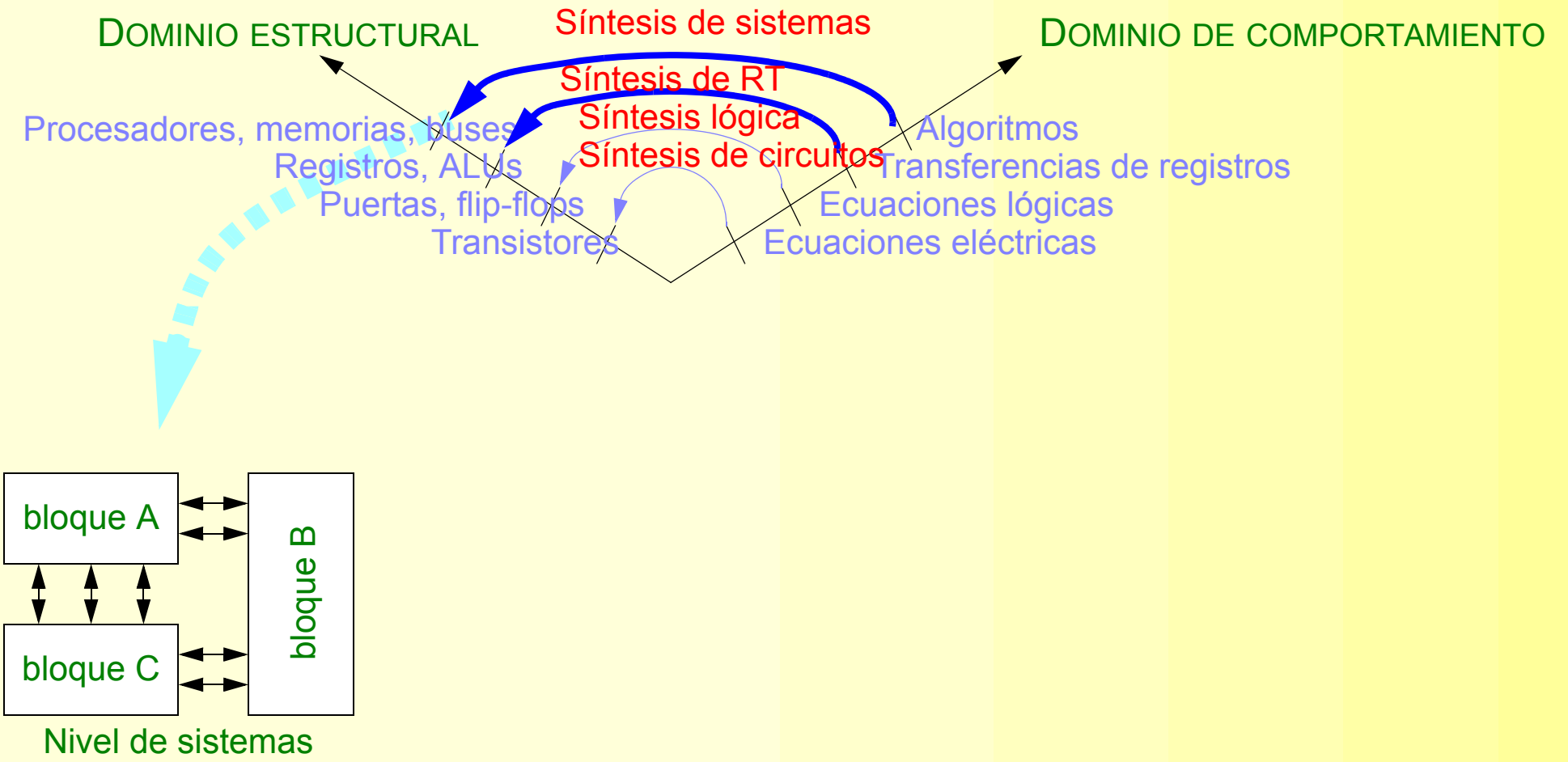
Procesos de diseño



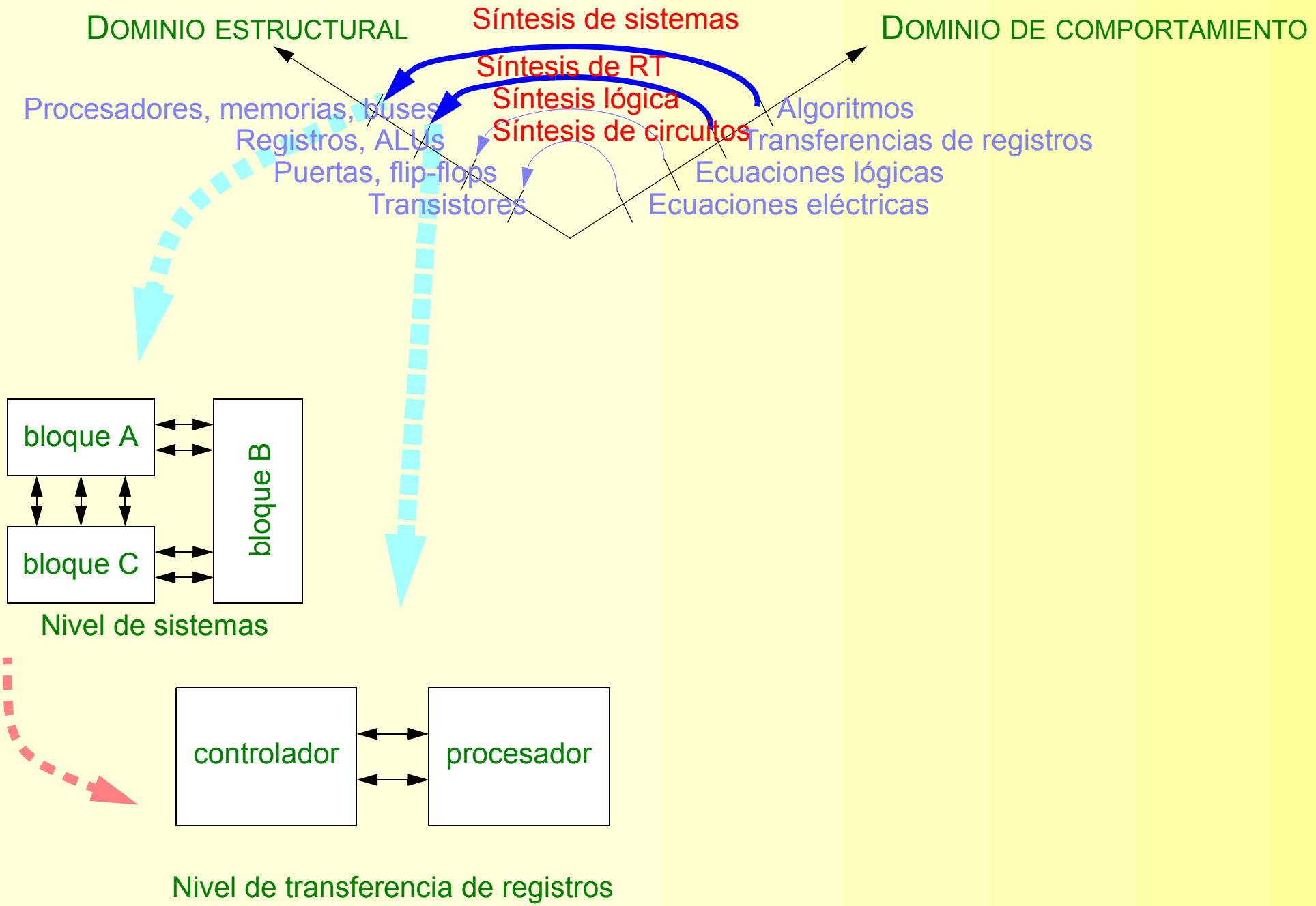
Procesos de diseño



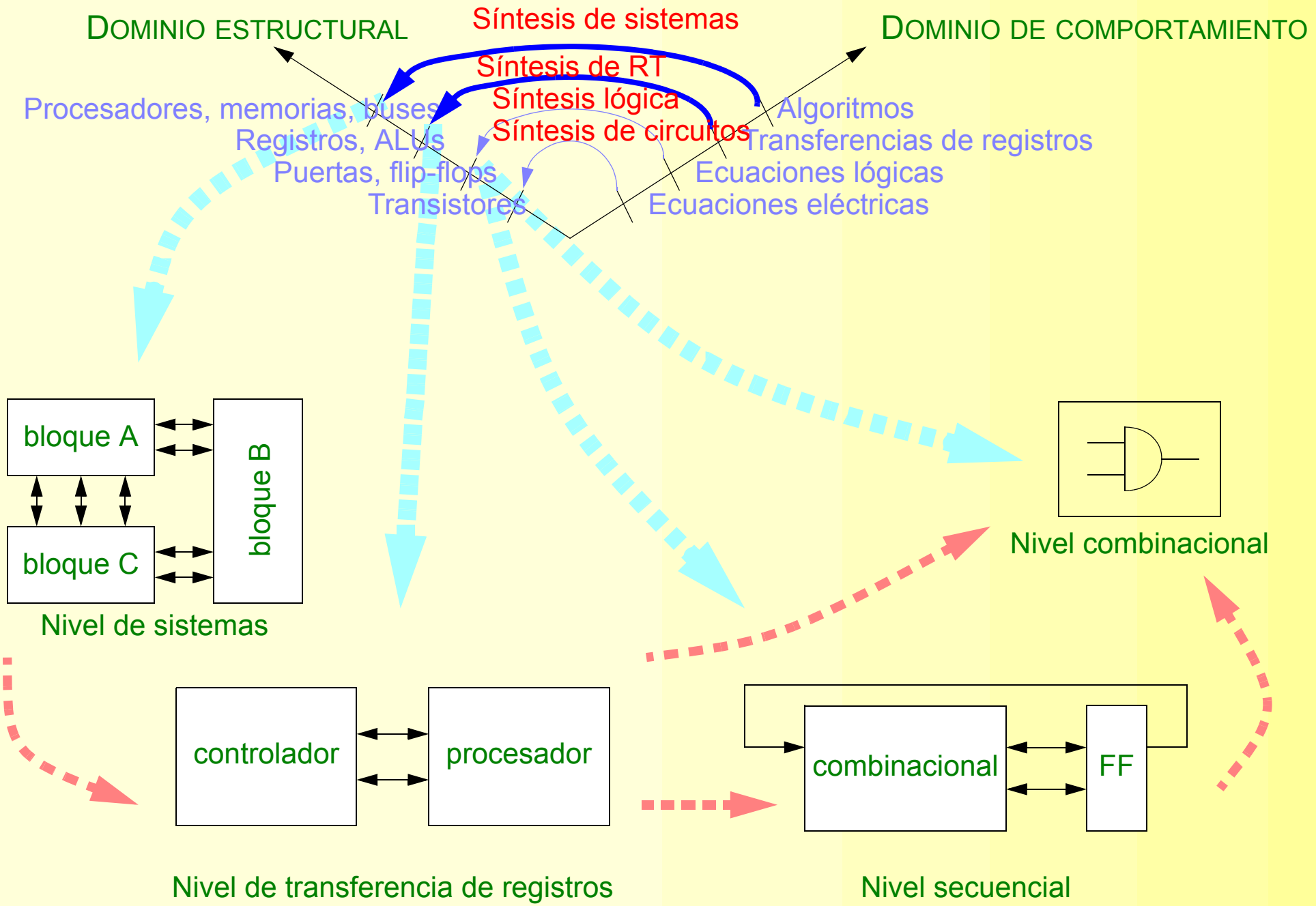
Procesos de diseño



Procesos de diseño



Procesos de diseño



COMPILACIÓN

PASA DE UNA DESCRIPCIÓN INFORMAL A OTRA INTERPRETABLE POR LOS ALGORITMOS USADOS EN EL PROCESO EN CUESTIÓN

LENGUAJES (VHDL)

- Tipos de datos
- Operadores
- Sentencias concurrentes
- Sentencias secuenciales
- Definiciones de usuario

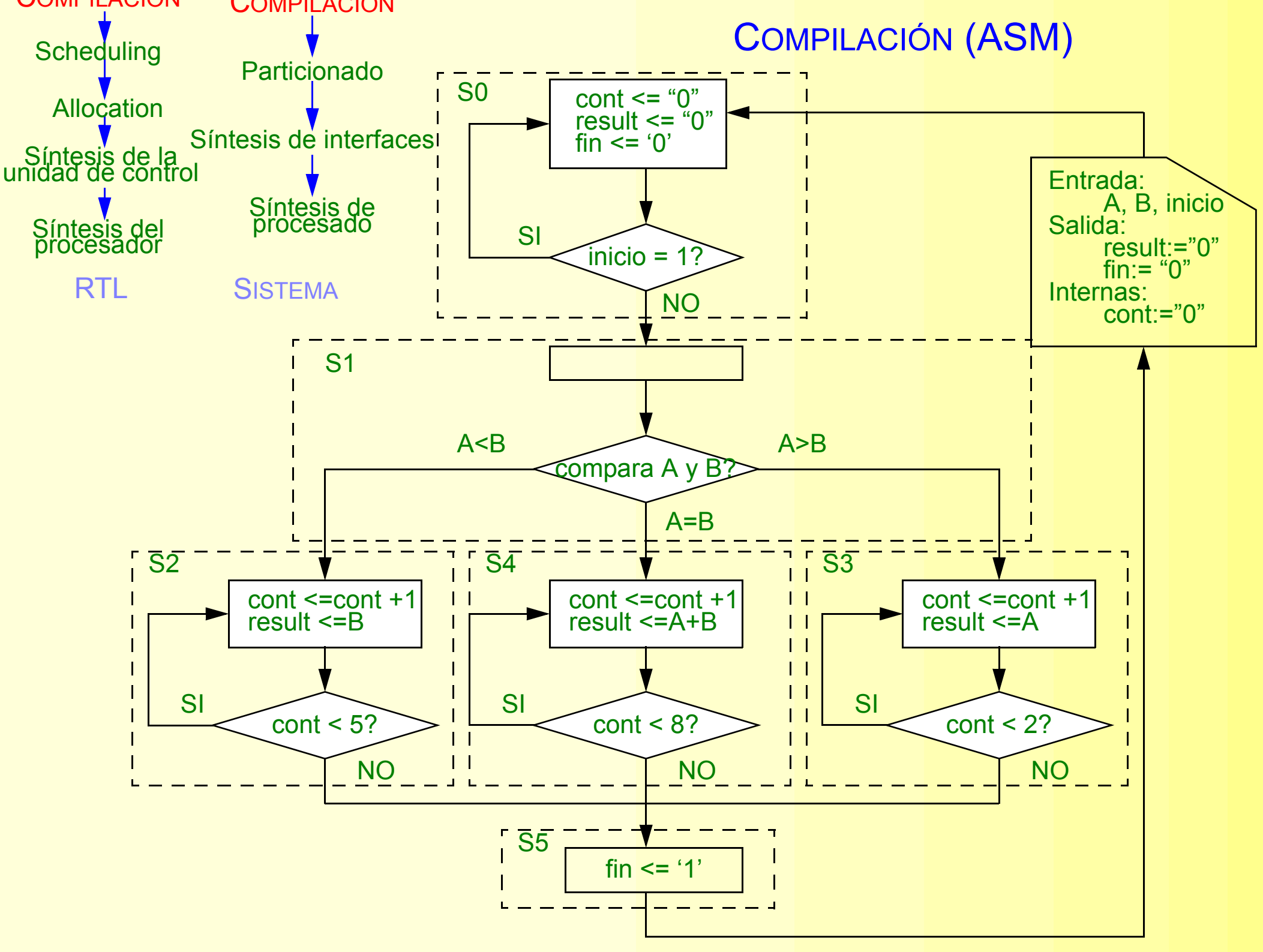
GRAFOS (ASM)

- Cajas de estado
- Cajas condicionales
- Cajas de selección
- Tarjeta identificativa

COMPILACIÓN (VHDL)

```
entity prueba is
port(       A, B : in bit_vector(7 downto 0);
            inicio, clk : in std_logic;
            result : out bit_vector(8 downto 0)
            fin : out bit);
end prueba;

architecture primera of prueba is
        type estado is (S0, S1, S2, S3, S4, S5);
        signal cont : bit_vector (3 downto 0);
        signal estado_presente, proximo_estado : estado;
begin
        P1:process(estado_presente)
        begin
          case estado_presente is
            when S0 =>
              cont <= “0000”;
              result <= “000000000”;
              fin <= ‘0’;
              if (inicio = ‘1’) then proximo_estado <= S1;
              else proximo_estado <= S0;
              end if;
            when S1 =>
              if (A < B) then proximo_estado <= S2;
              elsif (A > B) then proximo_estado <= S3;
              else proximo_estado <= S4;
              end if;
            when S2 =>
              result <= B;
              cont <= cont+1;
              if (cont < “0101”) then proximo_estado <= S2;
              else proximo_estado <= S5;
              end if;
            when S3 =>
              result <= A;
              cont <= cont+1;
              if (cont < “0010”) then proximo_estado <= S3;
              else proximo_estado <= S5;
              end if;
            when S4 =>
              result <= A+B;
              cont <= cont+1;
              if (cont < “1000”) then proximo_estado <= S4;
              else proximo_estado <= S5;
              end if;
            when S5 =>
              fin <= ‘1’;
              proximo_estado <= S0;
          end case;
        end process;
        B1:block (clk = ‘1’ and not clk’stable)
        begin
          estado_presente <= guarded proximo_estado;
        end block;
end;
```


COMPILACIÓN
Scheduling
Allocation
Síntesis de la unidad de control
Síntesis del procesador
RTL
COMPILACIÓN
Particionado
Síntesis de interfaces
Síntesis de procesado
SISTEMA
COMPILACIÓN (ASM)
S0
cont <= "0"
result <= "0"
fin <= '0'
SI
inicio = 1?
NO
Entrada:
A, B, inicio
Salida:
result:="0"
fin:= "0"
Internas:
cont:="0"
S1
A<B
compara A y B?
A>B
A=B
S2
cont <=cont +1
result <=B
SI
cont < 5?
NO
S4
cont <=cont +1
result <=A+B
SI
cont < 8?
NO
S3
cont <=cont +1
result <=A
SI
cont < 2?
NO
S5
fin <= '1'

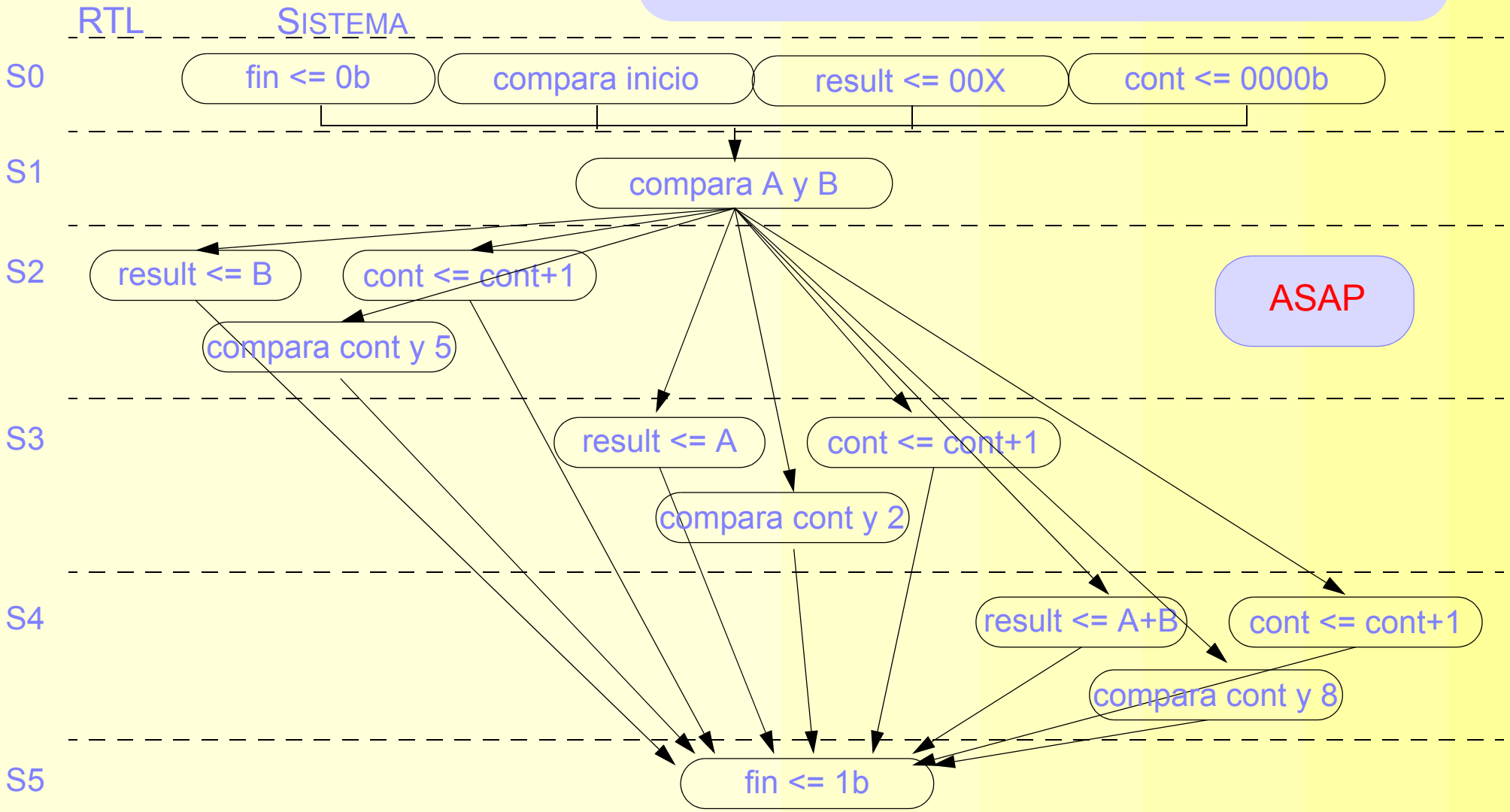
PARTICIONADO

EL PARTICIONADO ES LA TAREA QUE DETERMINARÁ LA FUNCIONALIDAD IMPLEMENTADA EN LOS DIFERENTES BLOQUES DE PROCESADO

- RECOCIDO SIMULADO
- MIGRACIONES DE GRUPO
- CLUSTERING
- PROGRAMACIÓN LINEAL
- ALGORITMOS GENÉTICOS

SCHEDULING

SCHEDULING ES LA TAREA QUE ASIGNA LAS OPERACIONES A INSTANTES DE TIEMPO EN FUNCIÓN DE RESTRICCIONES Y MINIMIZANDO FUNCIONES DE COSTE



Compilación

SCHEDULING

Allocation

Síntesis de la unidad de control

Síntesis del procesador

Compilación

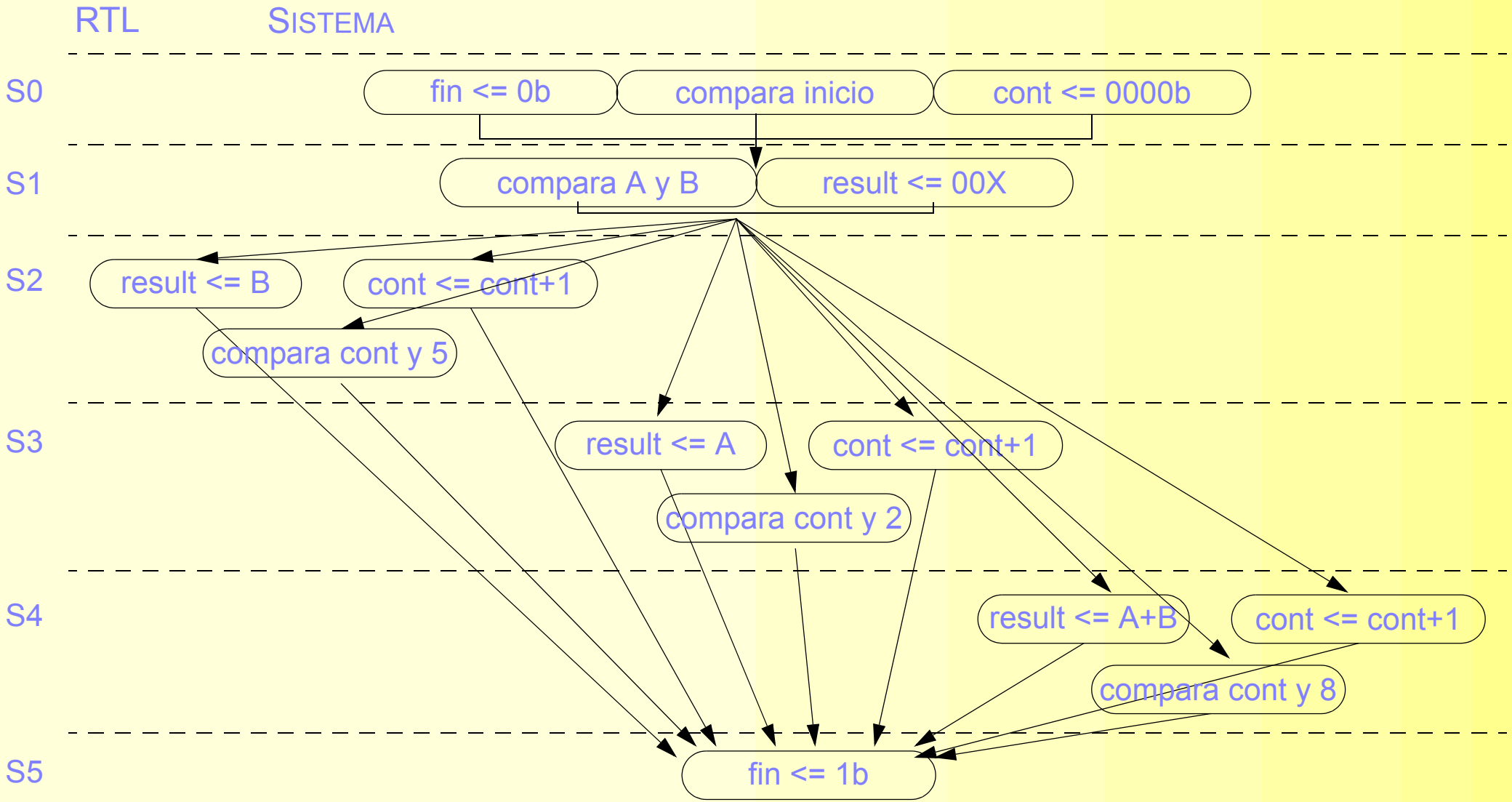
Particionado

Síntesis de interfaces

Síntesis de procesado

SCHEDULING (ALAP)

AS LAST AS POSSIBLE



Compilación

SCHEDULING

Allocation

Síntesis de la unidad de control

Síntesis del procesador

Compilación

Particionado

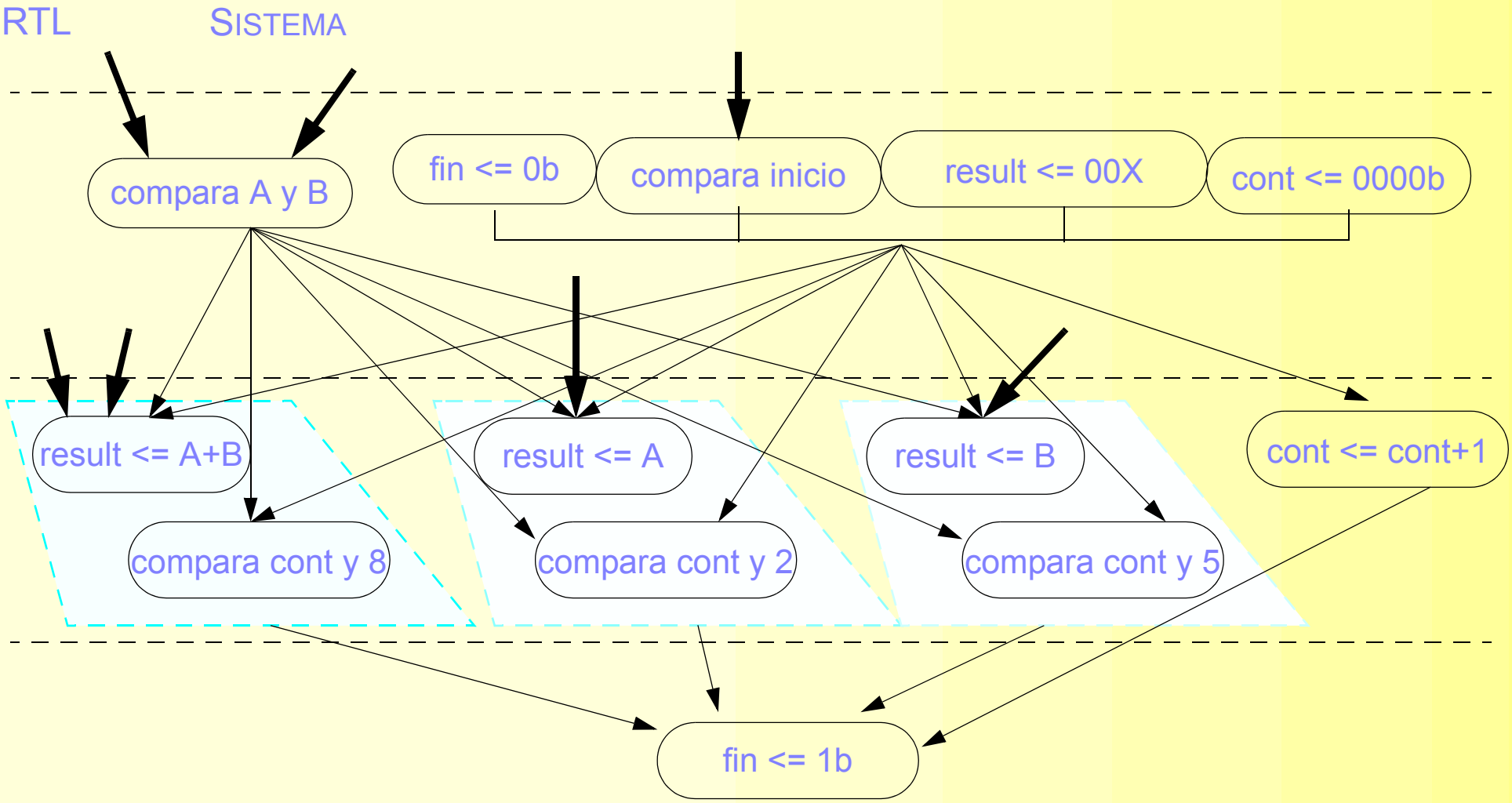
Síntesis de interfaces

Síntesis de procesado

SCHEDULING

RESTRICCIONES TEMPORALES

RESTRICCIONES DE RECURSOS



Compilación

SCHEDULING

Allocation

Síntesis de la unidad de control

Síntesis del procesador

RTL

Compilación

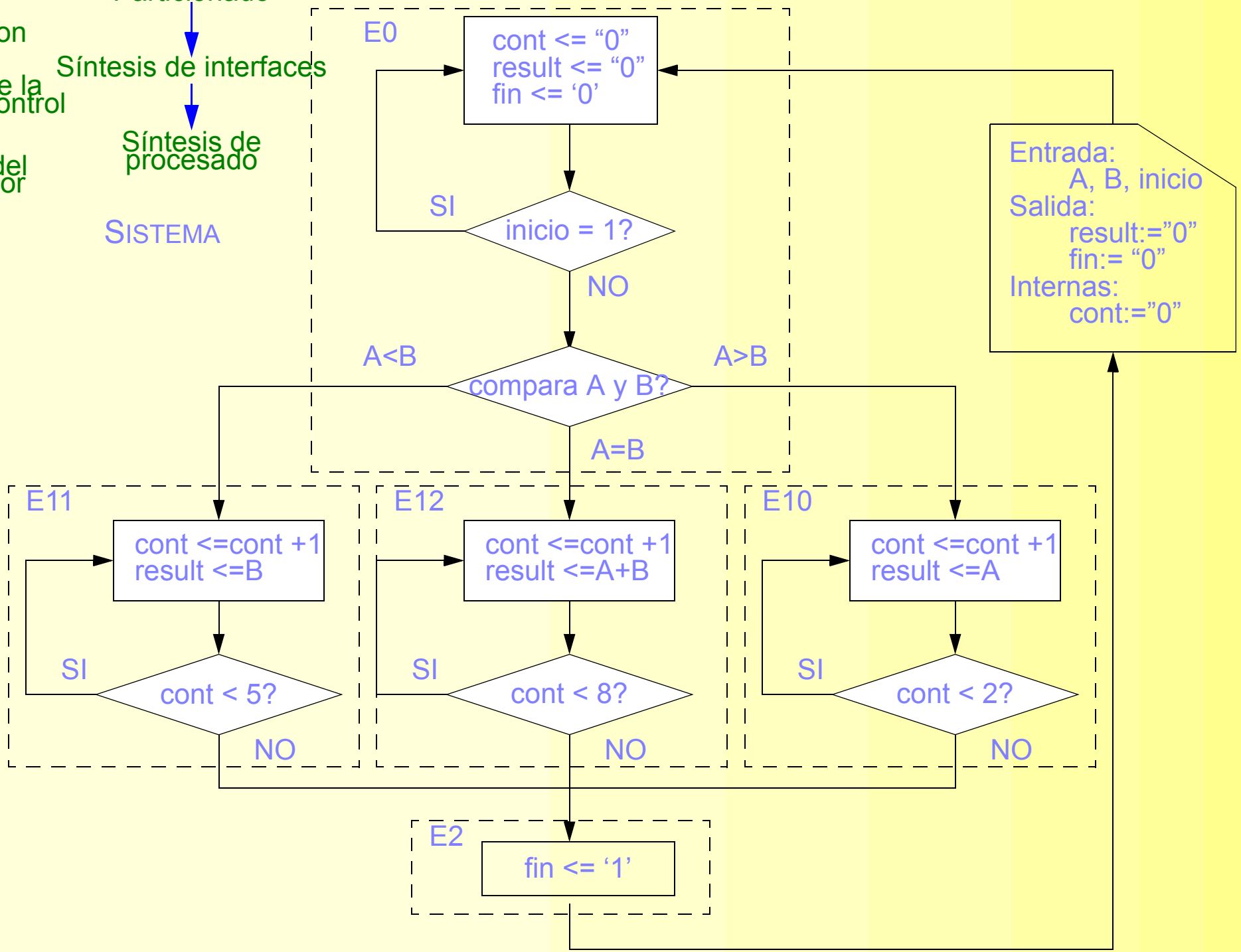
Particionado

Síntesis de interfaces

Síntesis de procesado

SISTEMA

SCHEDULING



ALLOCATION

ALLOCATION ES LA TAREA QUE ASIGNA LAS OPERACIONES A MÓDULOS HARDWARE EN FUNCIÓN DE SCHEDULING, RESTRICCIONES Y MINIMIZANDO FUNCIONES DE COSTE

CARACTERÍSTICAS DE LOS MÓDULOS

INTERCONEXIONES A TRAVÉS DE BUSES

INTERCONEXIONES A TRAVÉS DE MULTIPLEXORES

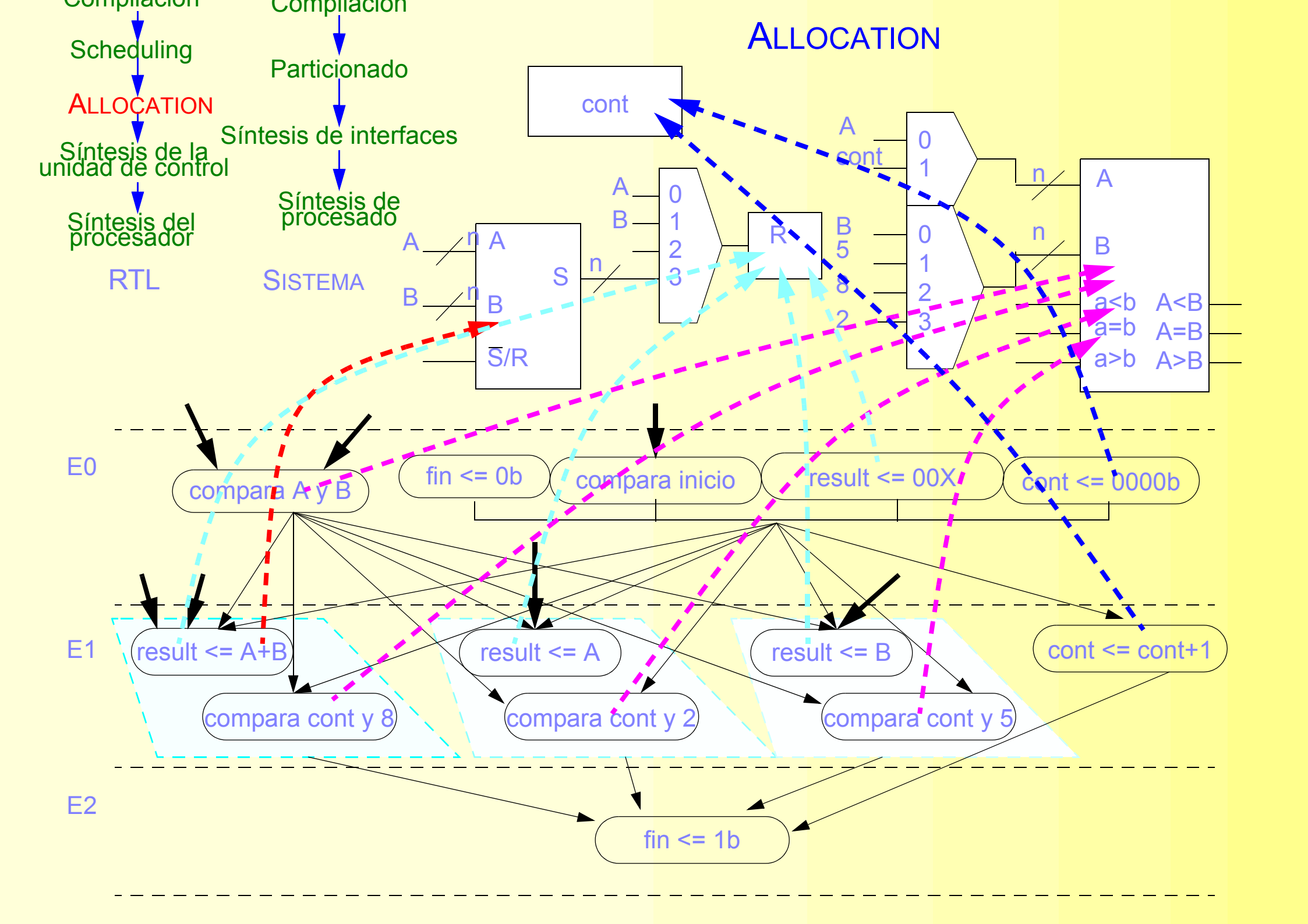
INTERCONEXIONES PUNTO A PUNTO

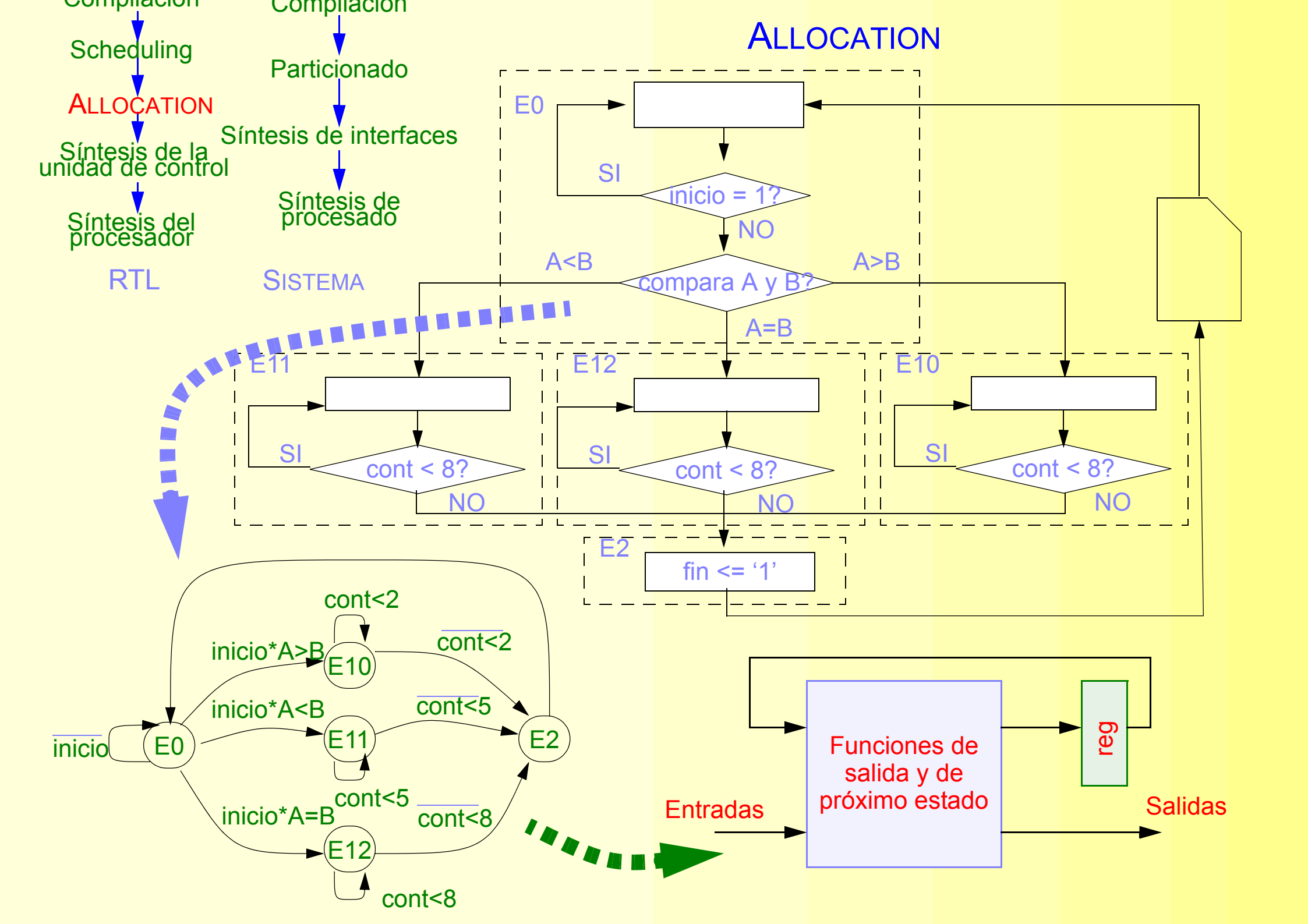
TRANSFERENCIA DE REGISTROS

UNIDADES FUNCIONALES

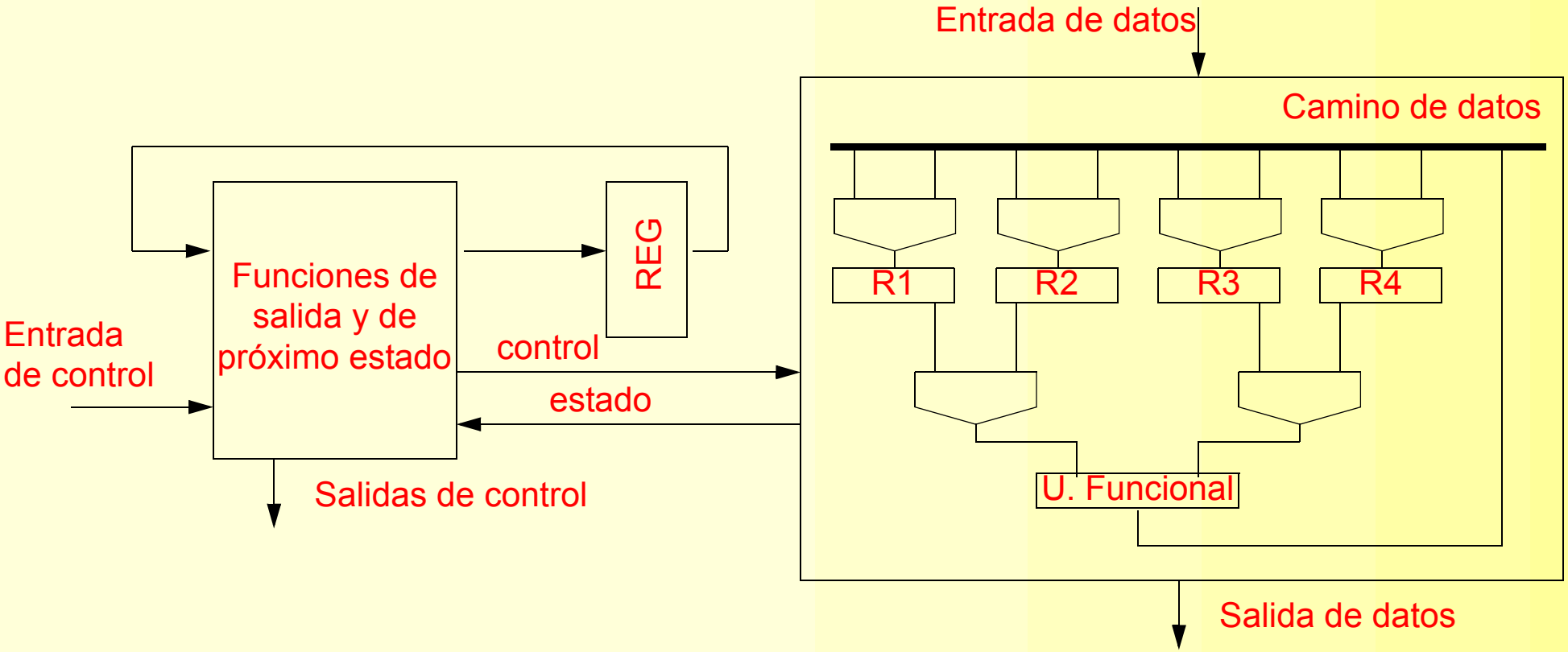
UNIDADES DE ALMACENAMIENTO

UNIDADES DE COMUNICACIONES


Compilación
Scheduling
ALLOCATION
Síntesis de la unidad de control
Síntesis del procesador
RTL
Compilación
Particionado
Síntesis de interfaces
Síntesis de procesado
SISTEMA
ALLOCATION
cont
A
cont
0
1
n
A
B
a<b A<B
a=b A=B
a>b A>B
A
B
n
S
S/R
R
B
5
8
2
0
1
2
3
E0
compara A y B
fin <= 0b
compara inicio
result <= 00X
cont <= 0000b
E1
result <= A+B
result <= A
result <= B
cont <= cont+1
compara cont y 8
compara cont y 2
compara cont y 5
E2
fin <= 1b


Compilación
Scheduling
ALLOCATION
Síntesis de la unidad de control
Síntesis del procesador
RTL
Compilación
Particionado
Síntesis de interfaces
Síntesis de procesado
SISTEMA
ALLOCATION
E0
SI
inicio = 1?
NO
A<B
compara A y B?
A>B
A=B
E11
SI
cont < 8?
NO
E12
SI
cont < 8?
NO
E10
SI
cont < 8?
NO
E2
fin <= '1'
cont<2
inicio*A>B
E10
cont<2
inicio*A<B
E11
cont<5
inicio
E0
E2
cont<5
inicio*A=B
cont<8
E12
cont<8
Entradas
Funciones de salida y de próximo estado
reg
Salidas

Estimación de prestaciones



TIEMPO DE EJECUCIÓN => VELOCIDAD DEL SISTEMA

ÁREA OCUPADA => TAMAÑO DEL SISTEMA

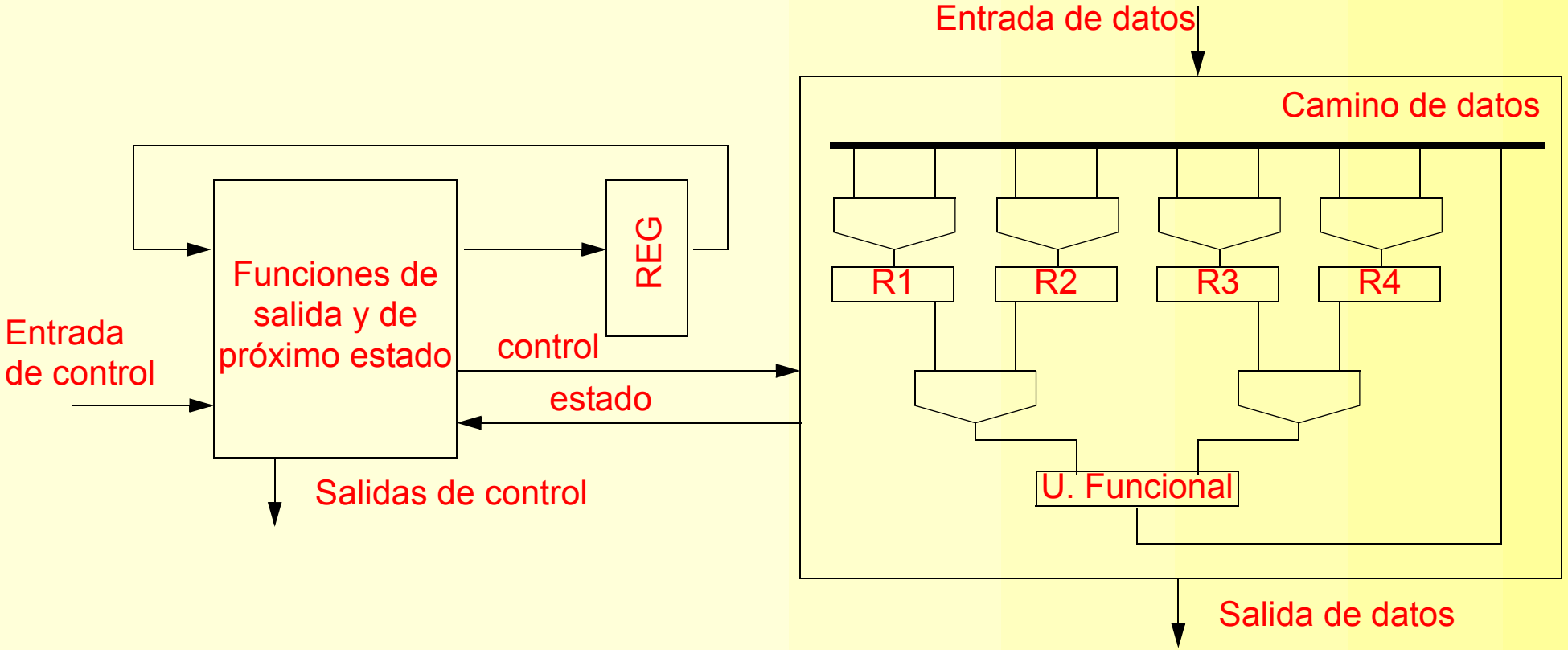
CONSUMO DE POTENCIA => COSTE ENERGÉTICO DEL SISTEMA

Estimación de prestaciones

TIEMPO DE EJECUCIÓN

$$t_p(DP) = t_{p(MUX_R)} + t_{p(Lectura)} + t_{p(MUX_F)} + t_{p(Unidad\ Funcional)} + t_{p(Setup)}$$

$$t_p(CU) = t_{p(Próximo\ estado)} + t_{p(Setup)} + t_{p(Lectura)} + t_{p(Lógica\ de\ control)}$$



Estimación de prestaciones

TIEMPO DE EJECUCIÓN

$$t_p(DP) = t_{p(MUX_R)} + t_{p(Lectura)} + t_{p(MUX_F)} + t_{p(Unidad\ Funcional)} + t_{p(Setup)}$$

$$t_p(CU) = t_{p(Próximo\ estado)} + t_{p(Setup)} + t_{p(Lectura)} + t_{p(Lógica\ de\ control)}$$

ALGORITMO DE ESTIMACIÓN

Tiempo de ejecución de micronodo = operaciones ejecutadas en un ciclo de reloj

Tiempo de ejecución de macronodo = operaciones que se pueden ejecutar en un ciclo de operación

$$TE_macronodo(j) = MAX_{op}([usos_op \cdot numbloques + pipeline - 1]retraso)$$

Estimación de prestaciones

Tiempo de ejecución

$$t_p(DP) = t_{p(MUX_R)} + t_{p(Lectura)} + t_{p(MUX_F)} + t_{p(Unidad\ Funcional)} + t_{p(Setup)}$$

$$t_p(CU) = t_{p(Próximo\ estado)} + t_{p(Setup)} + t_{p(Lectura)} + t_{p(Lógica\ de\ control)}$$

Algoritmo de estimación

Tiempo de ejecución de micronodo = operaciones ejecutadas en un ciclo de reloj

Tiempo de ejecución de macronodo = operaciones que se pueden ejecutar en un ciclo de operación

$$TE_macronodo(j) = MAX_{op}([usos_op \cdot numbloques + pipeline - 1]retraso)$$

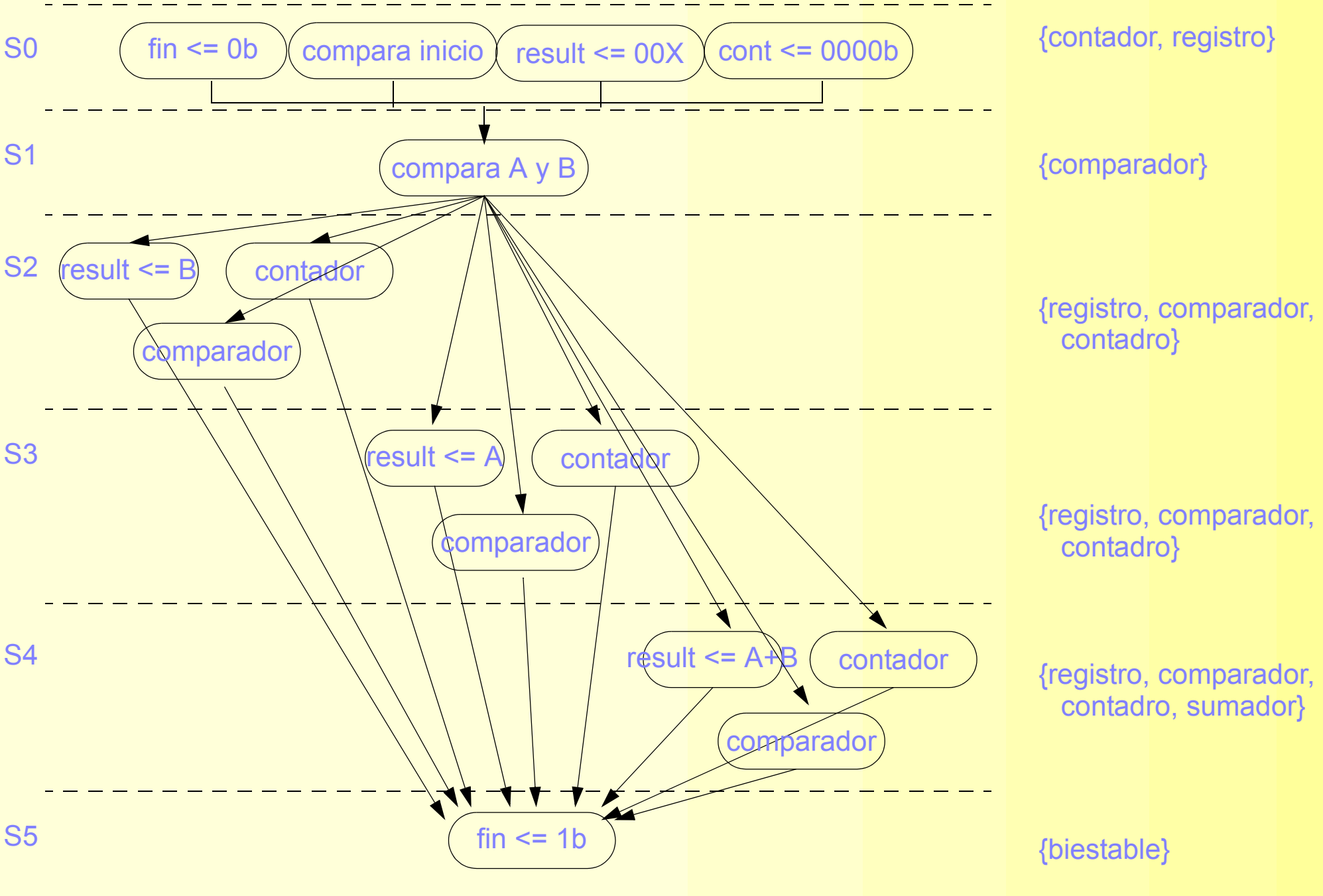
Agrupación de los cálculos anteriores con frecuencia y probabilidad

Si no hay bifurcaciones, la frecuencia y la probabilidad de cada macronod es igual a 1

Si hay bifurcaciones, a cada una se le debe asignar una frecuencia y una probabilidad

Estimación de prestaciones

Tiempo de ejecución (Ejemplo)



Estimación de prestaciones

TIEMPO DE EJECUCIÓN (EJEMPLO)

{contador, registro} ⇢ TE_S0 = MAX [(1cont/1+1-1)·1, (1reg/1+1-1)·1]

{comparador} ⇢ TE_S1 = (1comp/1+1-1)·1

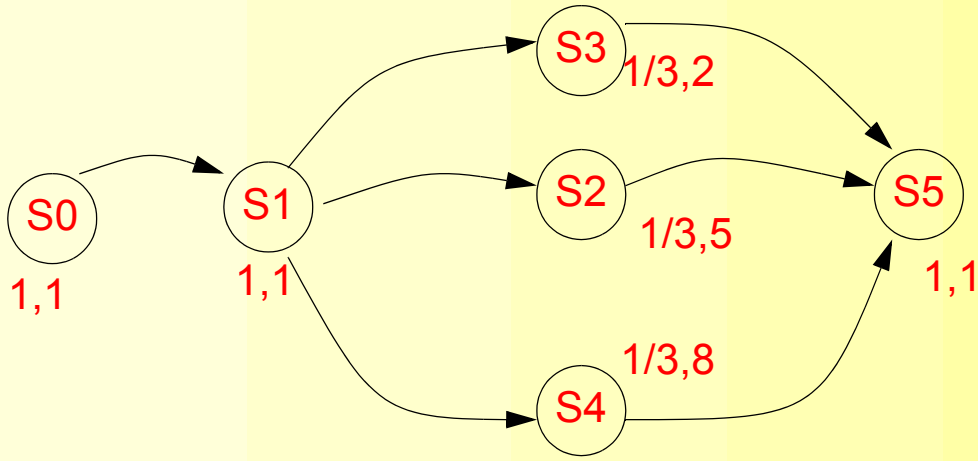
{registro, comparador, contadro} ⇢ TE_S2 = MAX [(1cont/1+1-1)·1, (1reg/1+1-1)·1, (1comp/1+1-1)·1]

{registro, comparador, contadro} ⇢ TE_S3 = MAX [(1cont/1+1-1)·1, (1reg/1+1-1)·1, (1comp/1+1-1)·1]

{registro, comparador, contadro, sumador} ⇢ TE_S4 = MAX [(1cont/1+1-1)·1, (1reg/1+1-1)·1, (1comp/1+1-1)·1, (1comp/1+1-1)·1]

{biestable} ⇢ TE_S5 = (1bies/1+1-1)·1

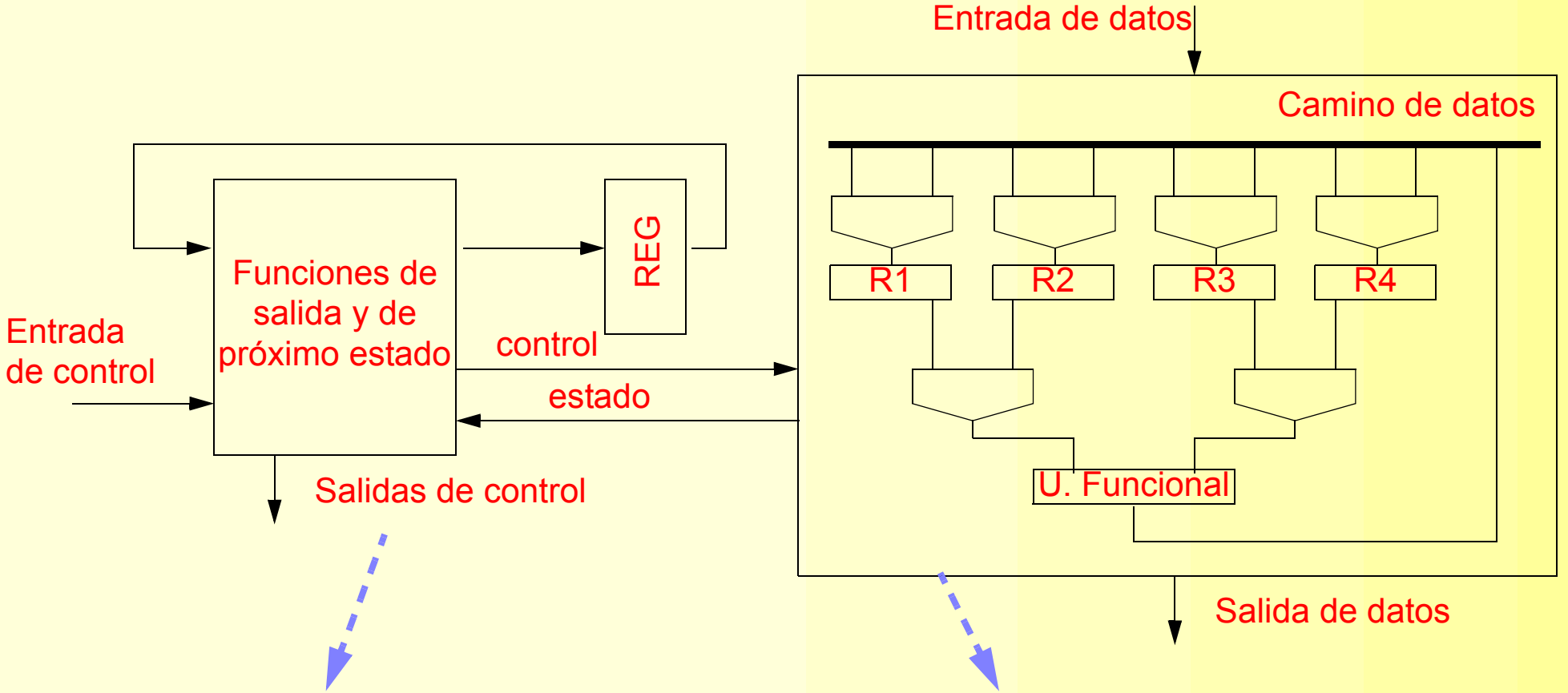
probabilidad, frecuencia



TE = 1·1·TE_S0 + 1·1·TE_S1 + 1/3·5·TE_S2 + 1/3·2·TE_S3 + 1/3·8·TE_S4 + 1·1·TE_S5

Estimación de prestaciones

ÁREA OCUPADA



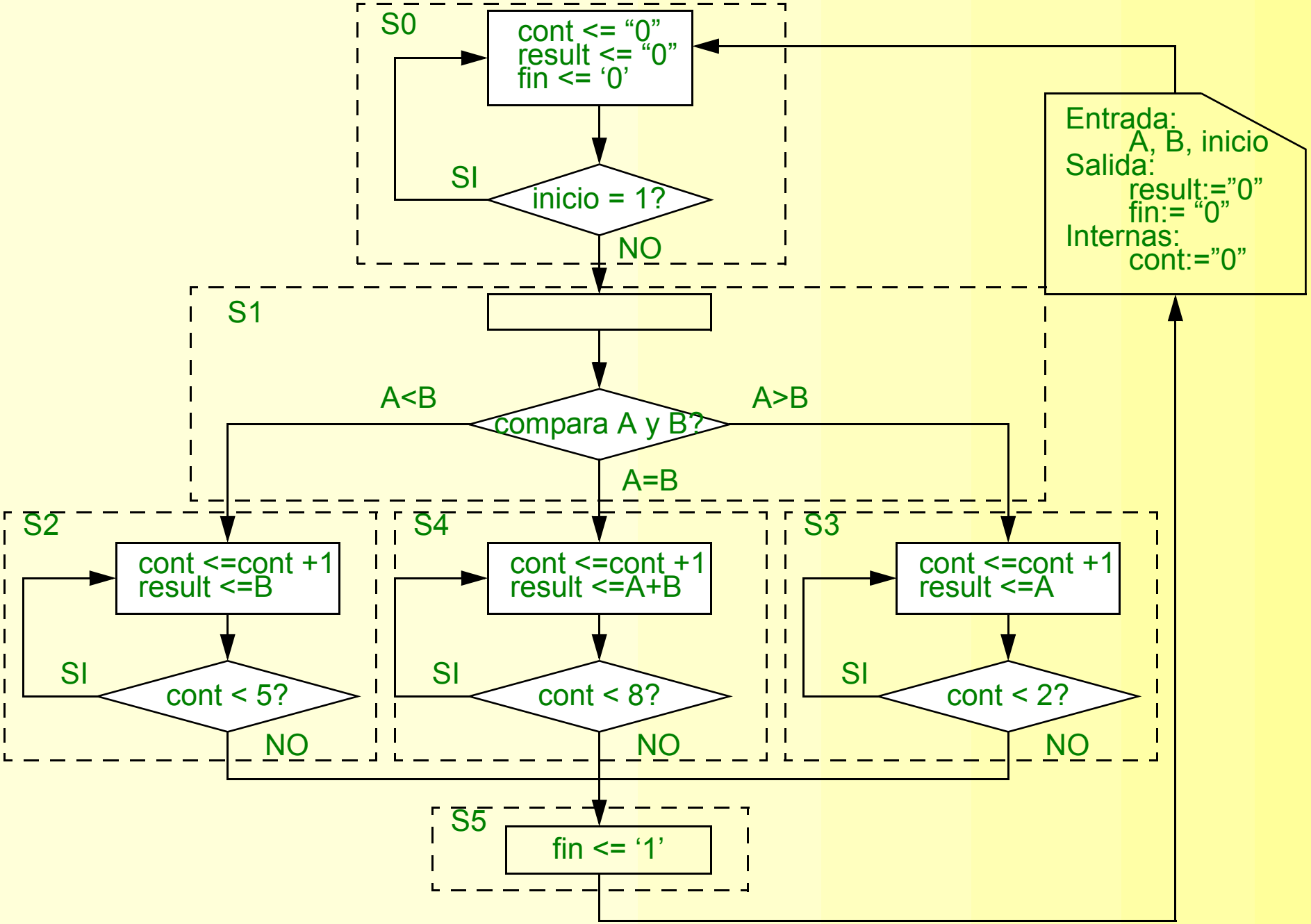
MEMORIA ROM Y BIESTABLES

UNIDADES FUNCIONALES Y MULTIPLEXORES (Nº Y TAMAÑO)

UNIDADES DE MEMORIA

Estimación de prestaciones

Área ocupada (Ejemplo)



Estimación de prestaciones

Consumo de potencia

$$P = C_L V_{DD}^2 f + (t_r + t_f) V_{DD} I_P f + V_{DD} I_{fugas} + V_{DD} I_{estática}$$

Simulación RTL => Estimación de la actividad de conmutación

↓

Modelo de energía

↓

Cuantificar el modelo de energía en función de la actividad