```
          salida<=valor2;
       .
       .
       WHEN estN=>
          salida<=valorN;
   END CASE;
   IF clk='1' AND clk'event THEN
     CASE presente IS
       WHEN est1=>
          presente<=f1(entrada);
       WHEN est2=>
          presente<=f2(entrada);
       .
       .
       WHEN estN=>
          presente<=fN(entrada);
     END CASE;
   END IF;
END PROCESS fsm;
```

Normalmente, las máquinas de estados suelen necesitar una señal de `reset` que lleve a la máquina a un estado conocido de partida. De hecho, es buena práctica, y debería ser obligatorio, poner un réset a cualquier descripción de un circuito que fuera secuencial.

Se muestra a continuación cómo sería la estructura de una máquina genérica que incorporara réset, reloj, y que fuera de tipo Mealy (la salida depende también de la entrada):

```
ENTITY maquina IS
PORT (entrada: IN tipoin; clk,reset: IN bit; salida: OUT tipout);
END maquina;

ARCHITECTURE mealy OF maquina IS
TYPE estado IS (est1,est2,...,estN);
SIGNAL presente: estado:=est1;         -- el inicial
BEGIN

--Bloque secuencial:
estados:
PROCESS(clk,reset)       -- réset asíncrono
BEGIN
   IF reset='1' THEN
     presente<=est1;     -- estado inicial
   ELSIF clk='1' AND clk'event THEN
     CASE presente IS
       WHEN est1=>
          presente<=f1(entrada);
       WHEN est2=>
          presente<=f2(entrada);
       .
       .
       WHEN estN=>
          presente<=fN(entrada);
     END CASE;
```