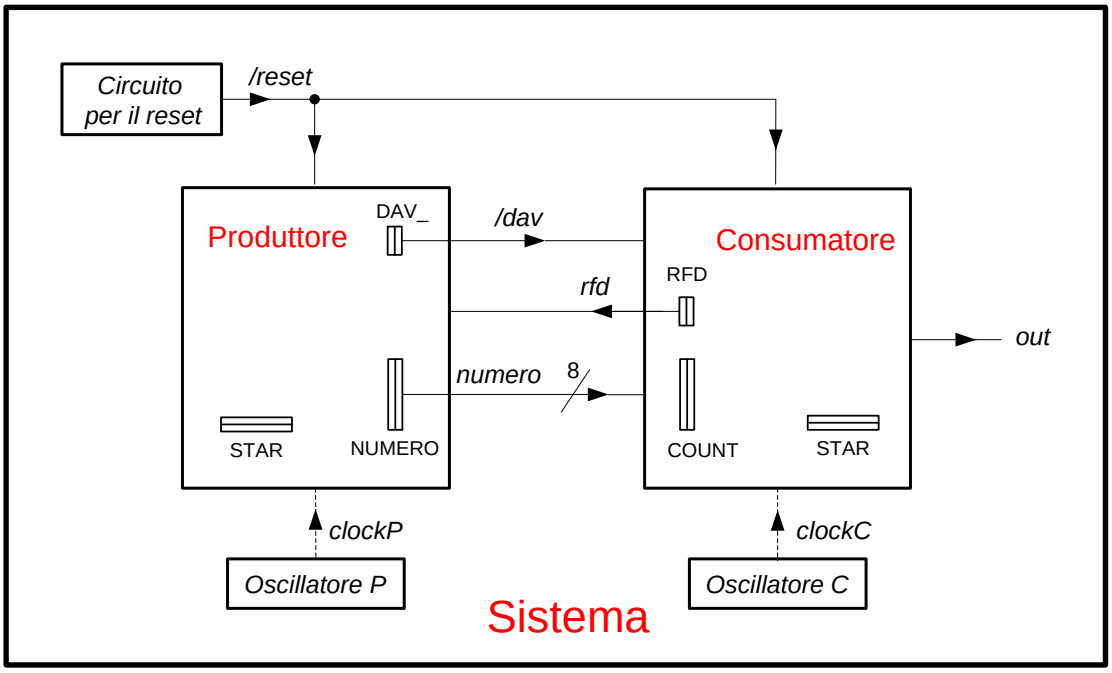
Sistema con Produttore e Consumatore
handshake *dav_* (*data valid*), *rfd* (*ready for data*)




```
module Sistema;
 // Terminazione forzata dopo 300 ns
 initial begin #300 $stop; end
 // Generatore del reset
 reg reset_; initial begin reset_=0; #1 reset_=1; end
 // Oscillatore per il clock del Produttore con periodo di 26 ns
 reg clockP ; initial clockP=0; always #13 clockP<=(!clockP);
 // Oscillatore per il clock del Consumatore con periodo di 16 ns
 reg clockC ; initial clockC=0; always #8 clockC<=(!clockC);
 // Collegamenti tra moduli
 wire[7:0] numero;
 wire dav_,rfd;
 // Variabile di uscita del Consumatore;
 wire out;
 // Moduli
 Produttore    PRO(dav_,rfd,numero,clockP,reset_);
 Consumatore   CON(dav_,rfd,numero,out,clockC,reset_);
endmodule
```

Lo schema di evoluzione del Produttore che useremo è molto semplice (ma non è il più efficiente)

1. Preparazione di un dato per il Consumatore
2. Apertura e chiusura dell'handshake
3. Ritorno al punto 1

Lo schema di evoluzione del Consumatore che useremo è molto semplice (ma non è il più efficiente)

1. Apertura dello handshake con prelievo del dato e chiusura dello handshake
2. Elaborazione del dato
3. Ritorno al punto 1

```
// Produttore di numeri naturali ad 8 bit
module Produttore(dav_,rfd,numero,clockP,reset_); // Produce numeri dispari
 input        clockP,reset_;
 input        rfd;
 output       dav_;
 output [7:0] numero;
 reg          DAV_;     assign dav_=DAV_;
 reg [7:0]    NUMERO;   assign numero=NUMERO;
 reg [1:0]    STAR;     parameter S0=0,S1=1,S2=2;
 always @(reset_==0) begin DAV_<=1; NUMERO<=0; STAR<=S0; end
 always @(posedge clockP) if (reset_==1) #3
   casex(STAR)
    S0:  begin NUMERO<=NUMERO+1; STAR<=S1; end
    S1:  begin DAV_<=0; STAR<=(rfd==1)?S1:S2; end
    S2:  begin DAV_<=1; STAR<=(rfd==0)?S2:S0; end
   endcase
endmodule

// Formatore di impulsi di durata, in periodi di clock, pari al numero ricevuto
module Consumatore(dav_,rfd,numero,out,clockC,reset_);
 input        clockC,reset_;
 input        dav_;
 output       rfd;
 input[7:0]   numero;
 output       out;
 reg          RFD;  assign rfd=RFD;
 reg          OUT;  assign out=OUT;
 reg[7:0]     COUNT;
 reg[1:0]     STAR; parameter S0=0,S1=1,S2=2;
 always @(reset_==0) #1 begin RFD<=1; OUT<=0; STAR<=S0; end
 always @(posedge clockC) if (reset_==1) #3
   casex(STAR)
    S0: begin RFD<=1; OUT<=0; COUNT<=numero; STAR<=(dav_==1)?S0:S1; end
    S1: begin RFD<=0; STAR<=(dav_==0)?S1:S2; end
    S2: begin OUT<=1; COUNT<=COUNT-1; STAR<=(COUNT==1)?S0:S2; end
   endcase
endmodule
```