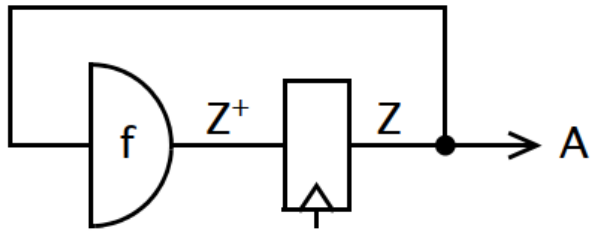
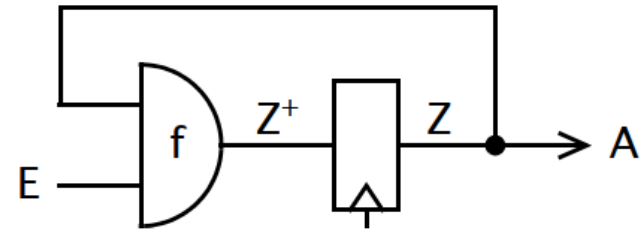
- Automatenmodelle

Autonomer Automat



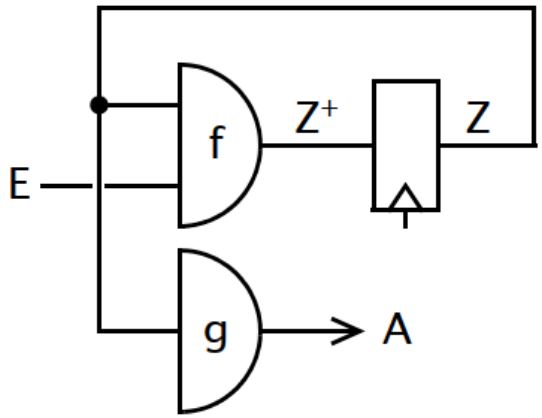
$Z^+ = f(Z), \quad A = Z$

Medwedjew-Automat



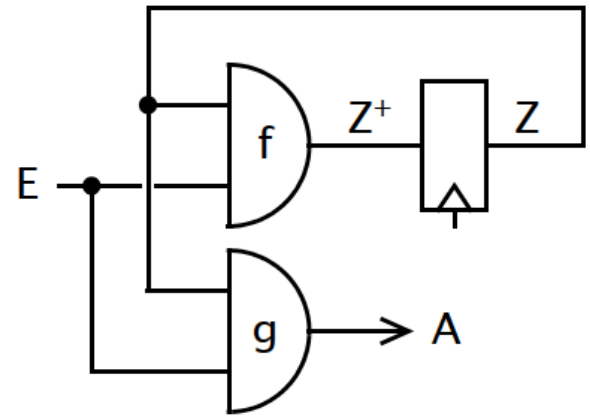
$Z^+ = f(Z, E), \quad A = Z$

Moore-Automat



$Z^+ = f(Z, E), \quad A = g(Z)$

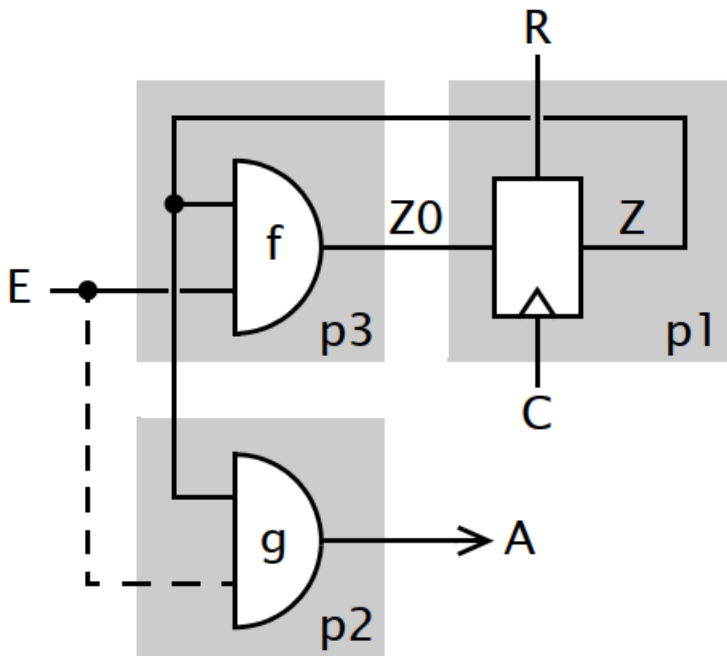
Mealy-Automat



$Z^+ = f(Z, E), \quad A = g(Z, E)$

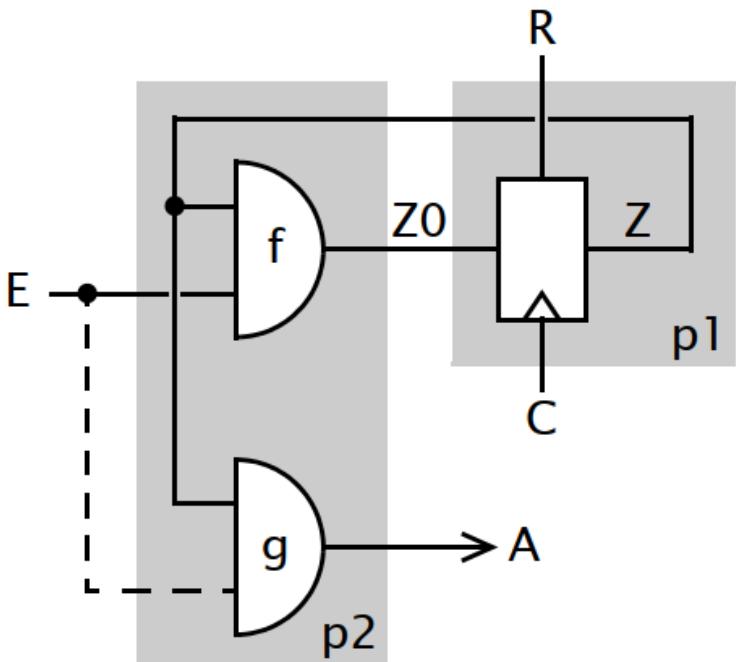
- Notation
  - Z der momentane Zustand
  - Z0 der Folgezustand (Ersatzsymbol für Z^+)
  - R asynchrones Reset
  - C Clock (Takt)
  - E Eingangssignale
  - A Ausgangssignale
  - f Übergangsschaltnetz
  - g Ausgangsschaltnetz

- Beschreibung von Mealy- und Moore-Automaten:
  - 3-Prozeß-Methode:
    - ein „getakteter“ Prozess beschreibt das Verhalten des Zustandsregisters
    - zwei „ungetaktete“ Prozesse beschreiben das Übergangsschaltnetz und das Ausgangsschaltnetz



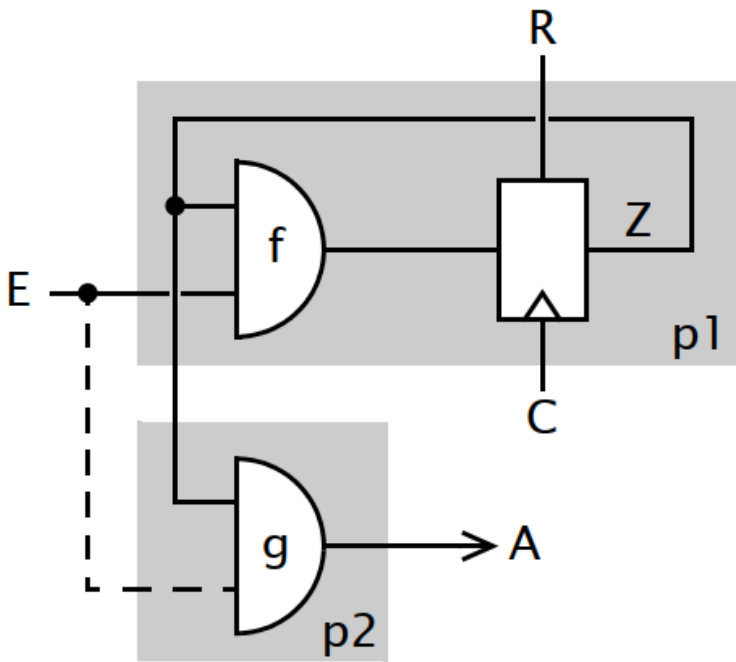

```
    SIGNAL Z, Z0: Zustand;

p1: PROCESS (R, C) IS -- Zustandsregister
       ...
       Z <= Z0;
       ...
    END PROCESS;
p2: PROCESS (E, Z) IS -- Ausgangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          A <= ...;
       END IF;
       ...
    END PROCESS;
p3: PROCESS (E, Z) IS -- Übergangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          Z0 <= ...;
       END IF;
       ...
    END PROCESS;
```

- Beschreibung von Mealy- und Moore-Automaten:
  - 2-Prozeß-Methode, 1. Variante:
    - ein „getakteter“ Prozess beschreibt das Verhalten des Zustandsregisters
    - ein „ungetakteter“ Prozess vereinigt in sich die Beschreibungen des Übergangs- schaltnetzes und des Ausgangschaltnetzes



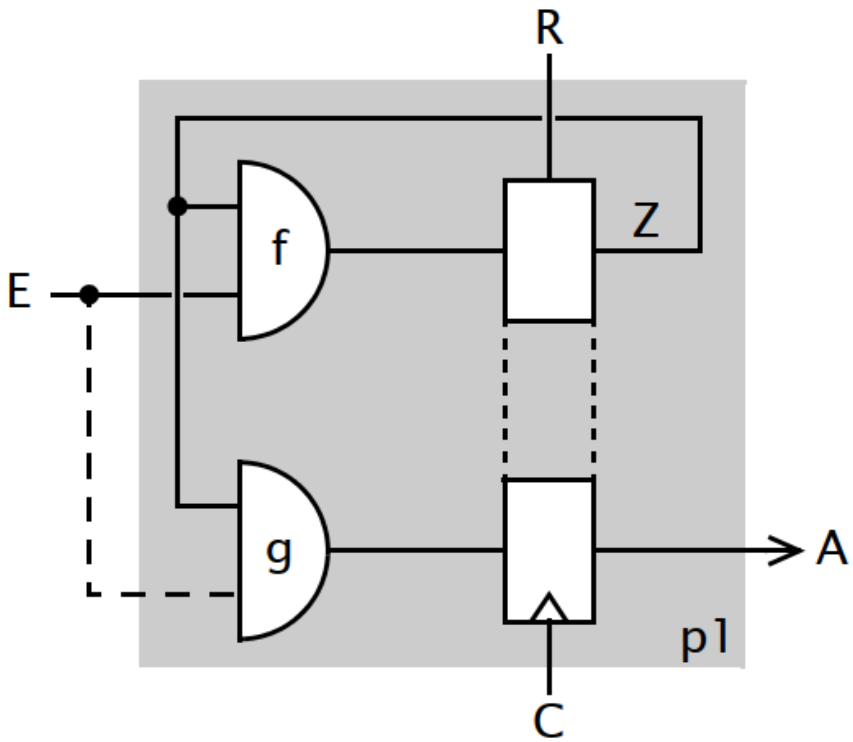

```
    SIGNAL Z, Z0: Zustand;

p1: PROCESS (R, C) IS -- Zustandsregister
       ...
       Z <= Z0;
       ...
    END PROCESS;

p2: PROCESS (E, Z) IS -- Ausgangsschaltnetz
                      -- Übergangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          A  <= ...;
          Z0 <= ...;
       END IF;
       ...
    END PROCESS;
```

- Beschreibung von Mealy- und Moore-Automaten:
  - 2-Prozeß-Methode, 2. Variante
    - ein „getakteter“ Prozess vereinigt in sich die Beschreibung des Zustandsregisters und die des Übergangsschaltnetzes
    - ein „ungetakteter“ Prozess beschreibt das Ausgangschaltnetz



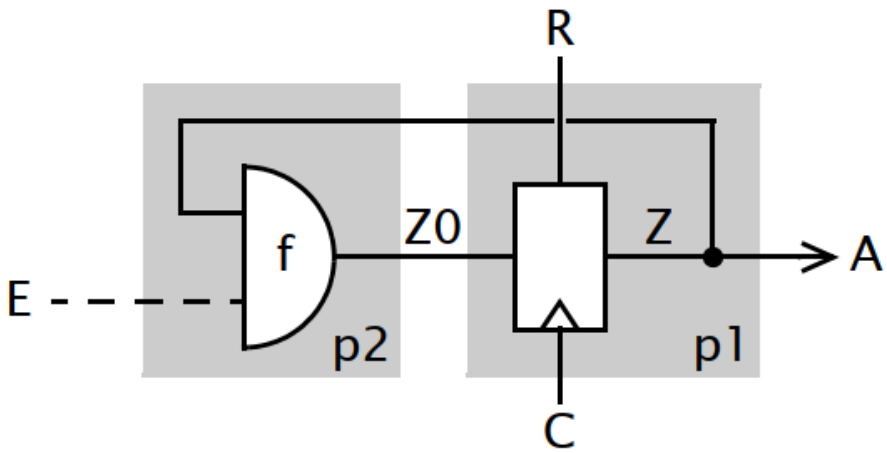

```
    SIGNAL Z: Zustand;

p1: PROCESS (R, C) IS -- Zustandsregister
                      -- Übergangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          Z <= ...;
       END IF;
       ...
    END PROCESS;

p2: PROCESS (E, Z) IS -- Ausgangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          A <= ...;
       END IF;
       ...
    END PROCESS;
```

- Beschreibung von Mealy- und Moore-Automaten:
  - 1-Prozeß-Methode:
    - ein „getakteter“ Prozes vereinigt in sich die Beschreibungen des Zustands-registers, des Übergangsschaltnetzes und des Ausgangsschaltnetzes.
    - Ausgangsflipflops beeinflussen das Verhalten des Schaltwerks



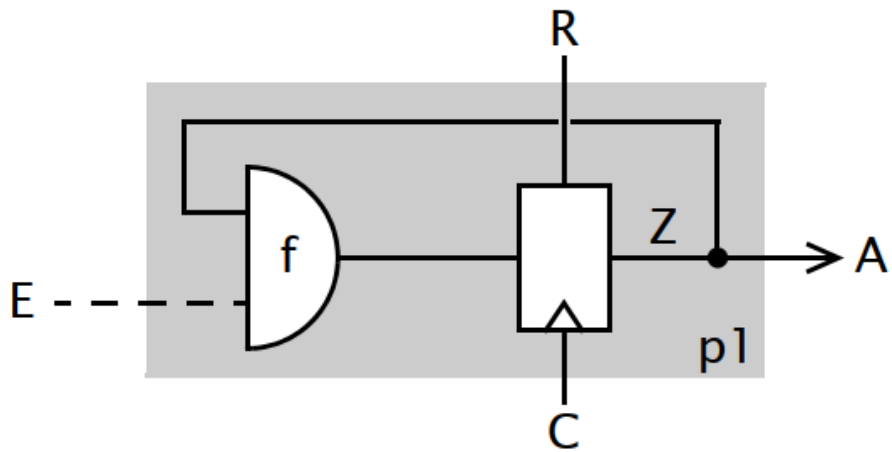

```
    SIGNAL Z: Zustand;

    -- Zustandsregister
    -- Übergangsschaltnetz
    -- Ausgangsschaltnetz + Flipflops
p1: PROCESS (R, C) IS
       ...
       IF Z=... AND E=... THEN
          A <= ...;
          Z <= ...;
       END IF;
       ...
    END PROCESS;
```

- Beschreibung von Autonomen und Medwedjew-Automaten:
  - 2-Prozeß-Methode:
    - ein „getakteter“ Prozess beschreibt das Verhalten des Zustandsregisters,
    - ein „ungetakteter“ Prozess beschreibt das Übergangsschaltnetz



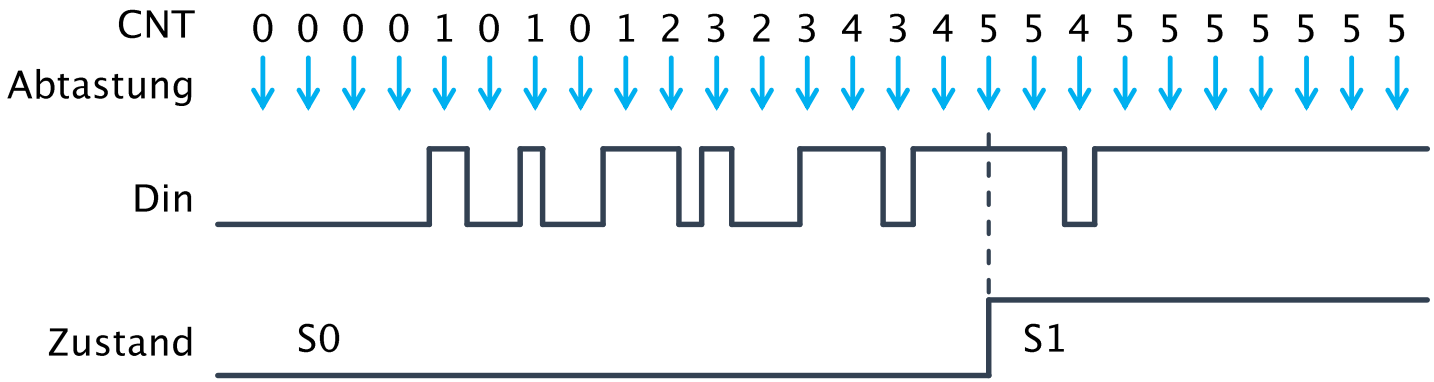

```
    SIGNAL Z, Z0: Zustand;

p1: PROCESS (R, C) IS -- Zustandsregister
       ...
       Z <= Z0;
       ...
    END PROCESS;

    A <= Z; -- Ausgangsschaltnetz

p2: PROCESS (E, Z) IS -- Übergangsschaltnetz
       ...
       IF Z=... AND E=... THEN
          Z0 <= ...;
       END IF;
       ...
    END PROCESS;
```

- Beschreibung von Autonomen und Medwedjew-Automaten:
  - 1-Prozeß-Methode:
    - ein „getakteter“ Prozess vereinigt in sich die Beschreibungen des Zustands-registers und des Übergangsschaltnetzes

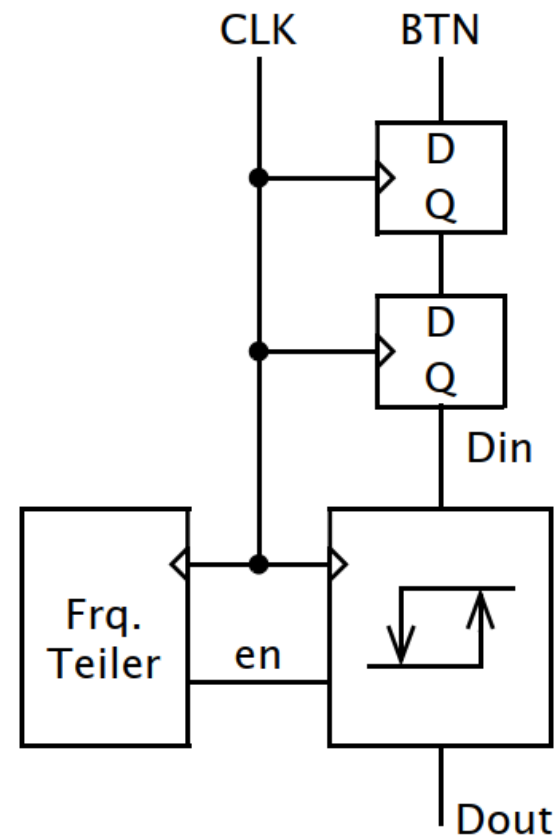



```
    SIGNAL Z: Zustand;

    -- Zustandsregister
    -- Übergangsschaltnetz
p1: PROCESS (R, C) IS
       ...
       IF Z=... AND E=... THEN
          Z <= ...;
       END IF;
       ...
    END PROCESS;

    A <= Z; -- Ausgangsschaltnetz
```

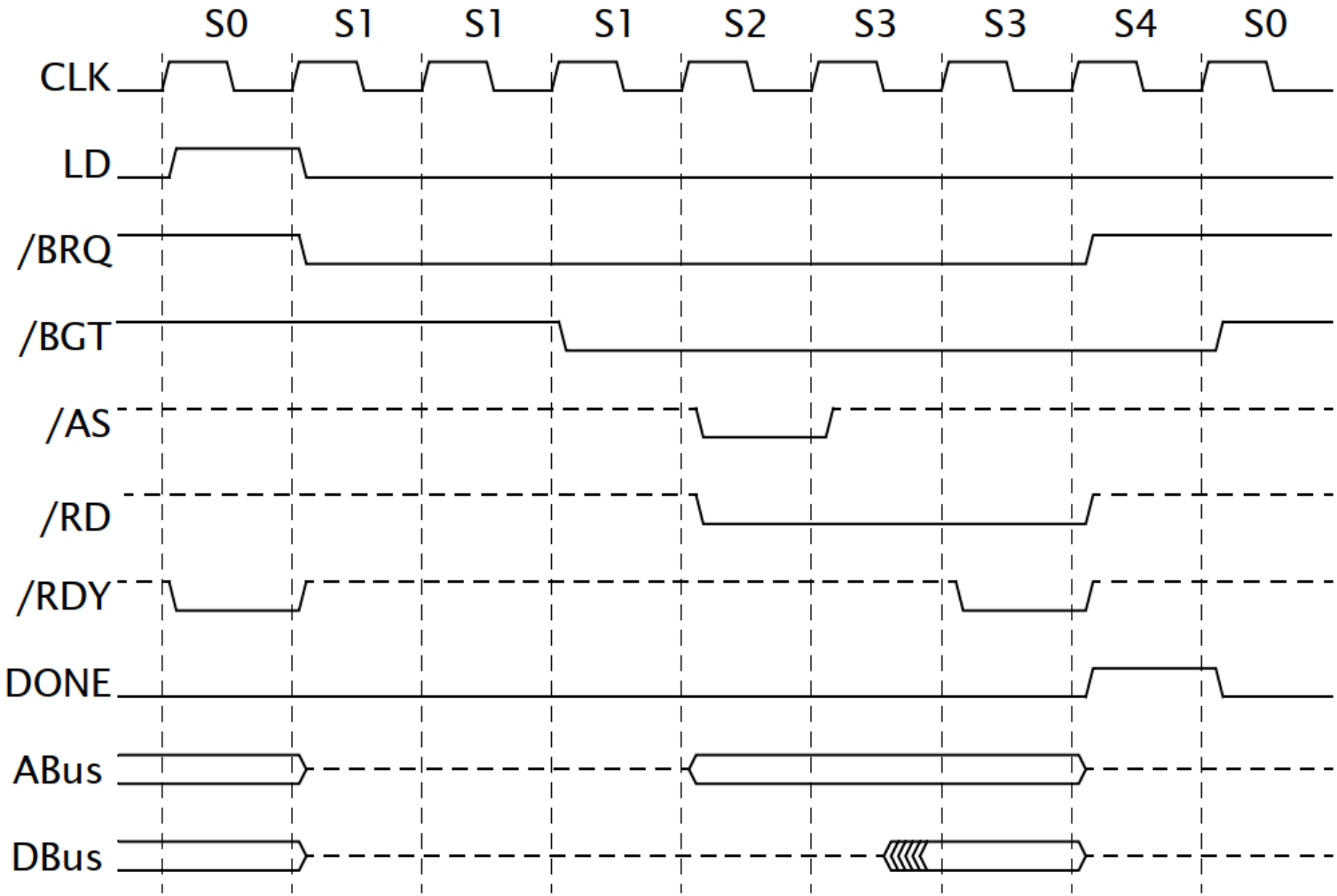
- Entprellen von Tasten durch Abtastung mit Hysterese



State	Din	CNT		State+	CNT+	Dout
S0	0	=0		S0	Cnt	0
S0	0	>0		S0	Cnt-1	0
S0	1	<N-1		S0	Cnt+1	0
S0	1	=N-1		S1	Cnt	0
S1	1	=N-1		S1	Cnt	1
S1	1	<N-1		S1	Cnt+1	1
S1	0	>0		S1	Cnt-1	1
S1	0	=0		S0	Cnt	1



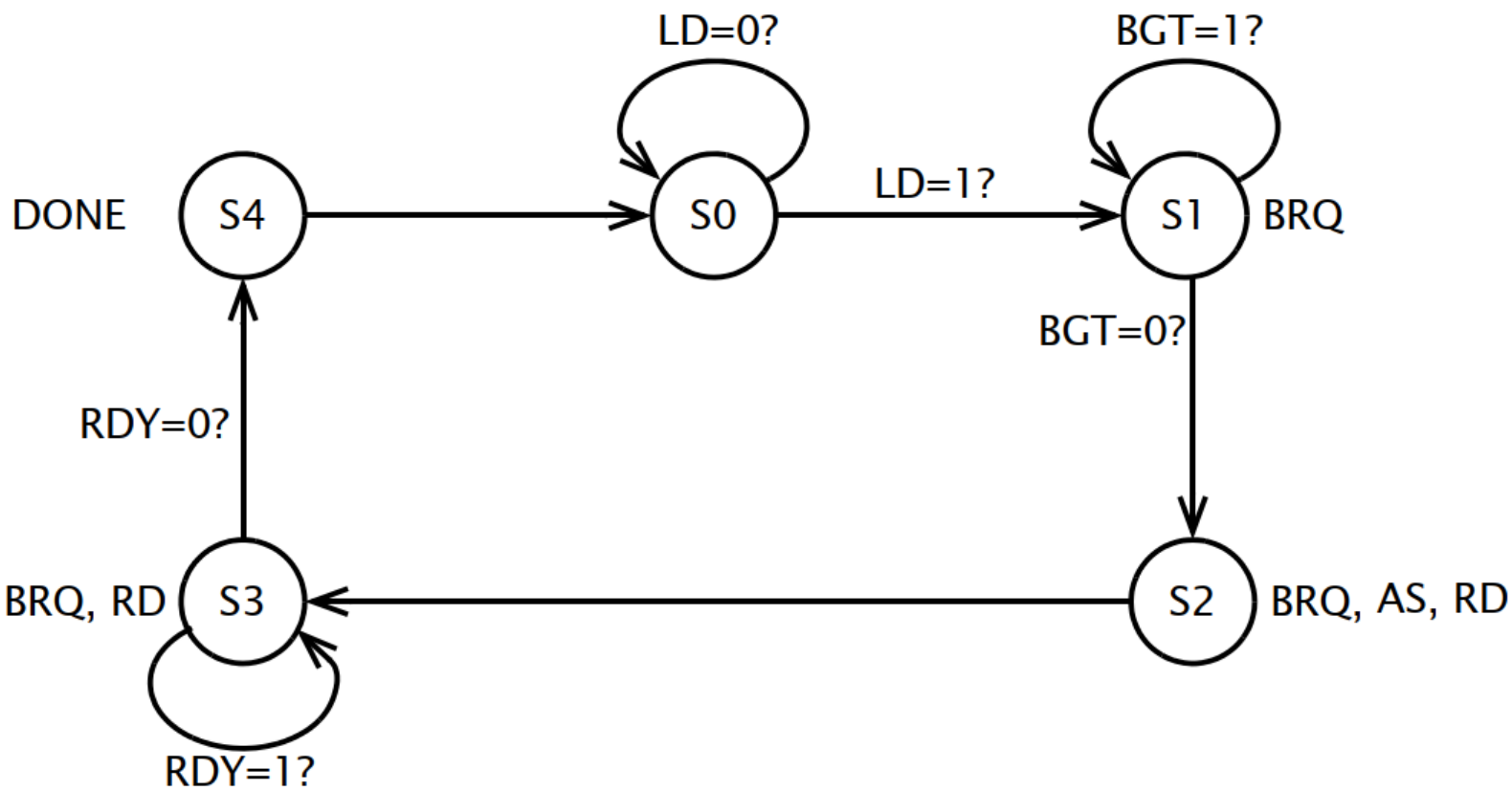
- Impulsdiagramm eines Busprotokolls



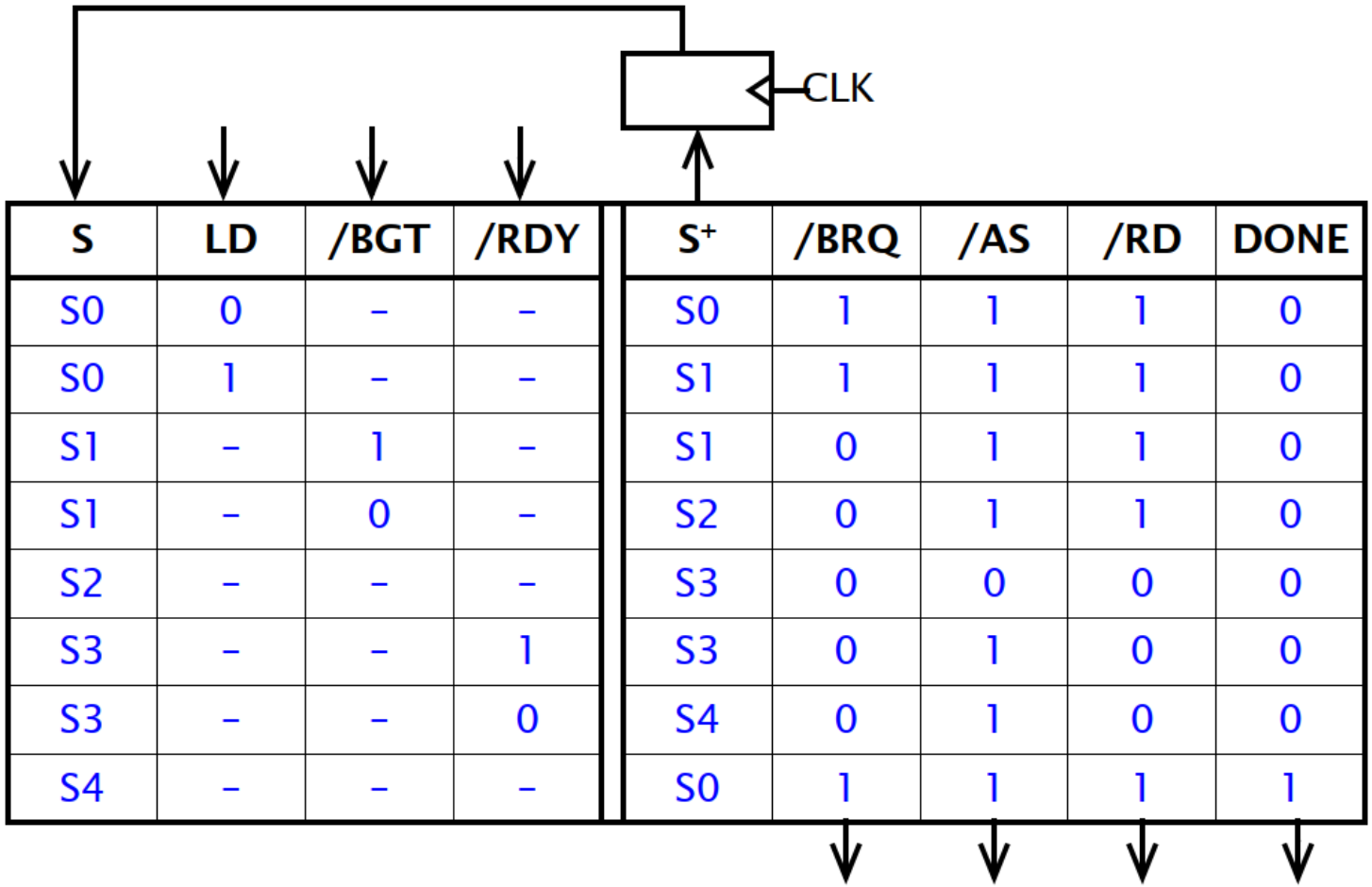
- Schnittstellenbeschreibung

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY bus_unit IS
   GENERIC(RSTDEF: std_logic := '1');
   PORT(RST:   IN  std_logic;  -- reset          RSTDEF active
        CLK:   IN  std_logic;  -- clock          rising edge active
        SWRST: IN  std_logic;  -- software reset RSTDEF active
        LD:    IN  std_logic;  -- load           high active, pulse
        BRQ:   OUT std_logic;  -- bus request    low active, (stable)
        BGT:   IN  std_logic;  -- bus grant      low active, (stable)
        AS:    OUT std_logic;  -- address strobe low active, pulse
        RD:    OUT std_logic;  -- read           low active, stable
        RDY:   IN  std_logic;  -- ready          low active, pulse
        DONE:  OUT std_logic); -- done           high active, pulse
END bus_unit;
```

- Zustandsgraph



- Steuertabelle für ein Moore-Steuerwerk



S	LD	/BGT	/RDY	S^+	/BRQ	/AS	/RD	DONE
S0	0	-	-	S0	1	1	1	0
S0	1	-	-	S1	1	1	1	0
S1	-	1	-	S1	0	1	1	0
S1	-	0	-	S2	0	1	1	0
S2	-	-	-	S3	0	0	0	0
S3	-	-	1	S3	0	1	0	0
S3	-	-	0	S4	0	1	0	0
S4	-	-	-	S0	1	1	1	1

- Algorithmus zur Umwandlung einer Steuertabelle für ein Moore-Steuerwerk in eine Steuertabelle für ein Mealy-Steuerwerk mit Ausgangsflipflops

Moore

i	state	cond	next	outs
1	S0	- 0	S0	1001
2	S0	- 1	S1	1001
3	S1	- -	S2	0111
4	S2	0 -	S2	1100
5	S2	1 -	S0	1100

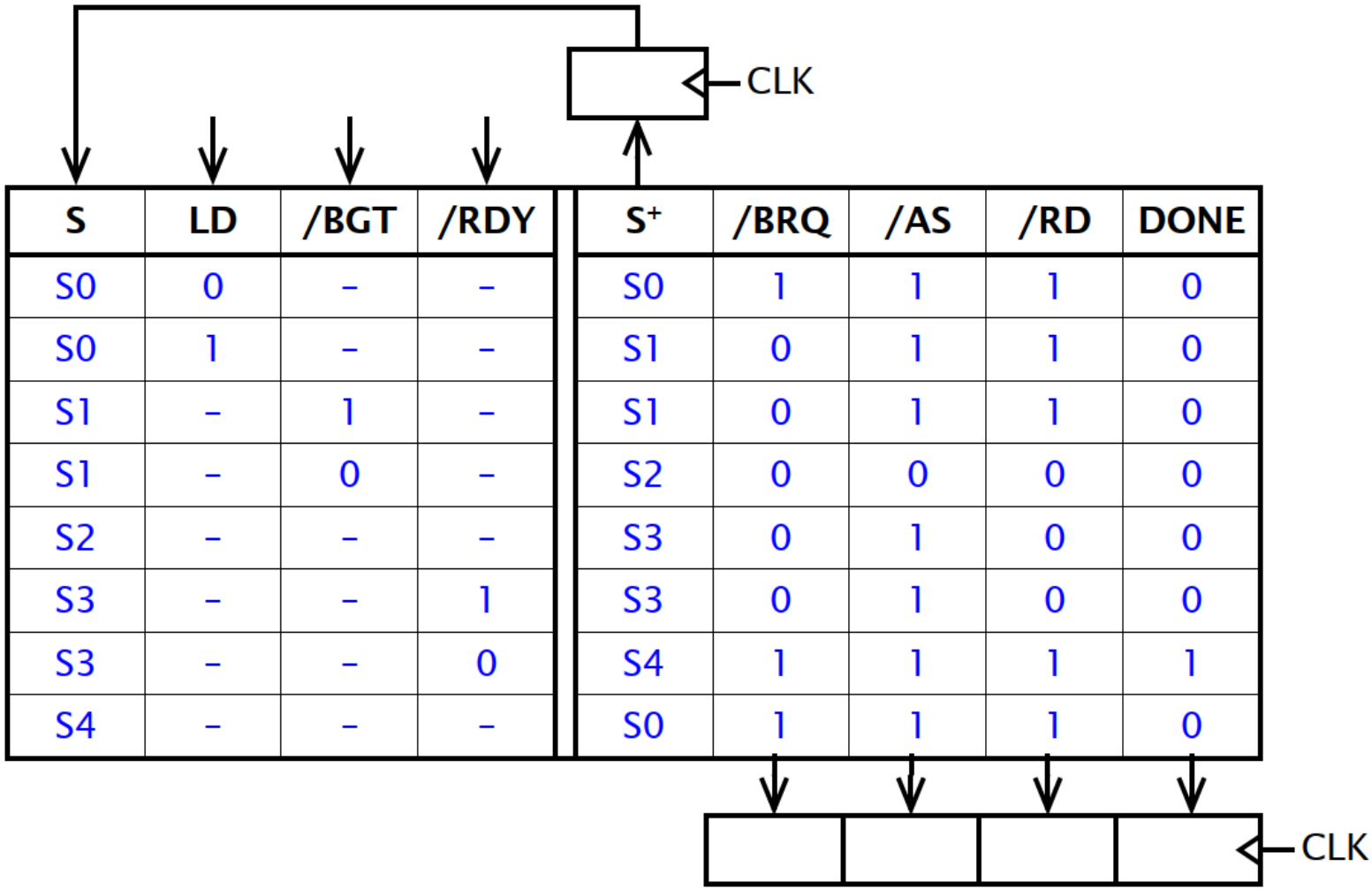
Mealy

i	state	cond	next	outs
1	S0	- 0	S0	1001
2	S0	- 1	S1	0111
3	S1	- -	S2	1100
4	S2	0 -	S2	1100
5	S2	1 -	S0	1001

```
int index(TState state) {
  for (int i=1; i<=N; i++)
    if (Moore[i].state == state)
      return i;
}
```

```
TState next; // Hilfsvariable
TOuts  outs; // Hilfsvariable

for (int i=1; i<=N; i++) {
  next = Mealy[i].next;
  outs = Moore[index(next)].outs;
  Mealy[i].outs = outs;
}
```

- Steuertabelle für ein Mealy-Steuerwerk mit Ausgangsflipflops



S	LD	/BGT	/RDY	S^+	/BRQ	/AS	/RD	DONE
S0	0	-	-	S0	1	1	1	0
S0	1	-	-	S1	0	1	1	0
S1	-	1	-	S1	0	1	1	0
S1	-	0	-	S2	0	0	0	0
S2	-	-	-	S3	0	1	0	0
S3	-	-	1	S3	0	1	0	0
S3	-	-	0	S4	1	1	1	1
S4	-	-	-	S0	1	1	1	0

- Architekturbeschreibung

```
ARCHITECTURE verhalten OF bus_unit IS
  TYPE TState IS (S0, S1, S2, S3, S4);
  SIGNAL state: TState;
  SIGNAL as_oe, rd_oe: std_logic;
BEGIN
  as <= '0' WHEN as_oe='1' ELSE 'Z';
  rd <= '0' WHEN rd_oe='1' ELSE 'Z';
  p1: PROCESS (rst, clk) IS
  BEGIN
    IF rst=RSTDEF THEN
      state <= S0;
      brq   <= '1';
      as_oe <= '0';
      rd_oe <= '0';
      done  <= '0';
    ELSIF rising_edge(clk) THEN
      brq   <= '1';
      as_oe <= '0';
      rd_oe <= '0';
      done  <= '0';
      CASE state IS
        WHEN S0 =>
          IF ld='1' THEN
            state <= S1;
            brq   <= '0';
          END IF;
        WHEN S1 =>
          brq <= '0';
          IF bgt='0' THEN
            state <= S2;
            as_oe <= '1';
            rd_oe <= '1';
          END IF;
        WHEN S2 =>
          state <= S3;
          brq   <= '0';
          rd_oe <= '1';
        WHEN S3 =>
          brq   <= '0';
          rd_oe <= '1';
          IF rdy='0' THEN
            done  <= '1';
            state <= S4;
           END IF;
        WHEN S4 =>
          state <= S0;
      END CASE;
    END IF;
  END PROCESS;
END verhalten;
```