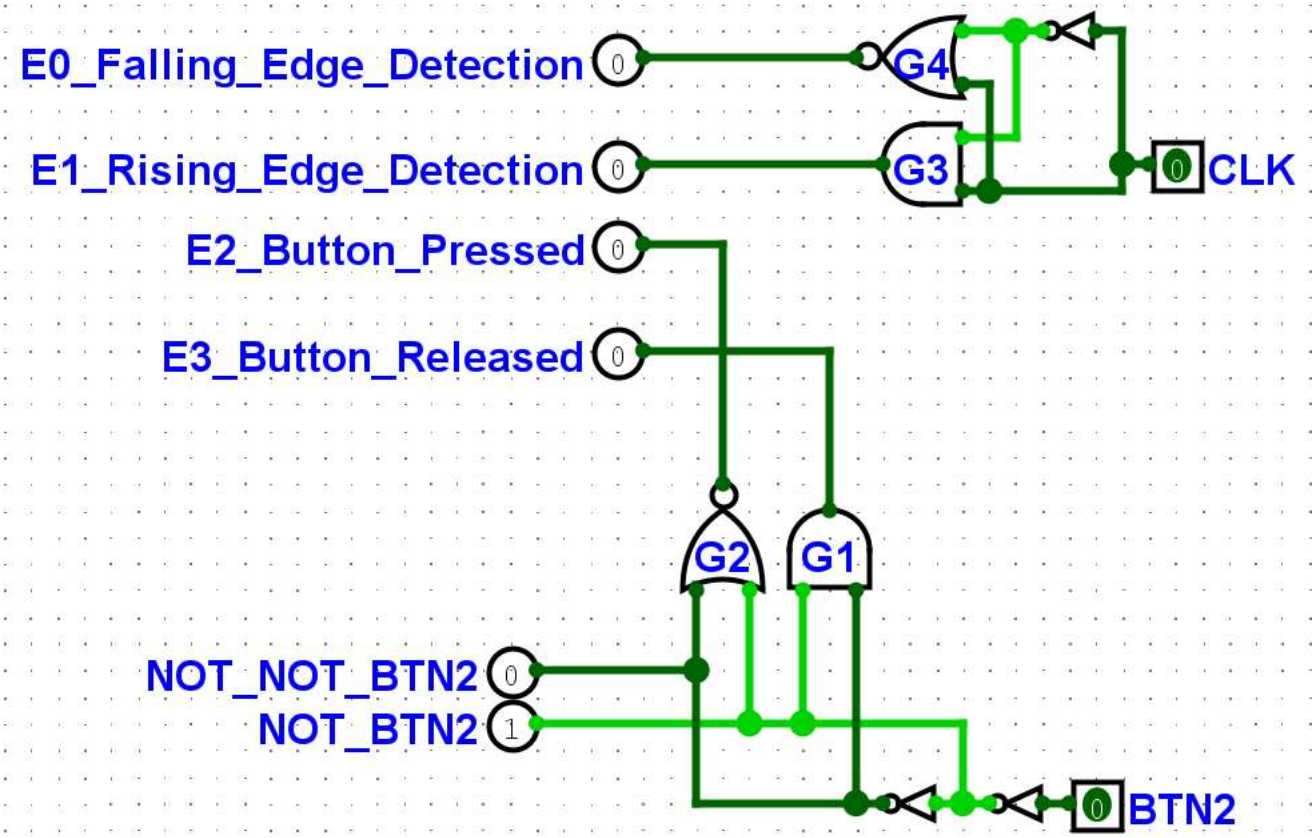
Aufgabe 1 (Messung von Verzögerungs- und Anstiegszeiten): Teilaufgabe a+b

Mittwoch, 16. März 2022 07:48



b) Wellenform-Diagramm

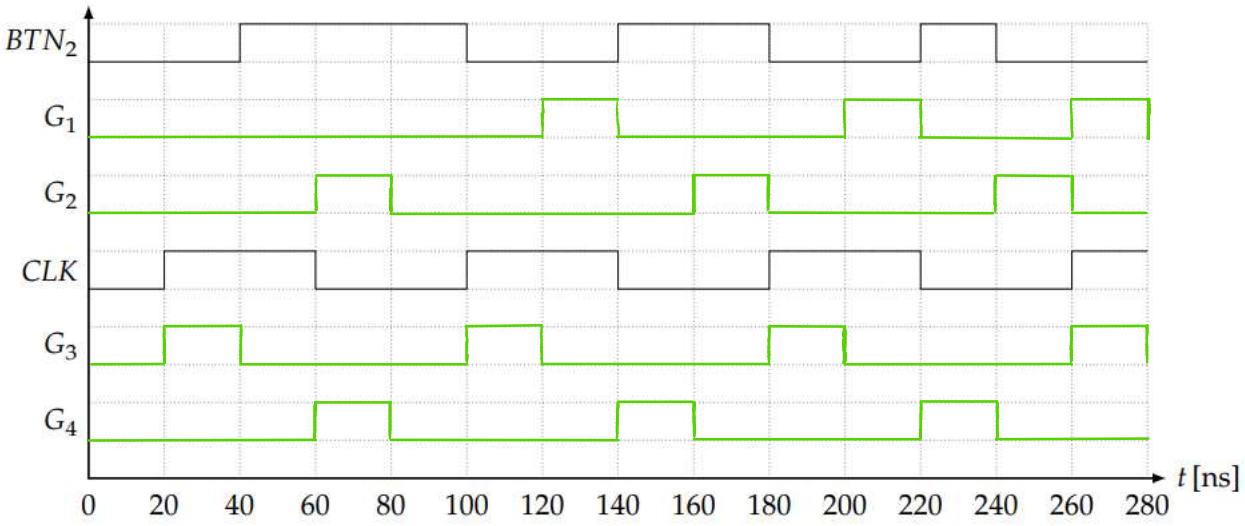
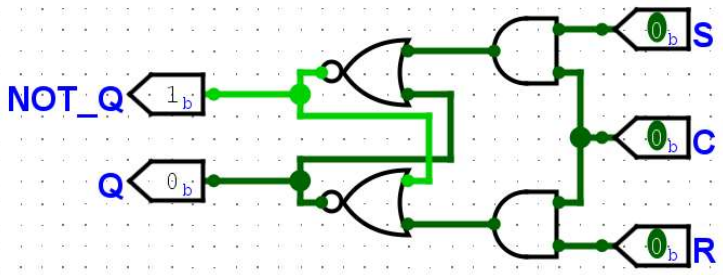
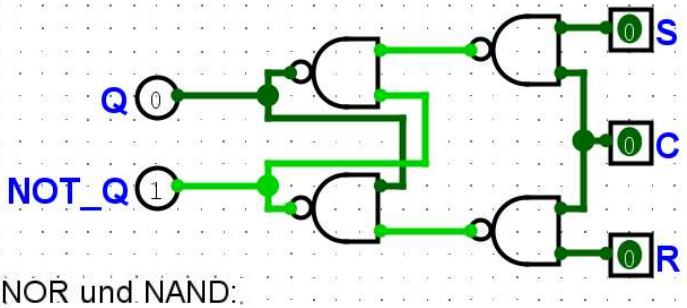


Abbildung 3: Signalverlauf für Gatter G_1 bis G_4, CLK und BTN_2

Aufgabe 1 (Messung von Verzögerungs- und Anstiegszeiten): Teilaufgabe c+d

Mittwoch, 16. März 2022 08:16

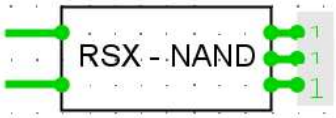




Unterscheiden von NOR und NAND:

NOR ist im illegalen Zustand LOW:

NAND ist im illegalen Zustand HIGH:



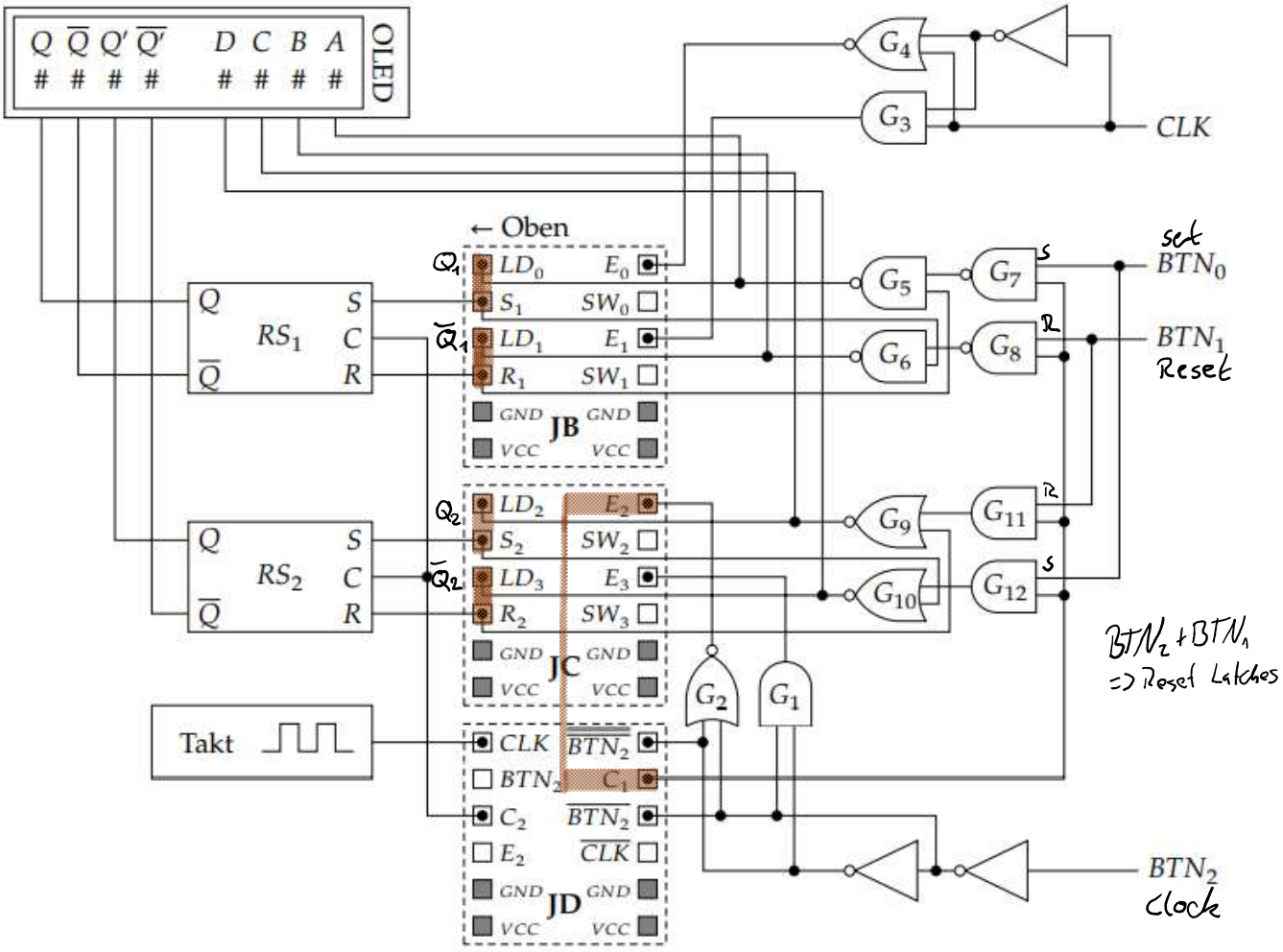
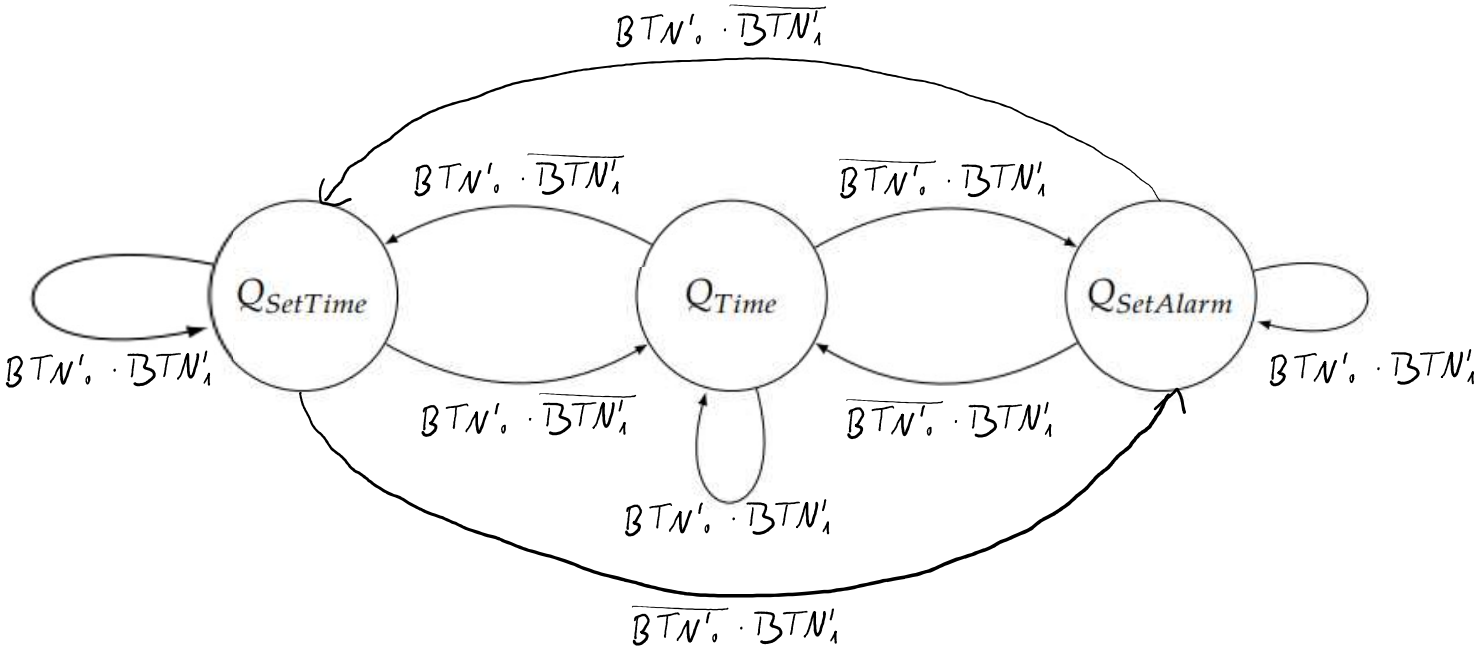


Abbildung 2: Beschaltung der Anschlüsse **JB** bis **JD** nach Programmierung des FPGAs

Aufgabe 2 (Entwurf eines Digitalen Weckers): Teilaufgabe a+b

Mittwoch, 16. März 2022 08:26

a) Zustandsdiagramm



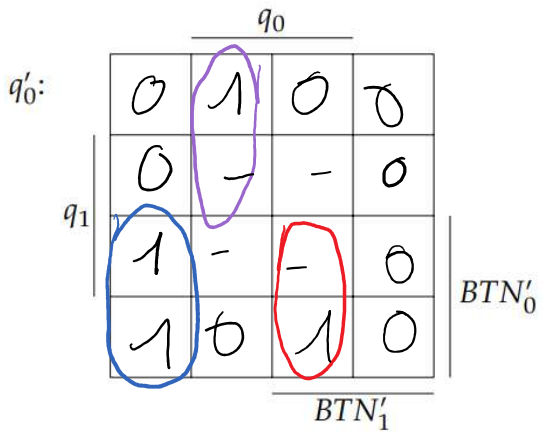
b) Automatentafel

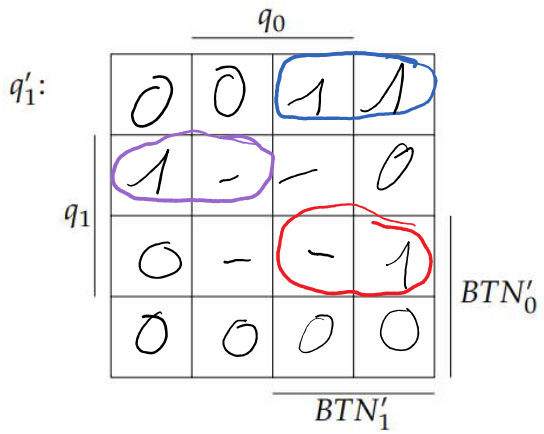
s^t		s^{t+1}					
		BTN'_0		BTN'_1		$BTN'_0 \cdot BTN'_1$	
q_1	q_0	q'_1	q'_0	q'_1	q'_0	q'_1	q'_0
0	0	0	1	1	0	0	0
0	1	0	0	1	0	0	1
1	0	0	1	0	0	1	0

Aufgabe 2 (Entwurf eines Digitalen Weckers): Teilaufgabe c+d

Mittwoch, 16. März 2022 08:43

c) Bestimmen Sie aus der Automatentafel die disjunktive Minimalform (DMF) der Zustandsüberführungsfunktion. Nutzen Sie die folgenden Symmetriediagramme und achten Sie auf die Variablenordnung:

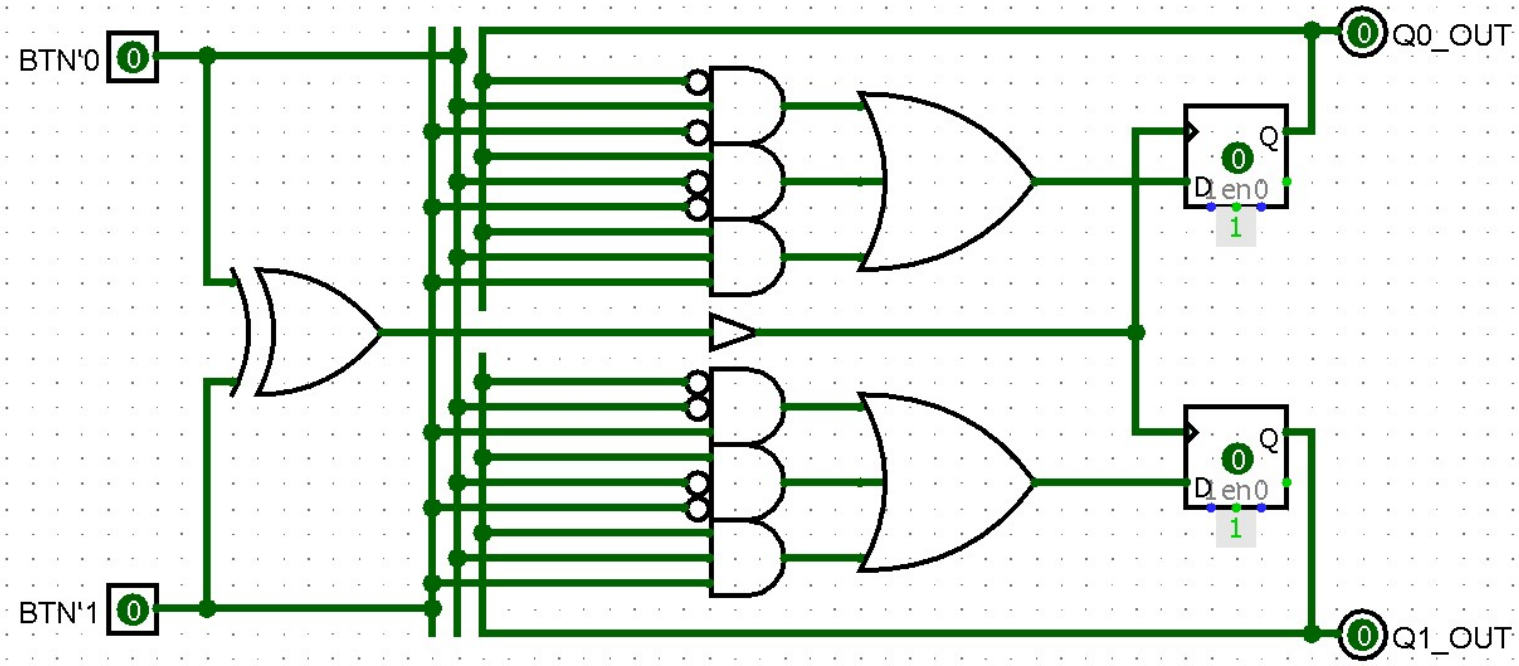




$\overline{q_0} \cdot BTN'_0 \cdot \overline{BTN'_1} +$
$q_0 \cdot \overline{BTN'_0} \cdot \overline{BTN'_1} +$
$q_0 \cdot BTN'_0 \cdot BTN'_1$

$\overline{q_1} \cdot \overline{BTN'_0} \cdot BTN'_1 +$
$q_1 \cdot \overline{BTN'_0} \cdot \overline{BTN'_1} +$
$q_1 \cdot BTN'_0 \cdot BTN'_1$

d) Wecker-Schaltwerk



Aufgabe 2 (Entwurf eines Digitalen Weckers): Teilaufgabe e

Mittwoch, 16. März 2022 09:31

```
    FOR i IN 0 TO 7 LOOP
        --TODO: Pseudocode umsetzen
        IF v_bcd0 > 4 then
            v_bcd0 := v_bcd0 + 3;
        END IF;
        IF v_bcd1 > 4 then
            v_bcd1 := v_bcd1 + 3;
        END IF;


        v_bcd1 := shift_left(v_bcd1, 1);
        v_bcd1(0) := v_bcd0(3);

        v_bcd0 := shift_left(v_bcd0, 1);
        v_bcd0(0) := v_number(7 - i); -- Laut Pseudocode: 8, aber 7 da A 0-indexed ist
        ----------------------------
    END LOOP;
    s_bcd0 <= v_bcd0;
    s_bcd1 <= v_bcd1;
END PROCESS;
o_ascii0 <= "0011" & std_logic_vector(s_bcd0); -- converting to 8 bits ascii
o_ascii1 <= "0011" & std_logic_vector(s_bcd1); -- converting to 8 bits ascii
```

Aufgabe 2 (Entwurf eines Digitalen Weckers): Teilaufgabe f

Mittwoch, 16. März 2022 09:25

```
-- TODO: Zaehle Uhr hoch
if sectrigger = '1' then
  secs <= secs + 1;
  if secs >= 59 then
    secs <= 0;
    mins <= mins + 1;
    if mins >= 59 then
      mins <= 0;
      hours <= hours + 1;
      if hours >= 23 then
        hours <= 0;
      end if;
    end if;
  end if;
end if;
-- TODO: Pruefe, ob Alarm ausgeloest werden muss
if hours = whours and mins = wmins and sw(0) = '1' then
  alarm <= '1';
else
  alarm <= '0';
end if;
case current_state is
  -- Zustand Time
  when NTIME =>
    -- TODO: Setze naechsten Zustand
    if btn_triggered(0) = '1' and btn_triggered(1) = '0' then
      next_state := SET_TIME;
    elsif btn_triggered(0) = '0' and btn_triggered(1) = '1' then
      next_state := SET_ALARM;
    end if;
  -- Zustand SetTime
  when SET_TIME =>
    -- TODO: Setze naechsten Zustand
    if btn_triggered(0) = '1' and btn_triggered(1) = '0' then
      next_state := NTIME;
    elsif btn_triggered(0) = '0' and btn_triggered(1) = '1' then
      next_state := SET_ALARM;
    end if;
    -- TODO: Setze Minute und Stunde mit BTN(2) bzw. BTN(3)
    if btn_triggered(2) = '1' then -- oder btn(2) = 1 and fasttrigger = 1,
                                   -- Um schnelleres Uhrstellen durch
                                   -- gedrückt halten zu ermöglichen
      mins <= mins + 1;
      if mins >= 59 then
        mins <= 0;
      end if;
    end if;
    -- Hours:
    if btn_triggered(3) = '1' then
      hours <= hours + 1;
      if hours > 23 then
         hours <= 0;
      end if;
    end if;
  -- Zustand SetAlarm
  when SET_ALARM =>
    -- TODO: Setze naechsten Zustand
    if btn_triggered(0) = '1' and btn_triggered(1) = '0' then
      next_state := SET_TIME;
    elsif btn_triggered(0) = '0' and btn_triggered(1) = '1' then
      next_state := NTIME;
    end if;
    -- TODO: Setze Minute und Stunde mit BTN(2) bzw. BTN(3)
    if btn_triggered(2) = '1' then -- oder btn(2) = 1 and fasttrigger = 1,
                                   -- Um schnelleres Uhrstellen durch
                                   -- gedrückt halten zu ermöglichen
      wmins <= wmins + 1;
      if wmins >= 59 then
        wmins <= 0;
      end if;
    end if;
    -- Hours:
    if btn_triggered(3) = '1' then
      whours <= whours + 1;
      if whours >= 23 then
        whours <= 0;
      end if;
    end if;
  -- Illegale Zustaende
  when others =>
    next_state := NTIME;
end case;

current_state <= next_state;
```