AGSTU
Arbete Genom STUdier
Utbildning

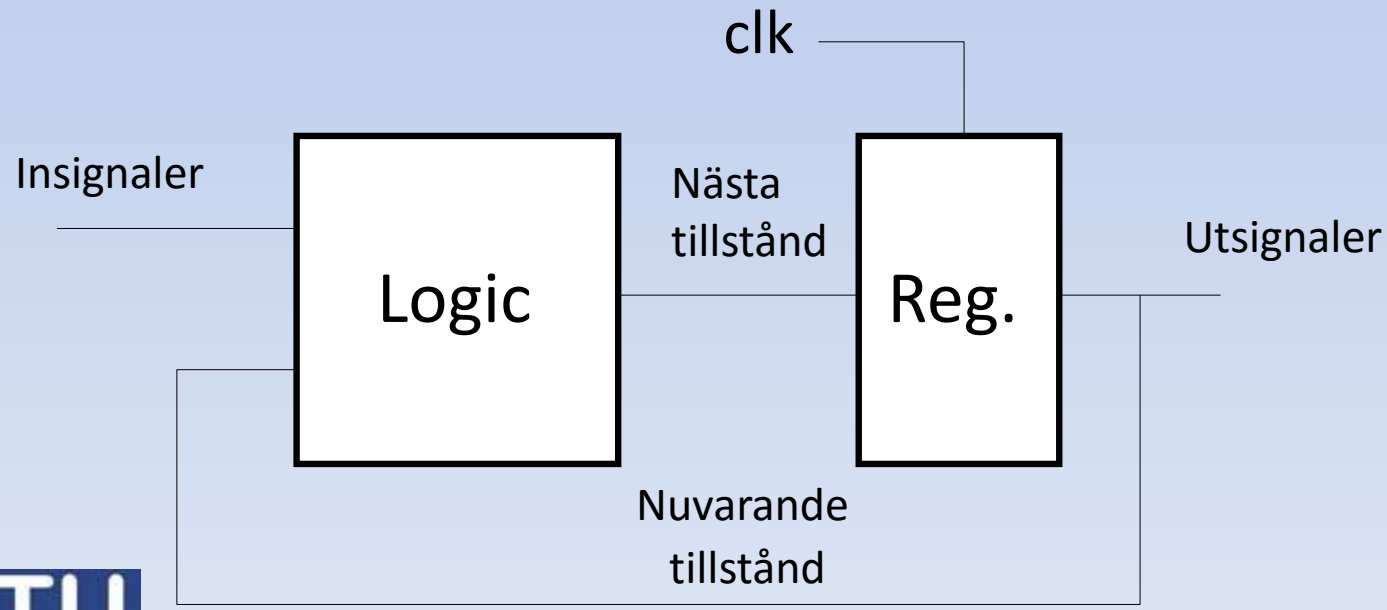
VHDL-programmering för inbyggda system
Välkommen

- Olika typer av tillståndsmaskiner
  - Mealy
  - Moore
  - Synkrona utgångar
  - Mealy+Moore+synkrona (Mealy+Moore)
  - Medvedev

AGSTU
Arbete Genom STUdier
Utbildning

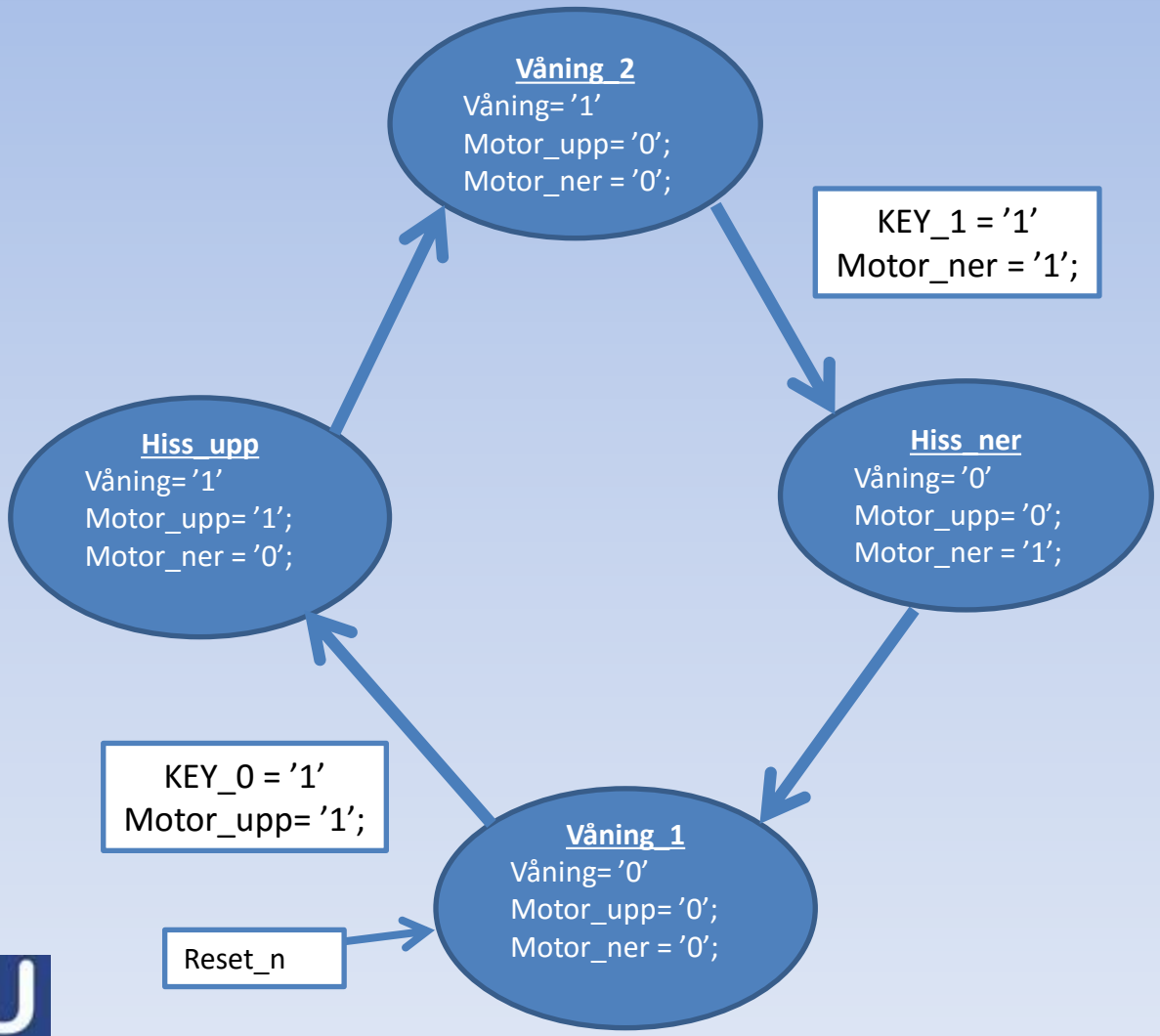
Tillståndsmaskin – grunderna

- Synkron maskin – asynkron reset
- Tidsbeteendet – system klockan



AGSTU
Arbete Genom STUdier
Utbildning

Vi fortsätter på hissen!



AGSTU
Arbete Genom STUdier
Utbildning

State Encoding

```
type state_type is (vaning_1,vaning_2, hiss_upp, hiss_ner);

-- Register to hold the current state
signal state : state_type;

attribute syn_encoding : string;
attribute syn_encoding of state_type : type is "gray";
```

gray

	Name	state_bit_1	state_bit_0
1	vaning_1	0	0
2	vaning_2	0	1
3	hiss_upp	1	1
4	hiss_ner	1	0

sequential

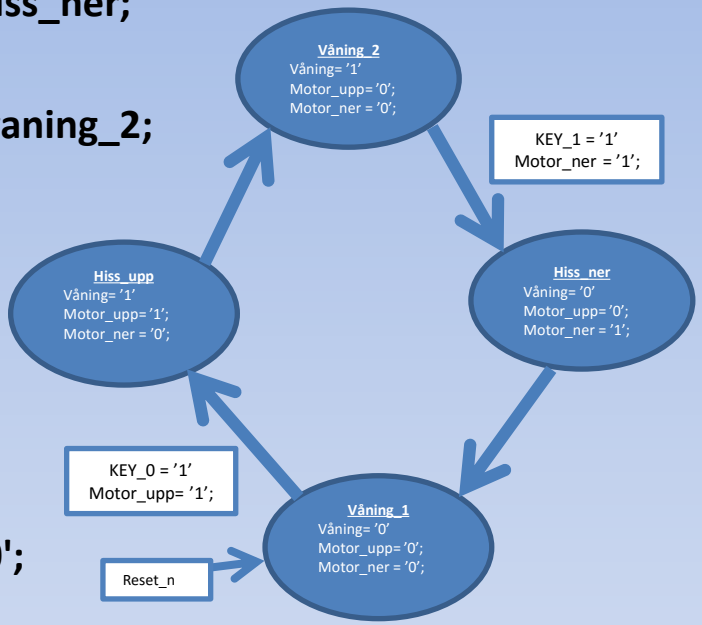
	Name	state_bit_1	state_bit_0
1	vaning_1	0	0
2	vaning_2	0	1
3	hiss_upp	1	0
4	hiss_ner	1	1

one-hot

	Name	hiss_ner	hiss_upp	vaning_2	vaning_1
1	vaning_1	0	0	0	0
2	vaning_2	0	0	1	1
3	hiss_upp	0	1	0	1
4	hiss_ner	1	0	0	1

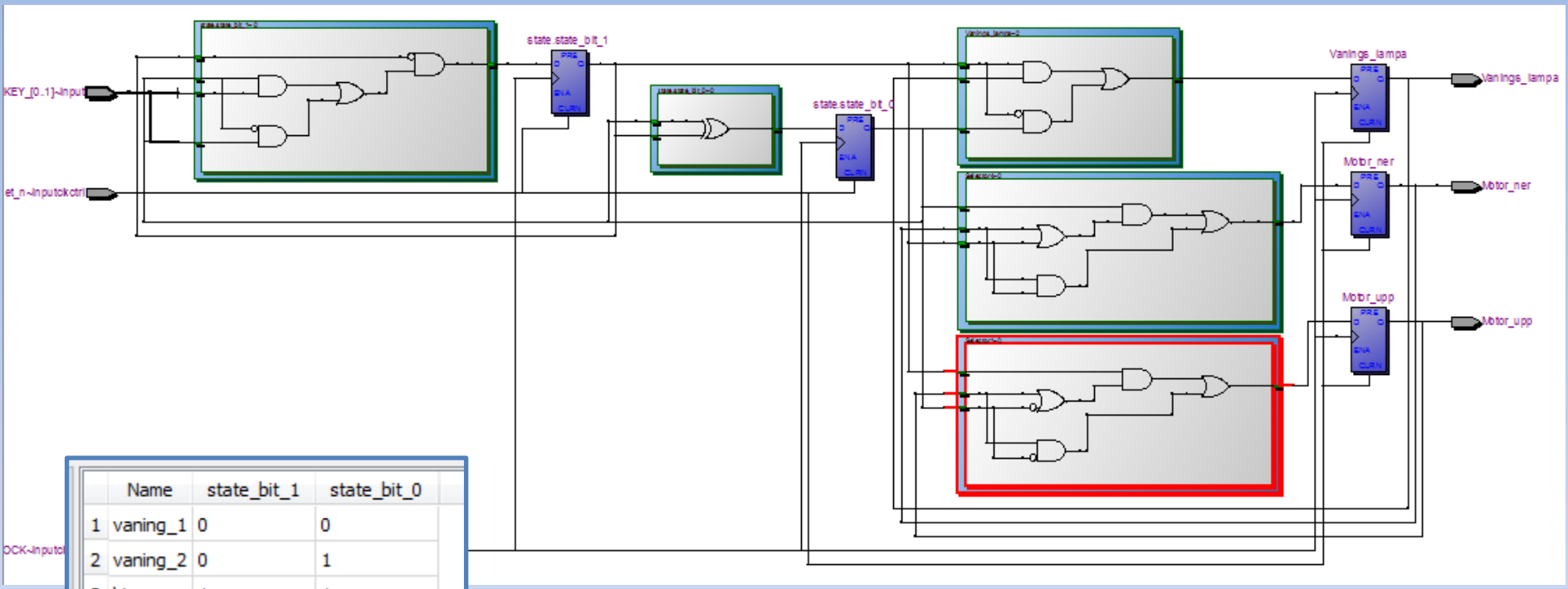
AGSTU
Arbete Genom STUdier
Utbildning

```
case state is
        when vaning_2=> -- hissen på våning 2
                Vanings_lampa <= '1'; motor_upp <= '0';
                if KEY_0 = '1' then -- knapp intryckt på våning 1
                        motor_ner <= '1';state <= hiss_ner;
                else
                        motor_ner <= '0'; state <= vaning_2;
                end if;
        when hiss_upp =>
                motor_upp <= '1'; state <= vaning_2;
        when hiss_ner =>
                motor_ner <= '1';state <= vaning_1;
        when vaning_1 => -- hissen på våning 1
                Vanings_lampa <= '0'; motor_ner <= '0';

                if KEY_1 = '1' then -- knapp intryckt på våning 2
                        motor_upp <= '1'; state <= hiss_upp;
                else
                        motor_upp <= '0'; state <= vaning_1;
                end if;
```

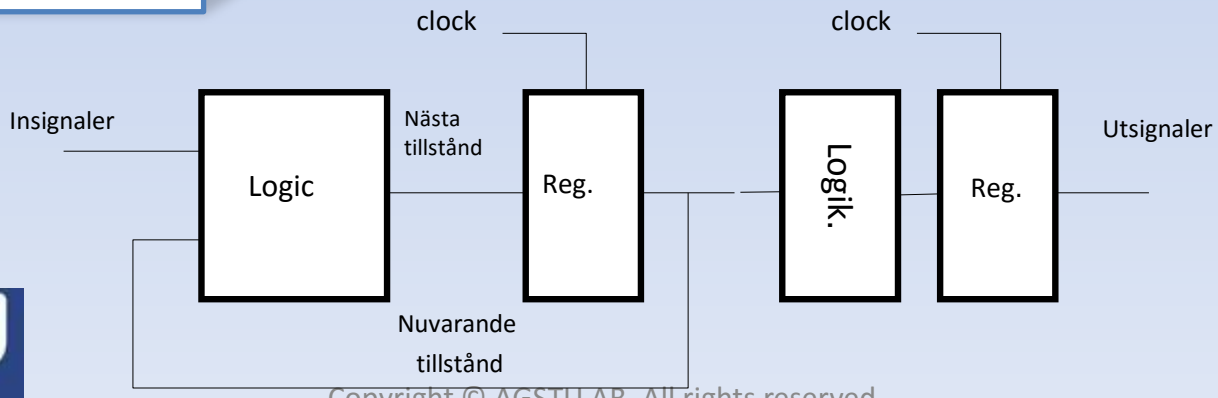


AGSTU
Arbete Genom STUdier
Utbildning

RTL nivå (sequential avkodning)



	Name	state_bit_1	state_bit_0
1	vaning_1	0	0
2	vaning_2	0	1
3	hiss_upp	1	1
4	hiss_ner	1	0



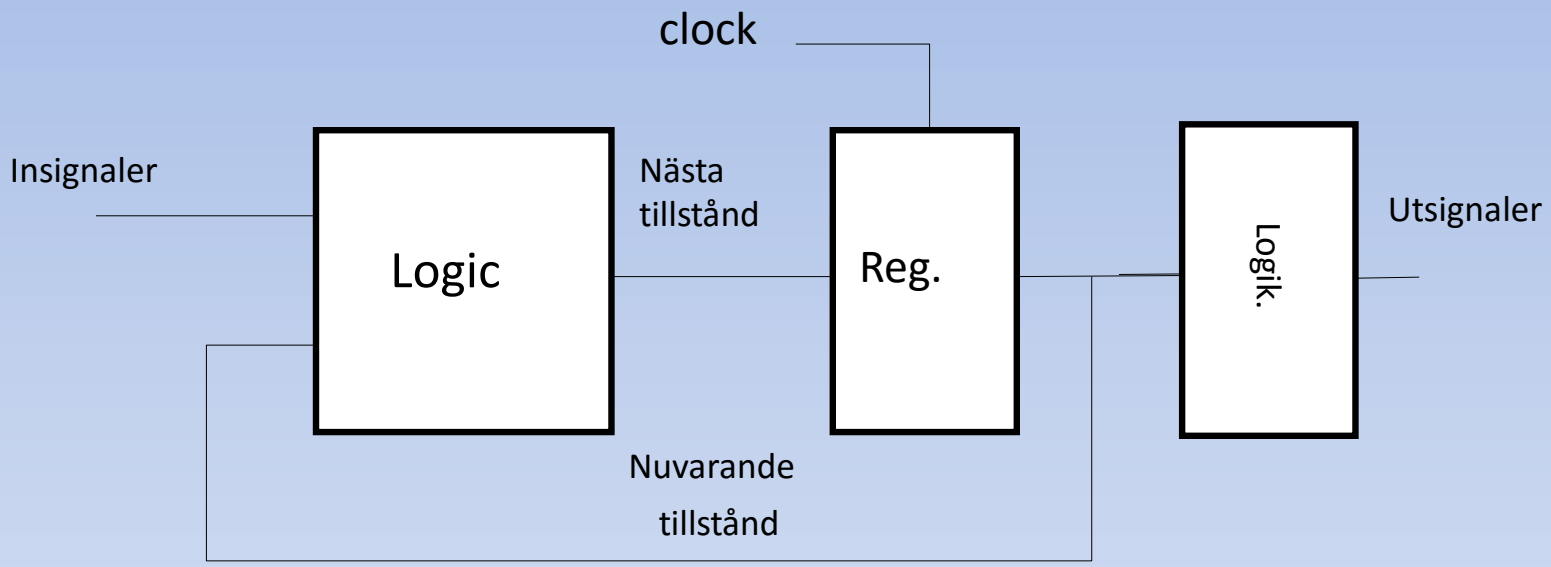
AGSTU Arbete Genom STUdier Utbildning

Tillståndsmaskins typer

- Moore: utgångar är enbart en funktion av tillståndet
- Mealy: utgångarna är en funktion av nuvarande tillstånd och samtliga ingångar
- Olika typer av maskiner, den vi använde var en Moore maskin med synkrona utgångar (se boken)

AGSTU
Arbete Genom STUdier
Utbildning

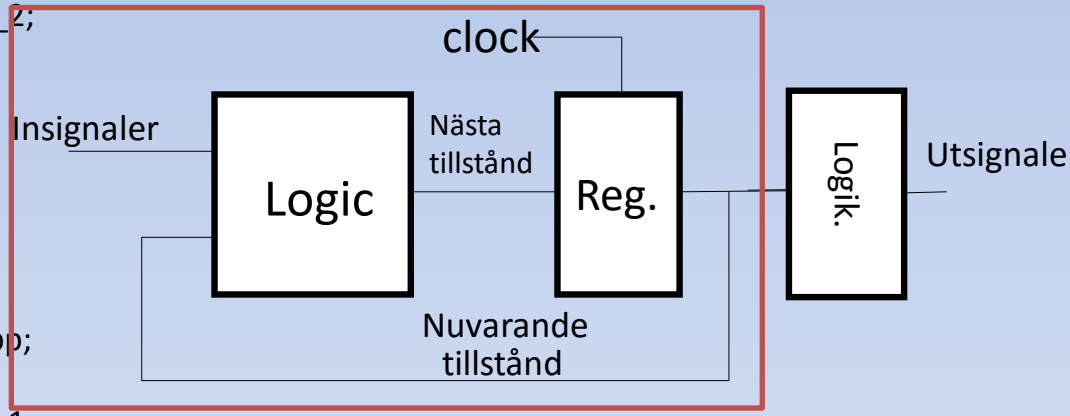
Moore tillståndsmaskin



Moore: utgångar är enbart en funktion av tillståndet

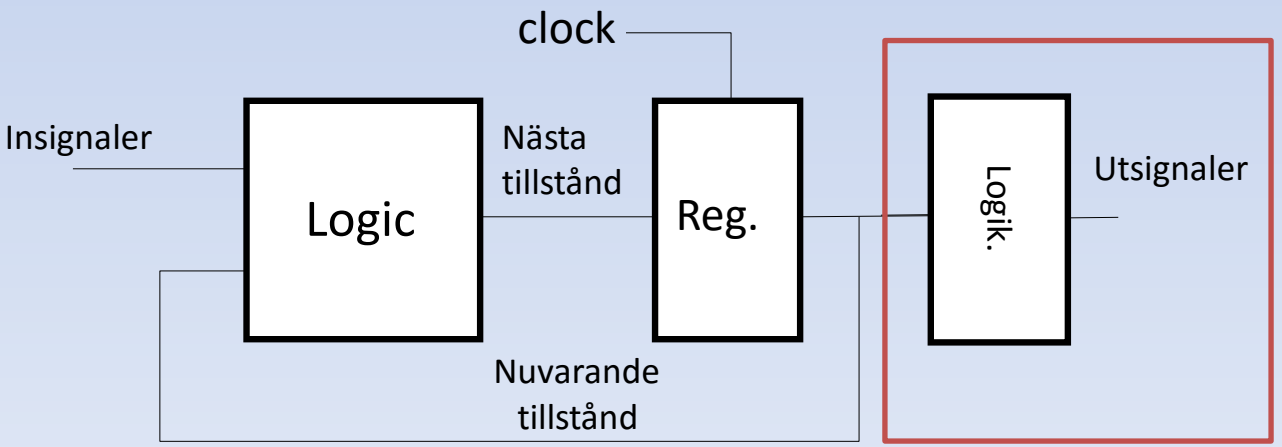
Moore - Hiss

```
process (reset_n, CLOCK)
begin
if reset_n = '0' then
                state <= vaning_1;
elsif (rising_edge(CLOCK)) then
                case state is
                when vaning_2=>
                                if KEY_0 = '1' then
                                                state <= hiss_ner;
                                else
                                                state <= vaning_2;
                                end if;
                when hiss_upp =>
                                state <= vaning_2;
                when hiss_ner =>
                                state <= vaning_1;
                when vaning_1 =>
                                if KEY_1 = '1' then
                                                state <= hiss_upp;
                                else
                                                state <= vaning_1;
                                end if;
                when others =>
                                state <= vaning_1;
                end case;
end if;
end process;
```

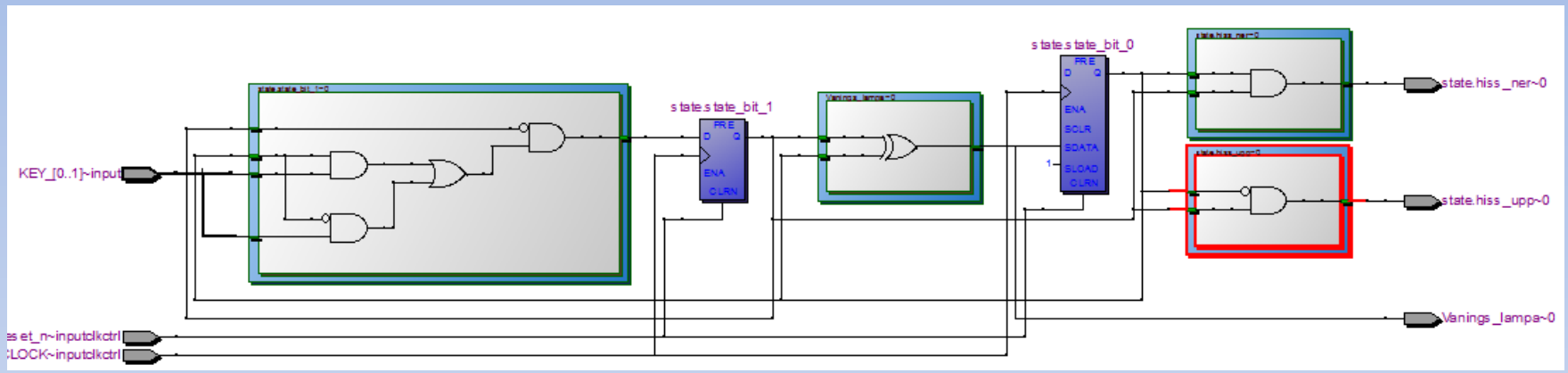


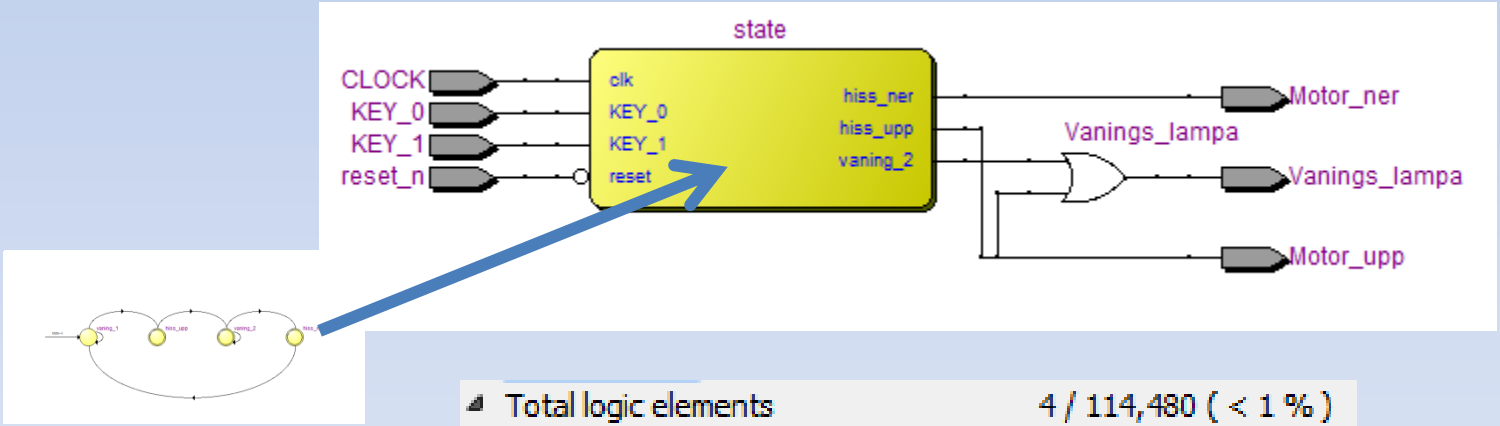
Moore - Hiss

```
process (state)
begin
case state is
when vaning_2=>
        Vanings_lampa <= '1'; motor_upp <= '0'; motor_ner <= '0';
when hiss_upp =>
        motor_upp <= '1'; motor_ner <= '0'; Vanings_lampa <= '1';
when hiss_ner =>
        motor_ner <= '1'; motor_upp <= '0'; Vanings_lampa <= '0';
when vaning_1 =>
        Vanings_lampa <= '0'; motor_upp <= '0'; motor_ner <= '0';
when others =>
        motor_ner <= '0'; motor_upp <= '0'; Vanings_lampa <= '0';
end case;
end process;
```



Moore - Hiss

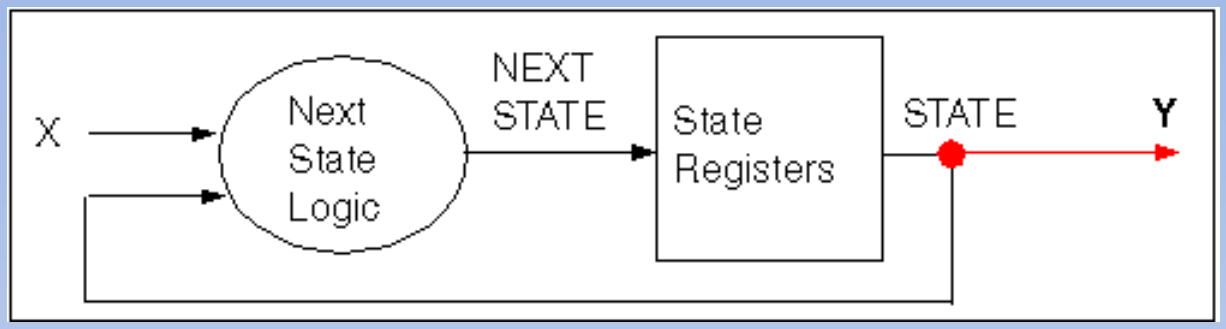




Total logic elements	4 / 114,480 (< 1 %)
Total combinational functions	4 / 114,480 (< 1 %)
Dedicated logic registers	2 / 114,480 (< 1 %)
Total registers	2

AGSTU
Arbete Genom STUdier
Utbildning

FSM: Medvedev



The output vector resembles the state vector

One Process

```
architecture RTL of MEDVEDEV is
   ...
begin

   REG: process (CLK, RESET)
   begin
      -- State Registers Inference with Logic Block
   end process REG ;
   Y <= STATE ;
end RTL ;
```

Two Processes

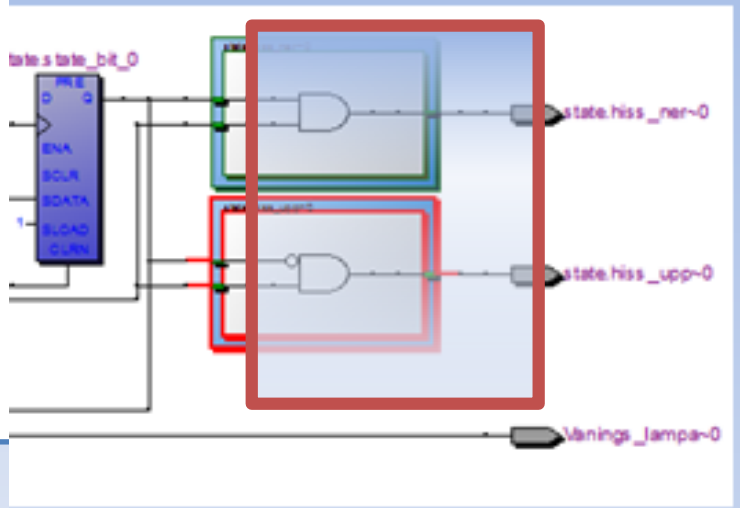
```
architecture RTL of MEDVEDEV is
   ...
begin
   REG: process (CLK, RESET)
   begin
      -- State Registers Inference
   end process REG ;

   CMB: process (X, STATE)
   begin
      -- Next State Logic
   end process CMB ;
   Y <= STATE ;
end RTL ;
```

AGSTU
Arbete Genom STUdier
Utbildning

Hand Coding

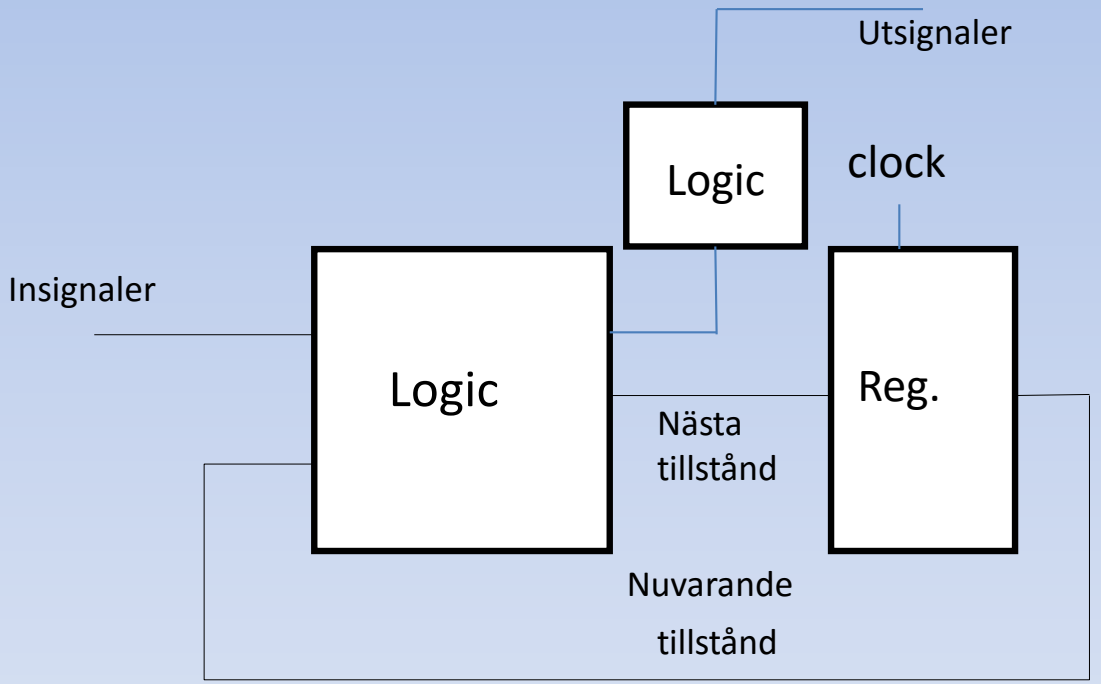
```
subtype STATE_TYPE is std_ulogic_vector (1 downto 0) ;
signal STATE : STATE_TYPE ;

constant Vaning_1   : STATE_TYPE := "00";
constant Vaning_2 : STATE_TYPE := “01";
constant Hiss_upp     : STATE_TYPE := "00";
constant Hiss_ner     : STATE_TYPE := “10";
…
   case STATE is
        when  Vaning_1  => …
        when  Vaning_2  => …
        when Hiss_upp      => …
        when Hiss_ner      => …
   end case ;

…..
Vaning_lampa <=  STATE(0);
Motor_ner <= STATE(1);
Motor_upp <= NOT STATE(1);
```

- Defining constants
- Control of encoding
- Safe FSM
- Portable design
- Disadvantage:
- More effort (especially when design changes)



AGSTU
Arbete Genom STUdier
Utbildning

Mealy tillståndsmaskin

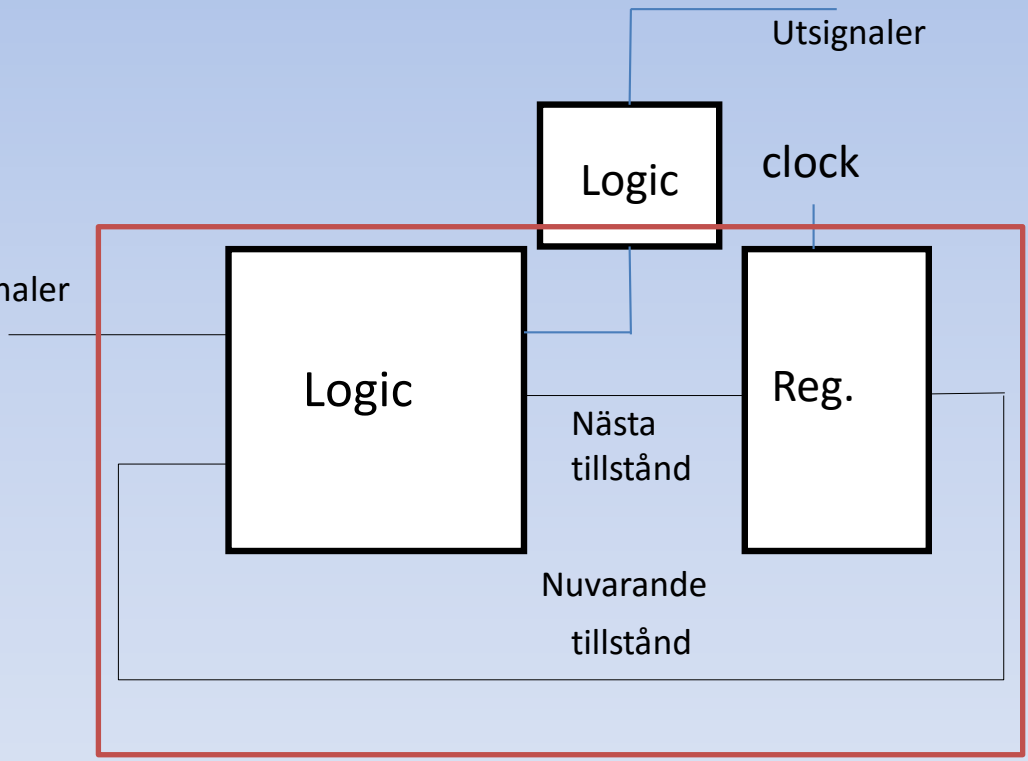


Mealy:
utgångarna är en funktion av nuvarande tillstånd och samtliga ingångar

AGSTU
Arbete Genom STUdier
Utbildning

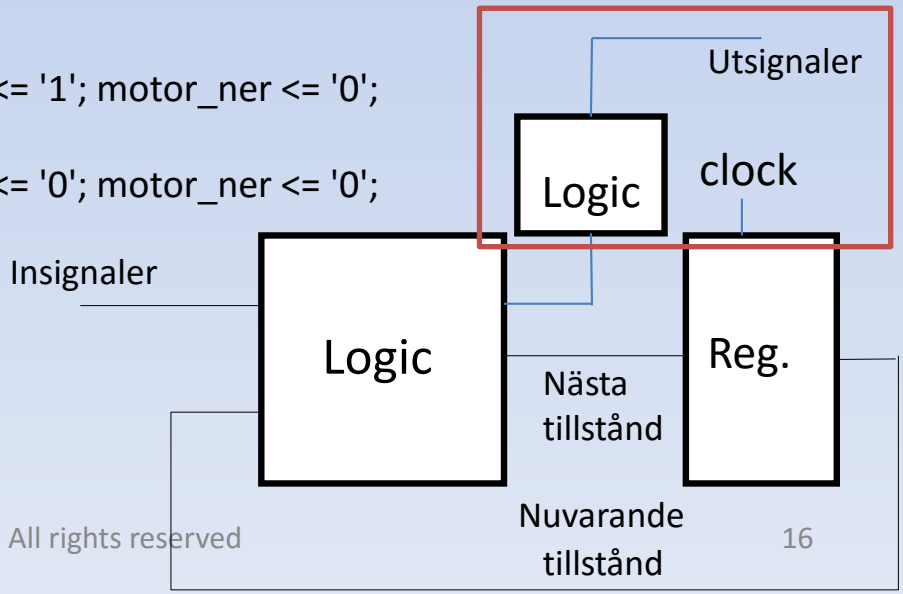
Mealy tillståndsmaskin

```
process (reset_n, CLOCK)
begin
if reset_n = '0' then
                state <= vaning_1;
elsif (rising_edge(CLOCK)) then
                case state is
                when vaning_2=>
                                if KEY_0 = '1' then
                                                state <= hiss_ner;
                                else
                                                state <= vaning_2;
                                end if;
                when hiss_upp =>
                                state <= vaning_2;
                when hiss_ner =>
                                state <= vaning_1;
                when vaning_1 =>
                                if KEY_1 = '1' then
                                                state <= hiss_upp;
                                else
                                                state <= vaning_1;
                                end if;
                when others =>
                                state <= vaning_1;
                end case;
end if;
end process;
```

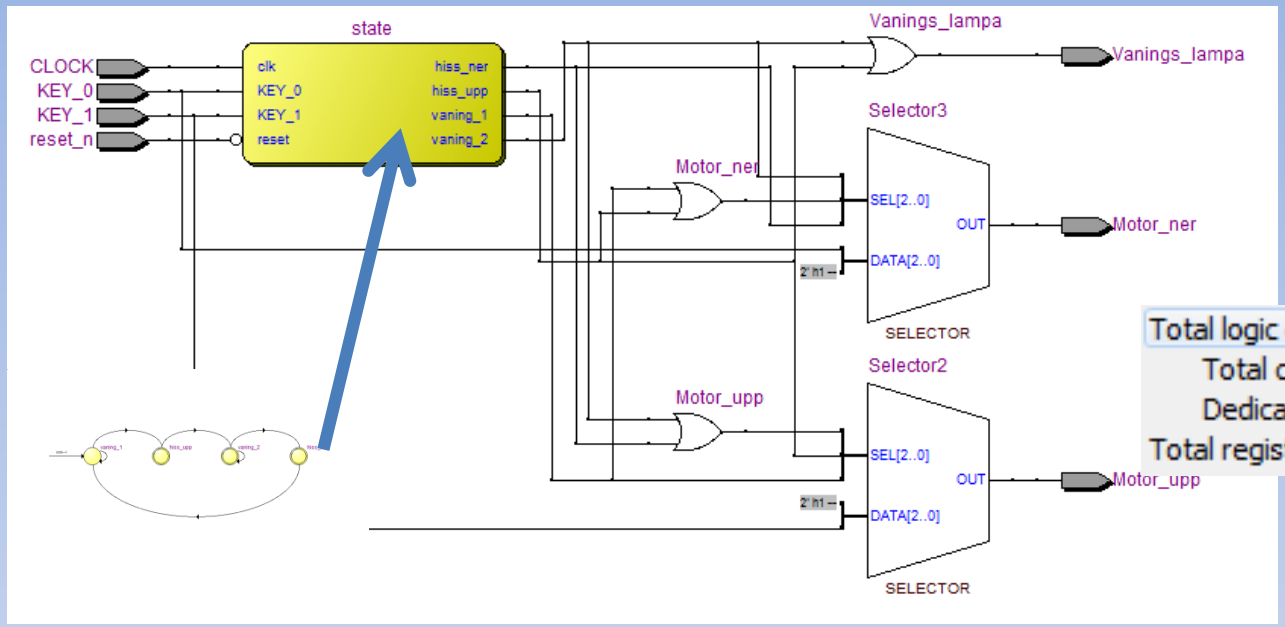


Mealy tillståndsmaskin

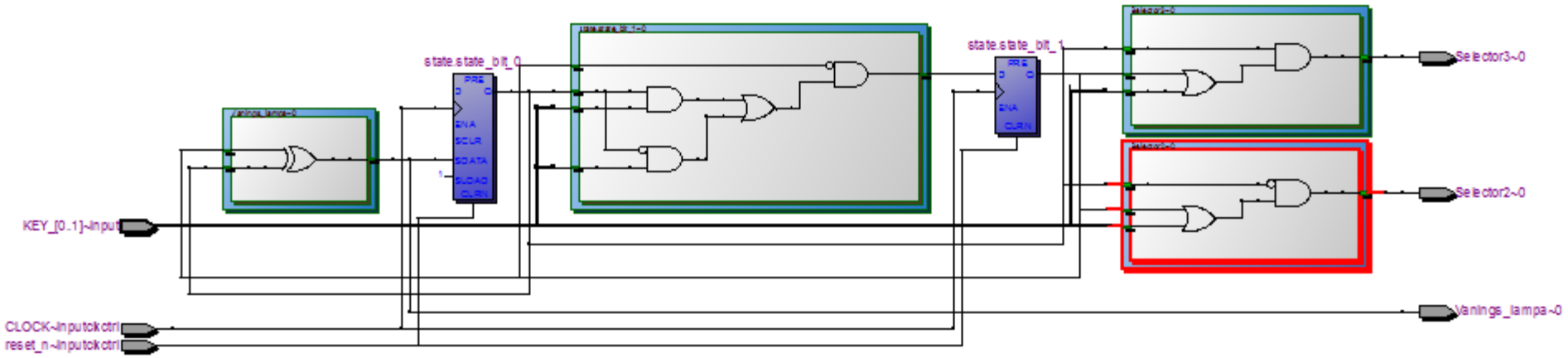
```
case state is
when vaning_2 =>
        if KEY_0 = '1' then
                Vanings_lampa <= '1'; motor_upp <= '0'; motor_ner <= '1';
        else
                Vanings_lampa <= '1'; motor_upp <= '0'; motor_ner <= '0';
        end if;

when hiss_upp =>
        Vanings_lampa <= '1'; motor_upp <= '1'; motor_ner <= '0';

when hiss_ner =>
        Vanings_lampa <= '0'; motor_upp <= '0'; motor_ner <= '1';

when vaning_1 =>
        if KEY_1 = '1' then
                Vanings_lampa <= '0'; motor_upp <= '1'; motor_ner <= '0';
        else
                Vanings_lampa <= '0'; motor_upp <= '0'; motor_ner <= '0';
        end if;

end case;
```



Mealy

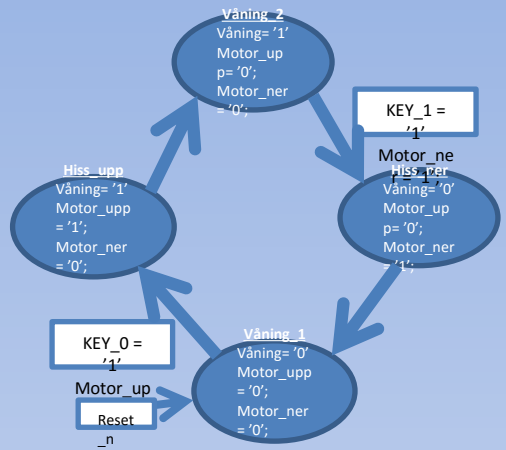


Total logic elements	4 / 114,480 (< 1 %
Total combinational functions	4 / 114,480 (< 1 %
Dedicated logic registers	2 / 114,480 (< 1 %
Total registers	2

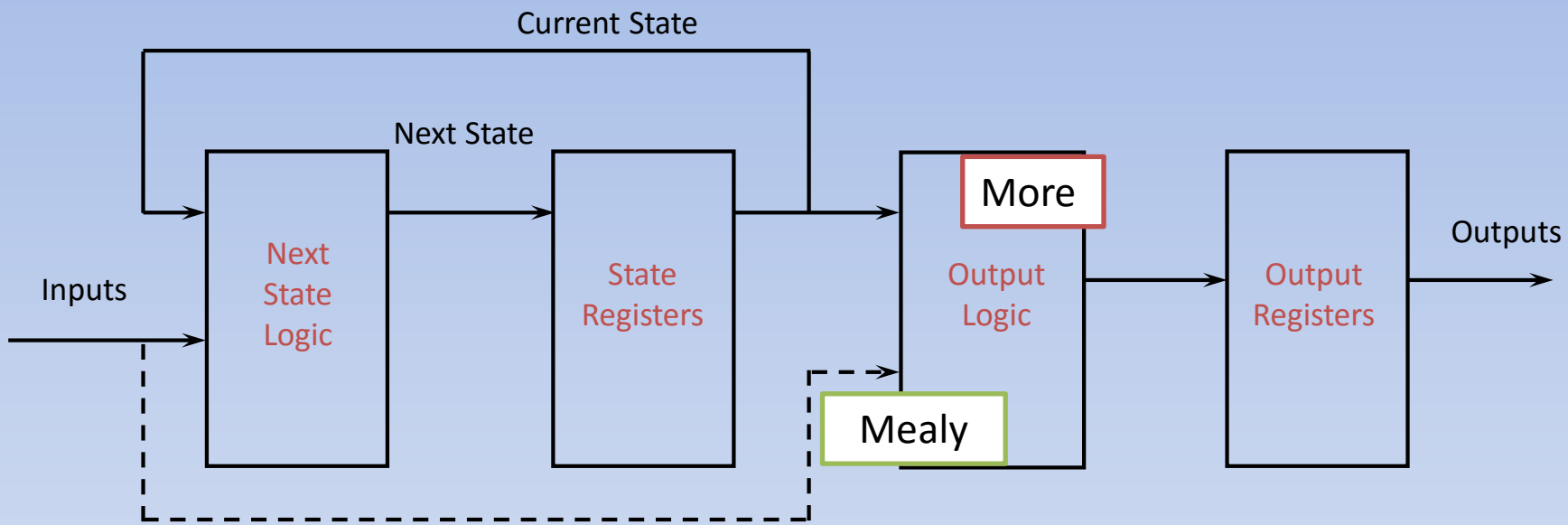


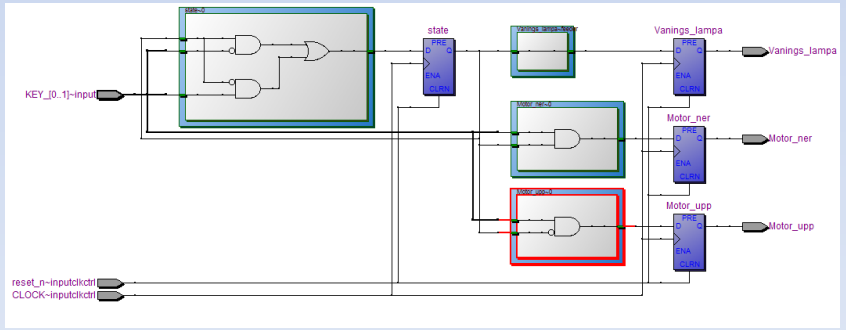
Mealy+Moore+ synchronous outputs

```
process (reset_n, CLOCK)
begin
if reset_n = '0' then
        state <= vaning_1;
        Vanings_lampa <= ’0'; motor_upp <= '0';
elsif (rising_edge(CLOCK)) then
case state is
        when vaning_2=> -- hissen på våning 2
                Vanings_lampa <= '1'; motor_upp <= '0';  --More
                if KEY_0 = '1' then -- knapp intryckt på våning 1
                        motor_ner <= '1'; state <= hiss_ner; -- Mealy
                else
                        motor_ner <= '0'; state <= vaning_2;
                end if;

        when hiss_upp =>
                motor_upp <= '1'; state <= vaning_2;

        when hiss_ner =>
                motor_ner <= '1';state <= vaning_1;

        when vaning_1 => -- hissen på våning 1
```



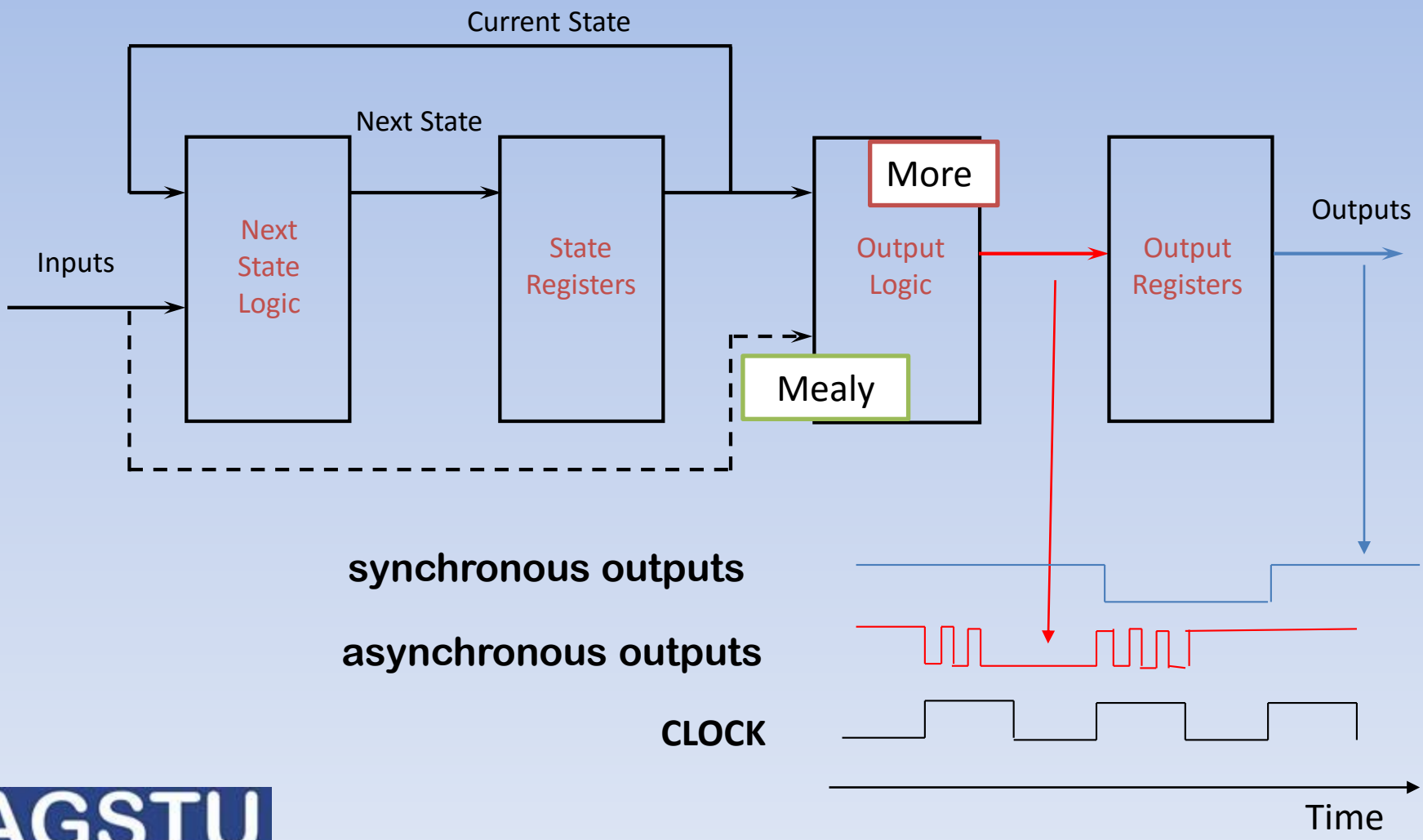
Mealy+Moore+ synchronous outputs





synchronous outputs

- Remove glitches by adding output registers
  - Adds a stage of latency



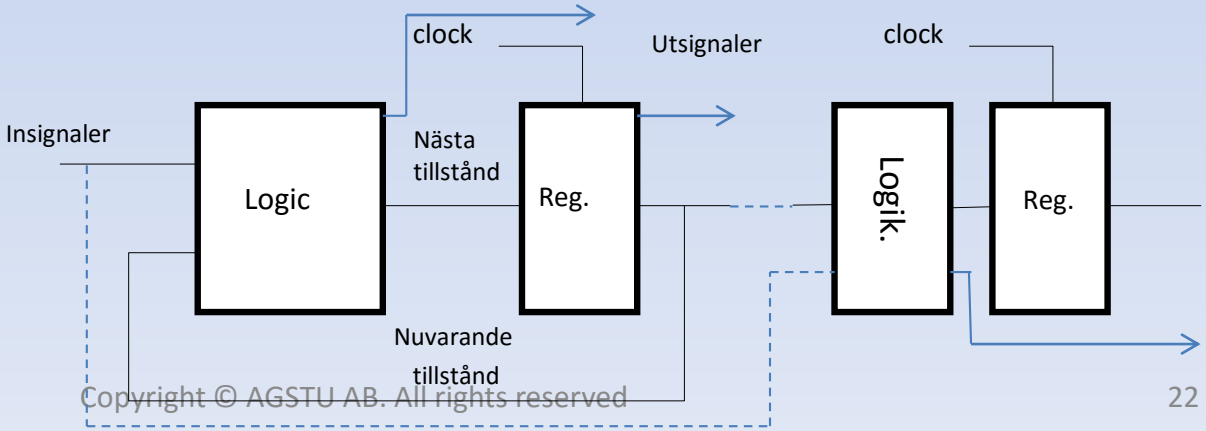
Jämförelse mellan maskinerna

#CLK från ingång till utgång	#0	#1	#2
Mealy	X		
Moore/Medvedev		X	
Med synkron utgång		Mealy	Moore/Medvedev
Mealy+Moore+ Medvedev synchronous outputs		X	X

AGSTU
Arbete Genom STUdier
Utbildning

Vilken tillståndsmaskin ska man välja?

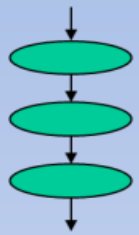
- Utsignal = tillstånd är billigast
- Utvecklingstid? Lätt att förstå? Ändra?
- Men, man kan också blanda typer.
  - Se boken sidan 246.
- I fortsättningen kan vi blanda alla varianter.
- Arbetsplatsen bestämmer oftast kodnings stil.



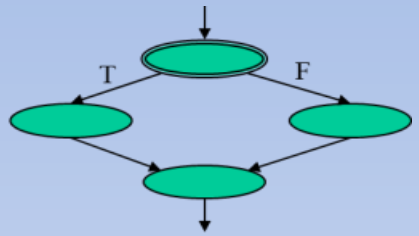
SLUT

AGSTU
Arbete Genom STUdier
Utbildning

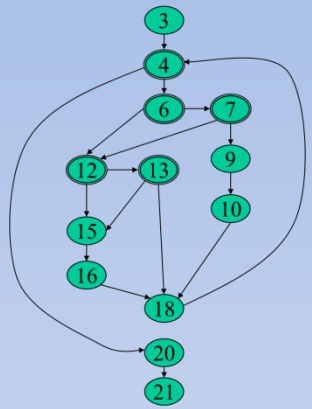
Olika konstruktioner med tillståndsmaskin

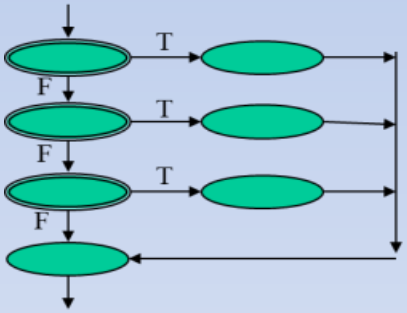


Sequence

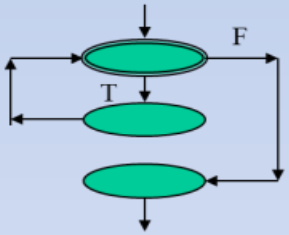


If-then-else





Selection



Repetition