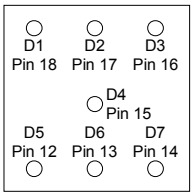
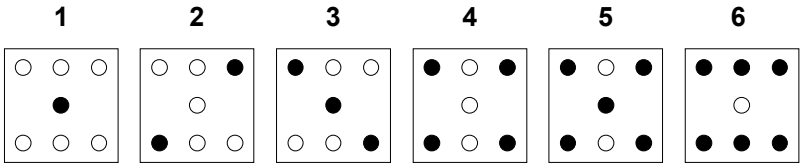
Entwurf des PLD für den Spielwürfel

Anforderungen

Takteingang: Pin 1 (Clock)
Steuereingang: Pin 2: LOW = Würfeln
HIGH = Halt und Anzeige
Ausgänge: Pin 12..19: LED gemäss Plan und Schema

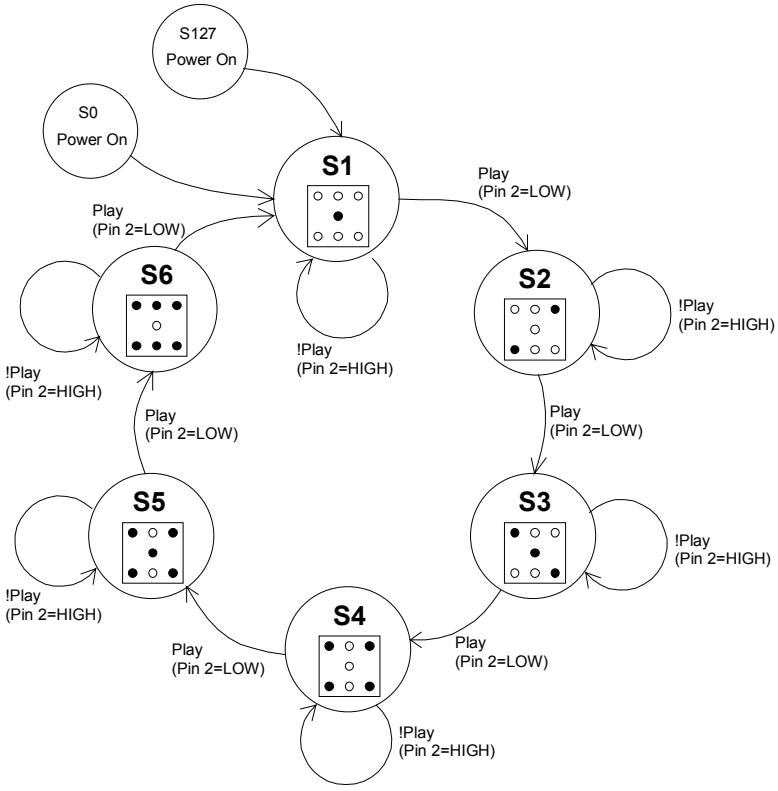


Anzeige der Spielresultate:



Ablaufdiagramm

Realisation als Zähler mit 7 Flip-Flops wobei jede LED an einen FF-Ausgang angeschlossen wird. Daraus entwickelt sich das State-Diagramm:



CUPL Code

In CUPL können wir diese Ablaufsequenz direkt übernehmen. Die States werden symbolisch über ihr Binäräquivalent definiert:

```
Pin 1  = clk ;
Pin 2  = !play ;        /* Inverse Logik! */
Pin 11 = OE;

/** Outputs **/
Pin [18..12]  =  ![Q6..Q0];      /*     D1   D2   D3    */  /* Inverse Logik !! */
                                 /*     18   17   16    */
                                 /*                     */
                                 /*          D4         */
                                 /*          15         */
                                 /*                     */
                                 /*     D5   D6   D7    */
                                 /*     12   13   14    */

/* Definition von lokalen Variablen.  */
field count = [Q6..Q0];          /* 7 FF-Ausganenge Q7..Q0 */
$define S0 'b' 0000000           /* Initial States nach dem Einschalten */
$define S127  'b' 1111111
/* Definition der Zaehlerzustaende (LED Muster) */
$define S1 'b' 0001000
$define S2 'b' 0010001       /* Beachten Sie: $define haben keine Semikolon! */
$define S3 'b' 1001100       /* 1 = LED brennt-> Pin = low mit inverser Logik */
$define S4 'b' 1010101
$define S5 'b' 1011101
$define S6 'b' 1110111

/** Sequenzdefinition **/

sequenced count {                /* Sequenzer mit D-FF */
present S0 next S1;              /* Zustaende 0,127 (Reset) fuehren zu S1 */
present S127 next S1;
present S1
  if play next S2;                /* Play gedrueckt: Weiterzaehlen */
  if !play next S1;               /* Play nicht gedrueckt: Zustand halten */
present S2
  if play next S3;
  if !play next S2;
present S3
   if  play next S4;
   if  !play next S3;
present S4
   if  play next S5;
   if  !play next S4;
present S5
   if play next S6;
   if !play next S5;
present S6
   if  play next S1;
   if  !play next S6;
}
```