Generator für Fibonacci-Zahlen

Entwickeln Sie eine programmierbare FSM, welche Fibonacci-Zahlen generiert und darstellt. Die FSM weise zwei Eingänge In0 und In1 der Bit-Breite 4 auf. Nach Aktivieren des Eingangs Start werden jetzt nacheinander Fibonacci-Zahlen an den beiden 7-Segment-Anzeigen ausgegeben: Zuerst In1, dann In0 + In1, dann In1 + (In0 + In1) etc. [1])
Die ausgegebenen Zahlen sollen im Bereich 0 bis 99 dezimal liegen. Bevor eine Zahl ausgegeben wird, die den Anzeigebereich überschreiten würde, gehe die FSM in einen Endlosloop über, so dass die höchste noch darstellbare Fibonacci-Zahl an der Anzeige stehen bleibt.
Erst nach Reset (bei aktiviertem Start) wird eine neue Zahlenreihe generiert.

Das Blockschaltbild für die Simulation soll nach folgender Struktur aufgebaut sein:

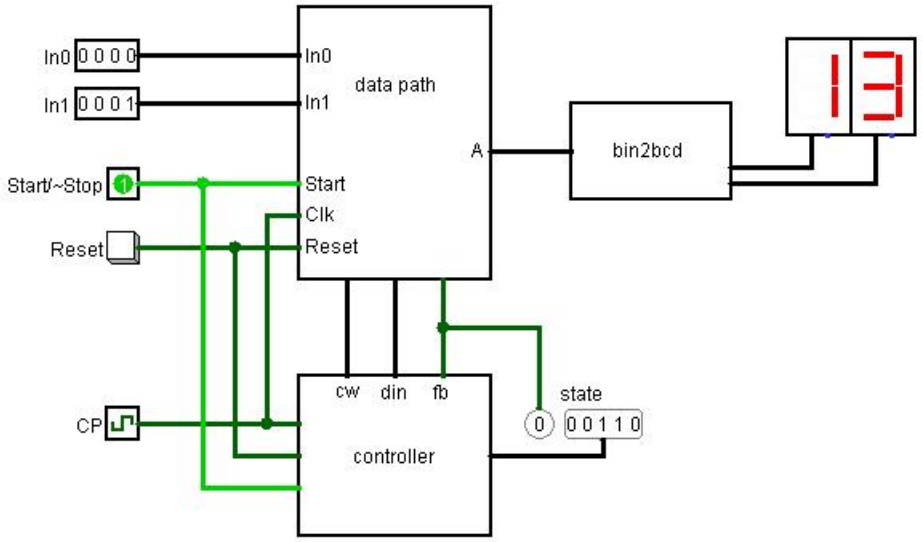

Abb. 1: Blockschaltbild des Fibo-Generators.
Nach Reset wird 0 an der Anzeige dargestellt. Mit In0 = 0 und In1 = 1 werden nacheinander die Werte 1, 2, 3, 5, 8, 13, 21, 34, 55 und 89 zur Anzeige gebracht.

Wichtig:
Die Schaltungen sollen gemäss Vorlesungsteil *"RNO, Hardware von unten" von Michael Hutter und Karl C. Posch, TU Graz (2010)* aufgebaut werden, d.h. die Hardware muss einen Datenpfad (data path) aufweisen und mit einer Kontroll-Logik (controller) ausgestattet sein. Diese Logik liefert den Microcode (control_word) für den Datenpfad. Der Datenpfad bestehe aus einer geeigneten Anzahl Register gewählter Breite, einer ALU sowie einer Ladelogik, um externe Werte entgegenzunehmen.

Die ALU soll so entworfen werden, dass sie die notwendigen Operationen anbietet. Bestandteil der Aufgabe ist, sich zu überlegen, welche Operationen zur Lösung der Aufgabe benötigt werden: Ein Teil des Kontrollwortes wird dann bestimmen, welche Operation zur Ausführung kommt.

[1]) Statt mit geeigneten In0 und In1 eine Folge von Fibonacci-Zahlen zu generieren, können mit andern Initialwerten natürlich andere Folgen erzeugt werden.

Vorgehen:

- Pseudoprogramm entwerfen:
  Inputs, Berechnungen, nötige Variablen und Outputs.
- Anzahl nötige Register und Registerbreite ableiten.
- Pseudoprogramm konkretisieren.
- ALU entwerfen.
- Daten-Input entwerfen.
- Daten-Output entwerfen.
- Datenpfad realisieren (Register, Rückkopplung, ALU, Output, Kontrollwort-Input: es braucht ein Feedback zum Controller? Weshalb?).
- Controller realisieren (Next-State-Logic, Konstanten-ROM, Programm-ROM, Kontrollwort-Output).
- Programm für die Abfolge der Kontrollwörter umsetzen.
- ROM mit Werten versehen.
- Testen mit Logisim.

Beachten Sie die Dateien *ifElseMechanismus.pdf* und *ifElseMechanismus.circ*, welche zeigen, wie eine Verzweigung (if - else) realisiert werden kann.

Die Dateien *fiboGenerator_V0.circ* und *bin2bcdConverter.circ* stellen Ihnen das Gerüst der FSM und die Ausgabelogik zur Verfügung.