Versuch 1, Digitalelektronik

1 Lernziel

Dies ist der Einführungsversuch. Nach der Durchführung sollten Sie mit den einfachsten Funktionen der Entwurfsumgebung und der Basisplatine (Breakout-Board) des Hardwarekits umgehen können.
Sie können danach mit einigen Anweisungen (unbedingte Zuweisung, when/else und with/select) in der nebenläufigen Umgebung einer *architecture* einen Entwurf durchführen und mit Hilfe der Basisplatine testen.

In Abbildung *1* finden Sie die Pins, die den Ein- und Ausgabeelementen zugeordnet sind.

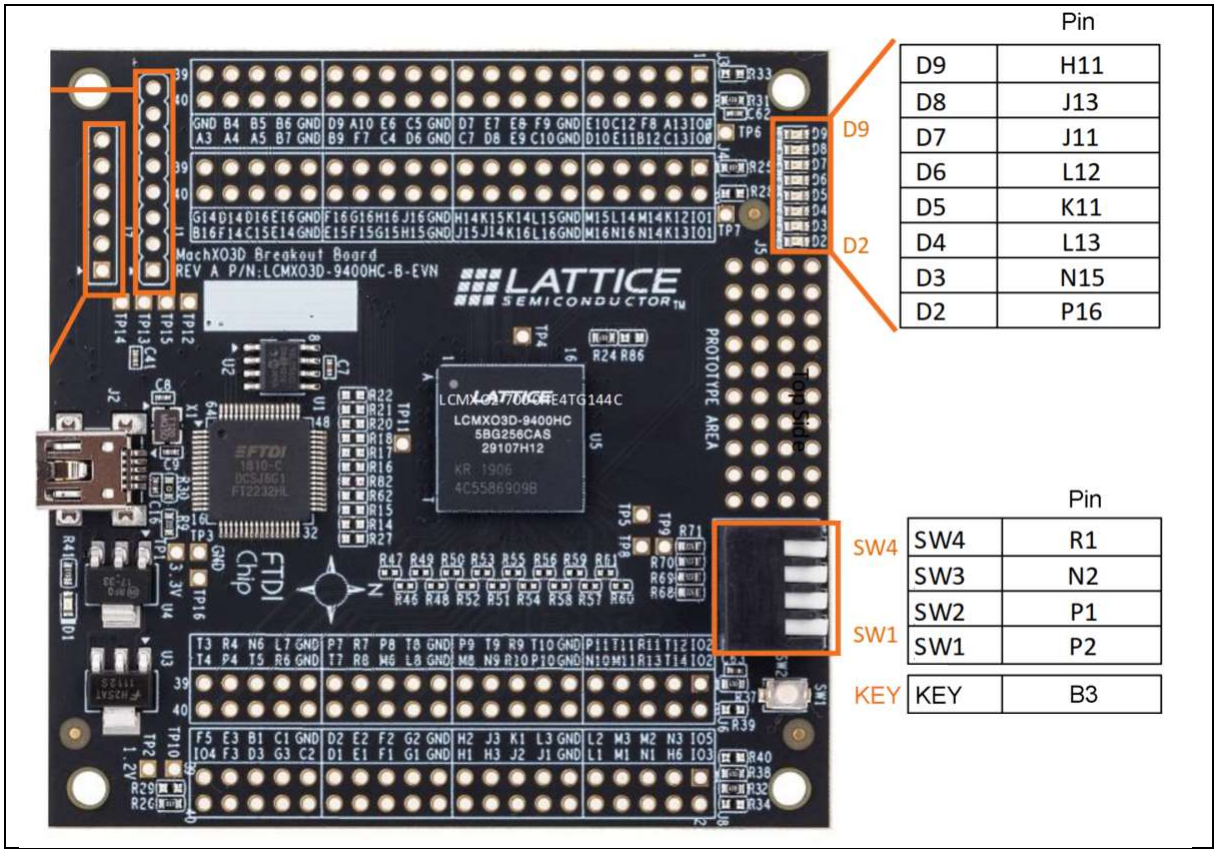

Abbildung 1: Ein-Ausgabeelemente auf dem Breakout-Board

Stellen Sie beim Test im Pin-Editor von Diamond in der Spalte *IO TYPE* immer *LVCMOS33* ein und in der Spalte *PULL MODE NONE*.
Leider müssen Sie das bei jedem Projekt neu machen, da es (offenbar) keinen voreinstellbaren Default gibt.

Für alle Versuche gilt: Benutzen Sie den Typ std_logic.
Geben Sie bei den Aufgaben 2, 3 und 4 jeweils eine VHDL-Datei mit jeweils nur einer entity und einer architecture ab und bei Aufgabe 5 eine mögliche Erklärung.

Es gibt nicht die eine einzige richtige Lösung. Solange keine Vorgaben existieren (wie z.B. verwenden Sie einen Vektor), können sie selber wählen, wie Sie eine Schaltung beschreiben.
Falls Sie eine Anweisung benutzen, die in der Vorlesung (bisher) nicht vorgestellt wurde, dann geben Sie an, warum Sie das getan haben und ggf. eine Quelle (woher Sie diese Anweisung haben). Bei richtiger Anwendung ist das ebenfalls eine zulässige Lösung.

Versuch 1, Digitalelektronik

2 AND/OR/XOR

Zum Test der Platine gibt es eine schon fertig kompilierte Schaltung zum Download (siehe Video). Der VHDL-Quelltext dazu fehlt aber. Damit werden die drei logischen Verknüpfungen and, or und xor auf dem Breakout-Board realisiert.
Führen Sie den Entwurf selbst durch und testen Sie, ob sich die Schaltung so wie die Vorlage verhält.

Achten Sie darauf, dass die nicht benutzten LED wie in der Vorlage ständig ausgeschaltet sind.

3 Multiplexer

Da nur 5 Eingabeelemente zur Verfügung stehen, kann nur ein 2:1-Multiplexer (zwei Eingänge, ein Ausgang) sinnvoll getestet werden. Er hat zwei Betriebsarten.

3.1 **Normalbetrieb** (Taste KEY nicht gedrückt)

Mit Hilfe von SW1 soll entweder SW3 oder SW4 auf die LED D2 durchgeschaltet werden.
Die anderen LED sind ständig ausgeschaltet. Befindet sich SW1 in der Stellung „unten“, wird SW3 durchgeschaltet.
Für SW3 und SW4 gilt (sofern gerade durchgeschaltet): Stellung unten -> LED an.

3.2 **Testbetrieb** (Taste KEY gedrückt)

Mit Hilfe des Tasters KEY soll die LED getestet werden können. Das bedeutet, dass unabhängig von der Stellung der Schalter SW1, SW3 und SW4 die LED D2 leuchtet, wenn der Taster gedrückt wird.
Die übrigen LED sind ständig ausgeschaltet.

4 1-aus-8 Demultiplexer

Der Demultiplexer hat drei Betriebsarten.

4.1 **Normalbetrieb** (Taste KEY nicht gedrückt, SW1 oben)

Mit Hilfe der Schalter SW2, SW3 und SW4 wird genau eine der LED D2-D9 eingeschaltet. Die Zuordnung können Sie frei wählen, muss aber dokumentiert werden.
Schreiben Sie diese Dokumentation in den VHDL-Code. Eine kurze Dokumentation ist besser als eine lange, sofern der Nutzer damit die benötigte Information bekommt. Hier muss man wissen, welche LED bei welcher Eingangskombination (Schalter) leuchtet.

4.2 **Disable** (Taste KEY nicht gedrückt, SW1 unten)

In dieser Betriebsart leuchtet keine der LED.

4.3 **Testbetrieb** (Taste KEY gedrückt)

Solange die Taste gedrückt ist, leuchten alle LED.

Deklarieren Sie in der *entity* den *port* für die LED als Vektor mit acht Elementen. Die übrigen Anschlüsse deklarieren Sie wie in den bisherigen Versuchen als Einzelsignale.

5 Hardware (nicht VHDL)

In dem Demonstrationsbeispiel aus dem Video (Schalter steuert LED) haben sieben der LED ständig leicht geleuchtet. Woran kann das liegen?