**Rechnertechnik** **Übungszettel 7**
**SS2013**

Sie sollen in dieser Übung einen Rechner mit Hilfe von Xilinx-ISE erstellen, der sich an den Rechner-entwurf aus der Vorlesung orientiert.
Der Rechner zeichnet sich durch die folgenden Eigenschaften aus:

- Registerfile mit 4 Registern a 4 Bit, 1 Leseport, 1 Schreibport, r0 ist immer 0 und kann nicht beschrieben werden, 1 Schreibsignal (ist gegeben)
- 1 4-Bit Addierer ohne Flags (kann nur addieren, Subtraktion muss nachgebildet werden) (ist gegeben, Sie dürfen natürlich auch Ihren Addierer aus den vorangegangenen Übungen nutzen)
- 1 4-Bit Inverter
- 3 unterschiedliche Befehle:
    - ADD rx, #const[4-Bit], rz (d.h. rz=rx+#const)
    - SUB rx, #const[4-Bit], rz (ACHTUNG: Der Addierer kann nur addieren!)
    - INVERT rx, rz (d.h. rz=not rx)
- 4-Bit Inkrementierer für PC und/oder sonstige Verwendung (ist gegeben)

Die einzelnen Bauteile, sowie weitere Teile (Multiplexer, Register mit und ohne Reset und Schreibsignal, diverse PROMs) stehen als Bauteilbibliothek zur Verfügung; Dokumentationen dazu finden Sie auf Moodle. **Sie dürfen allerdings nur <u>einen</u> Addierer und <u>einen</u> Inverter in Ihrem Datenpfad einsetzen!**

## <u>Aufgabe 1: Rechner-Entwurf (ohne Pipelining)</u>

Konzipieren Sie (am besten auf Papier) einen Datenpfad für Ihren Rechner. Überlegen Sie dabei insbesondere, wie Sie die Subtraktion mit den zur Verfügung stehenden Bausteinen realisieren. Als Befehlsspeicher verwenden Sie bitte einen PROM-Baustein, der keinen Takt benötigt, sondern das Ergebnis sofort liefert.
Identifizieren Sie anschließend für jeden Befehl die einzelnen Zyklen seines Ablaufes und machen Sie sich anhand Ihrer Datenpfad-Skizze deutlich, welche Teile des Datenpfades in welchem Zyklus des Befehls benutzt werden.
Konzipieren Sie dann die Steuersignale des Datenpfades und notieren Sie für jeden Zyklus jedes Befehls die Belegung der Steuersignale tabellarisch.
Legen Sie das Befehlsformat für die drei Befehle so fest, wie Sie es im Befehlsregister vorfinden möchten (es sollte so festgelegt werden, dass die Ansteuerung möglichst einfach ist, vgl. auch Kurz-Doku zum Programm *genasm*).
Entwerfen Sie abschließend den Zustandsautomaten, der die einzelnen Zyklen der Befehle abarbeitet, erarbeiten Sie die Zustandsübergangstabelle und integrieren Sie für jeden Zustand die Belegung der Steuersignale aus der tabellarischen Auflistung. Kodieren Sie den Automaten so, dass Sie ein PROM zur Implementierung nutzen können. Alternativ können Sie hier auch qfsm für die Erzeugung des Automaten nutzen.

## Aufgabe 2: Rechner-Implementierung (ohne Pipelining)

Übertragen Sie den in Aufgabe 1 erstellten Datenpfad und das dort erstellte Steuerwerk nach Xilinx-ISE und testen Sie Ihren Entwurf. Sie können eine vordefinierte Baustein-Bibliothek verwenden, die alle wesentlichen Datenpfad-Elemente enthält. Diese Elemente finden Sie in einem zipFile auf Moodle, welches auch das Programm *genasm* enthält. Entpacken Sie bitte diese Datei in Ihrem work-Verzeichnis. Dokus zu *genasm* und der Bauteilbibliothek finden Sie ebenfalls auf Moodle. Sollten Sie bei den Speicherbausteinen (PROM) weniger Bits benötigen als die Bauteile anbieten, lassen Sie die überflüssigen Bits einfach unbenutzt (Achten Sie darauf, dass ungenutzte Adress-Eingangsbits an den PROM-Bausteinen eine vernünftige Belegung haben!).
Testen Sie Ihren Datenpfad bitte mit dem folgenden Assemblerprogramm:

```
ADD r0, #0b1011, r1  ; r1=0+11
INVERT r1,r1         ; invertieren, müsste also 0b0100=4 rauskommen
SUB r1, #3, r2       ; r2=r1-3 = 4-3 = 1
ADD r2, #0, r3       ; r3=r2+0 = 1+0=1
; Ende des Tests
```

Damit Sie die Tests komfortabel erstellen können, nutzen Sie bitte das Programm *genasm*. Zur Benutzung müssen Sie zunächst eine Konfigurationsdatei erstellen, in der Ihre Belegung des Befehlsregisters und die einzelnen Befehle beschrieben sind. *genasm* kann dann obiges Programm in eine Textdatei übersetzen, die Sie direkt in Ihr verwendetes PROM bei der Simulation einlesen können (vgl. Doku zu *genasm*).