RA FS 14 Serie 5

Thoma Papadhimitri, Urs Gerber, Delio Vicini

Die fünfte Serie ist bis Dienstag, den 13. Mai 2014 um 13:00 Uhr zu lösen und in schriftlicher Form in der Übungsstunde abzugeben. Für Fragen steht im ILIAS jederzeit ein Forum zur Verfügung. Allfällige unlösbare Probleme sind uns so früh wie möglich mitzuteilen, wir werden gerne helfen.
Viel Spass!

Theorieteil

Gesamtpunktzahl: 7 Punkte

1 Static Branch Structure (3 Punkte)

Welche Probleme treten bei folgendem Assembler-Loop auf, wenn standardmässig *Predict Not Taken* gewählt wird.

```
1       addi $s1, $zero, 1024 // s1 := 1024
2 loop: addi $s1, $s1, -1 // s1--
3       jal subroutine // call subroutine()
4       bne $s1, $zero, loop // if (s1 != 0) jump loop
```

Optimieren Sie das Codebeispiel auf *Predict not Taken*.

2 Antidependencies (1 Punkt)

Erklären Sie was eine Antidependency ist.

3 Branch-Delay-Slots (1 Punkt)

Erklären Sie was ein Branch-Delay-Slot ist.

4 Dynamic Branch Structure (2 Punkte)

Welche Dynamic Branch Prediction (1-Bit oder 2-Bit) schliesst bei folgendem Codestück besser ab? Begründen Sie Ihre Antwort.

```
1           addi $s2, $zero, 0 // s2 := 0
2 outerLoop: addi $s1, $zero, 0 // s1 := 0
3 innerLoop: bne $s1, $s2, outerLoop // if (s1 != s2) jump outerLoop
4           addi $s1, $zero, 1 // s1 := 1
5           j innerLoop // jump to the inner loop
```

Programmierteil

1D-Pong

Es geht in dieser Serie darum, ein einfaches Spiel für zwei Spieler zu implementieren. Spieler L spielt mit der Taste `KEY3`, Spieler R mit der Taste `KEY1`. Das Spiel ist im Wesentlichen in folgende Teil gegliedert:

Startphase Zum Entscheiden der Laufrichtung.

Spielphase Das leuchtende LED (der "Ball") wandert in eine Richtung

Richtungswechsel erfolgt bei `LEDR0` oder `LEDR9`, wobei der jeweilige Spieler in diesem Moment (innerhalb eines bestimmten Zeitraums) die Taste `KEY1` bzw. `KEY3` zu drücken hat um die Richtung umzukehren.

Ball verpasst Wartet der Spieler zu lange so verliert dieser das Spiel und der andere Spieler erhält einen Punkt, die Punktzahl wird jeweils auf `HEX0` und `HEX3` angezeigt. Anschliessend beginnt die Spielphase wieder (Lauflicht startet in der Mitte).

Ball getroffen Hat der Spieler die Taste im richtigen Moment gedrückt, so wird die Laufrichtung geändert und die Geschwindigkeit etwas erhöht ("Levelwechsel").

Details

Zu Beginn sollen abwechslungsweise die `LEDR4` und `LEDR5` leuchten. Wenn beide Spieler gleichzeitig ihre Tasten drücken geht das Spiel los, sobald die Tasten wieder losgelassen werden. Das Licht wandert von links nach rechts (oder von rechts nach links, je nachdem ob zuletzt `LEDR4` oder `LEDR5` geleuchtet hat).

Wenn `LEDR0` (für Spieler R) bzw. `LEDR8` (für Spieler L) leuchtet, muss der Spieler seine Taste drücken. Das Licht wandert dann wieder in die andere Richtung. Drückt ein Spieler zu früh oder zu spät, verliert er das Spiel. Mit jedem Richtungswechsel wird das Licht ein wenig schneller.

Wenn ein Spieler das Spiel verliert, sollen die 4 LEDs auf der Seite seines Gegners (dem Sieger) blinken. Dem Gegner wird ein Punkt gutgeschrieben, der aktuelle Punktestand wird auf dem Hex-Display angezeigt. Das Spiel startet (mit der aktuellen Geschwindigkeit) von der Mitte aus. Zum Bestimmen der Laufrichtung kann z.B. die aktuelle Zeit als Zufallsvariable verwendet werden.

Als Option kann noch `KEY2` zum Neustart des Spiels verwendet werden.

Sie dürfen das Spiel mit eigenen Ideen erweitern, hier empfiehlt es sich allenfalls sogar den "DE1 Media Computer"[1] anstatt des "DE1 Basic Computers" zu verwenden. Bitte denken Sie daran, Ihre Erweiterungen ausführlich zu dokumentieren.

Aufgaben

(a) Ihre Aufgabe ist es, das Lauflicht der Serie 4 zu einem 1D-Pong (wie oben beschrieben) zu erweitern.

(b) Stellen Sie sicher, dass das Assemblerprogramm *ausführlich und sinnvoll* kommentiert ist. Als Richtwert gilt, dass jede Zeile kommentiert werden soll, man kann zwei Zeilen zusammenfassen, falls dies Sinn macht.

Dies ist eine notwendige Voraussetzung, damit der Programmierteil als erfüllt gilt.

(c) Stellen Sie sicher, dass Ihre Implementation ohne Fehler und Warnungen kompilierbar ist.

Dies ist eine notwendige Voraussetzung, damit der Programmierteil als erfüllt gilt.

(d) Erstellen Sie aus Ihrer Lösung eine Zip-Datei namens `<nachname>.zip` (wobei `<nachname>` natürlich durch Ihren Nachnamen zu ersetzen ist).

(e) Geben Sie die Datei elektronisch durch Hochladen in Ilias ab.

(f) Drucken Sie *zusätzlich* sämtliche für die Lösung relevanten Dateien aus und geben Sie diese in der Übungsstunde zusammen mit der restlichen Serie 5 ab.

[1] `http://www.altera.com/education/univ/materials/comp_org/examples/unv-examples.html`

(g) Denken Sie daran, dass die Altera Boards am Dienstag, den 13. Mai 2014 zurückgegeben werden. Bringen Sie die Boards bitte in die Übungsstunde, Sie erhalten dann ebenfalls die Kaution in der Höhe von 100 Franken zurück. Sollte es Ihnen nicht möglich sein, an der Rückgabe der Boards teilzunehmen, setzen Sie sich bitte frühzeitig mit den Assistenten in Verbindung.