FH Bielefeld
University of
Applied Sciences

SoSe 2018
Prof. M. König

Abgabe:10.06.2018

Praktikum 6 (10 Punkte) – Assembler

Die Abgabe der Lösungen (Beantwortung der Fragen und Quellcode) erfolgt im Ilias bis zum angegebenen Abgabezeitpunkt. Die Besprechung und Bewertung der **eingereichten** Lösungen erfolgt am folgenden Praktikumstermin.

Praktikum - Vorbereitung

In der Vorlesung wurden ARM-Assembler und Inline-Assembler für den ARM-GCC besprochen. Lesen Sie hierzu für weitere Informationen die Webseite http://www.ethernut.de/en/documents/arm-inline-asm.html und http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0553a/CIHJJEIH.html (Cortex-M4 Devices Generic User Guide, Kapitel 3 „The Cortex-M4 Instruction Set“ im Ilias).

Praktikum

Aufgabe 1 *(4 Punkte)*

Ergänzen Sie das Listing 6.1 mit Assembler-Befehlen in dem angegebenen Bereich, so dass Ihr ergänzter Programmcode die Nummer `number` verändert. Die Nummer muss bei jedem Durchlauf verdoppelt werden, bis sie den Wert 128 erreicht hat. Danach muss die Nummer wieder auf den Wert 1 gesetzt werden und die Verdoppelung von vorne beginnen.

```
// our working number
volatile uint8_t number = 1;

// baudrate for serial communication
const int baudRate = 9600;

// initialize Serial
void setup() {
  Serial.begin(baudRate);
}

// main loop
void loop() {
  // print number
  Serial.println(number);

  asm volatile(

  //TODO: insert program here

  );
}
```

Listing 6.1: Rahmen für Aufgabe 1

Aufgabe 2 *(5 + 1 Punkte)*

Ergänzen Sie das Listing 6.2 mit Assembler-Befehlen in dem angegebenen Bereich, so dass das Array `fibData` mit den ersten dreizehn Fibonacci-Zahlen gefüllt wird. Dabei ist die dritte bis dreizehnte Fibonacci-Zahl jeweils aus ihren beiden vorhergehenden zu berechnen. Hinweis: Eine mögliche Wissenslücke bezüglich Fibonacci-Zahlen könnte Wikipedia füllen (http://de.wikipedia.org/wiki/Fibonacci-Folge). Beachten Sie, dass nicht alle Register benutzt werden können und verwenden Sie z.B. die Register R4 bis R7.

```
// our working data
const uint8_t lastFiboIndex = 13;
uint8_t fibData[lastFiboIndex];
const int waitTime = 1000;

// baudrate for serial communication
const int baudRate = 9600;

void setup() {
  // initialize Serial
  Serial.begin(baudRate);

  // init first two Fibonacci numbers
  fibData[0] = 1;
  fibData[1] = 1;

  asm volatile(

  // TODO: insert program here

  );

}

// main loop
void loop() {
  // print data
  for (int i = 0; i < lastFiboIndex; i++)
  {
    Serial.println(fibData[i]);
  }
  // delay 1s
  delay(waitTime);
}
```

Listing 6.2: Rahmen für Aufgabe 2

Bestimmen Sie für den Assembler-Code beider Programme die Anzahl der Taktzyklen, die diese benötigen.

Überlegen Sie Sich weiterhin, wie Sie sicherstellen können, dass durch Ihren Programmcode veränderte Register nach der Durchführung ihre ursprünglichen Werte bekommen.