Embedded Systems: Aufgabenblatt 6

FH Bielefeld
University of
Applied Sciences

SoSe 2015
Prof. M. König

Abgabe: KW 21 2015

Aufgabe 6 (10 Punkte) - Assembler

Praktikum - Vorbereitung

In der Vorlesung wurde die AVR-Assembler und Inline-Assembler für AVR-GCC besprochen. Lesen Sie hierzu für weitere Informationen die Webseite http://www.rn-wissen.de/index.php/Inline-Assembler_in_avr-gcc.

Praktikum

Aufgabe 6.1 *(2+2 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.

Schreiben Sie zwei Versionen mit Inline-Assember. Die erste Version muss mittels Platzhaltern (beginnend mit `%`) für den “Datenaustausch” (Input/Output) zwischen C und Assembler arbeiten, die zweite darf keine Platzhalter verwenden. Bestimmen Sie ferner für den Assembler-Code beider Programme die Anzahl Taktzyklen, die diese benötigen. Hinweis: Arbeiten Sie mit Register größer `r15`.

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

// initialize Serial
void setup() {
  Serial.begin(9600);
}

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

  asm volatile(

  //TODO: insert program here

  );
}
```

Listing 5.1: Rahmen für Aufgabe 6.1

Aufgabe 6.2 *(3+3 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 sind 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).

```
// our working data
uint8_t fibData[13];

void setup() {
  // initialize Serial
  Serial.begin(9600);

  // 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 < 13; i++)
  {
    Serial.println(fibData[i]);
  }
  // delay 1s
  delay(1000);
}
```

Listing 6.2: Rahmen für Aufgabe 6.2

Wie bei Aufgabe 6.1: Schreiben Sie zwei Versionen mit Inline-Assember. Die erste Version muss einen Platzhalter (beginnend mit `%`) verwenden, die zweite darf keine Platzhalter verwenden. Bestimmen Sie ferner für den Assembler-Code beider Programme die Anzahl Taktzyklen, die diese benötigen. Hinweis: Arbeiten Sie mit Register größer `r15`.

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