LEHRSTUHL FÜR INFORMATIK III
RWTH Aachen
Univ.-Prof. Dr.-Ing. Manfred Nagl,
Christian Fuß, Ulrike Ranger

Die SWT-Programmiersprache Ada 95
Übung zur Vorlesung
SS 2005
02.06.2005

Übungsblatt 04

Allgemeines

Mit diesem Übungsblatt soll der CAN-Treiber aus Übungsblatt 2 weiterentwickelt werden. Der CAN-Treiber nutzt einen CAN-Controller Hardwarebaustein, der folgende Schnittstelle hat. Die Datei **can_controller.ads** befindet sich mit Vorlagen für die anderen Dateien im ZIP-Archiv, das zu dieser Übung runtergeladen werden kann.

```
PACKAGE can_controller IS

  SUBTYPE bit IS integer RANGE 0..1;

  -- CAN message in sequential bitform:
  --      1 bit : transmitted - set to 1 by controller on transmission
  --     16 bits: id_raw (0 <= id_raw <= 65535)
  --      4 bits: dlc_raw (0 <= dlc_raw <= 8)
  --   8*16 bits: data_fld (0 <= data_fld(0..7) <= 65535)
  TYPE bit_msg IS ARRAY (0 .. 148) OF bit;

  registers : ARRAY (0 .. 3) OF bit_msg;

END can_controller;
```

Aufgabe 10 Message-Queue (2+4+1 Punkte)

Erweitern Sie den CAN-Treiber um eine Message-Queue, in der Nachrichten zwischengespeichert werden können, falls alle Register des Controllers belegt sind. Diese Queue soll nicht an der öffentlichen Schnittstelle zu sehen sein.

a) Definieren Sie dazu einen Typ **`msg_queue_struct`**, der die Daten in einem Feld hält. Definieren Sie für das Datenfeld einen Feldtypen mit unspezifizierten Feldgrenzen.

b) Implementieren Sie die Zugriffsmethoden **`enqueue_msg`** und **`dequeue_msg`**, zum Schreiben und Lesen aus der Queue. Falls die Queue voll oder leer ist, sollen Sie die Ausnahmen **`queue_full_exception`** bzw. **`queue_empty_exception`** erwecken.

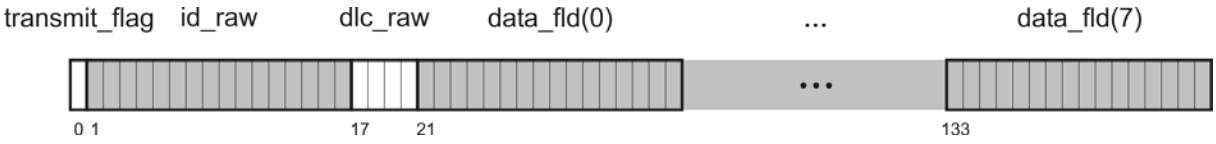
c) Worin unterscheidet sich eine **`raise`**-Anweisung mit zugehörigem Ausnahmebehandler von einem Sprung zu einem Programmstück, das den gleichen Ausnahmebehandlungscode enthält?

Aufgabe 11 Leeres Register finden (2 Punkte)

Implementieren Sie eine Methode **`get_register`**, die ein leeres Register im Controller ermittelt und dessen Index zurückgibt.

Aufgabe 12 Nachricht in Bitfeld kodieren (4 Punkte)

Implementieren Sie eine Methode **`encode_msg`**, die eine Nachricht vom Typ **`can_msg_transmit_struct`** in ein Bitfeld vom Typ **`bit_msg`** kodiert. Eine **`bit_msg`** ist dabei wie folgt aufgebaut.



Aufgabe 13 Nachricht übertragen (4 Punkte)

Implementieren Sie die Methode **`transmit`** so aus, dass sie ein leeres Register ermittelt, die übergebene Nachricht in ein Bitfeld kodiert und dieses in das ermittelte Register schreibt. Falls alles gut geht, geben Sie den Wert **`k_can_tx_ok`** zurück.

Behandeln Sie auch die Ausnahmen, die dabei auftreten können. Bei vollen Registern soll die Nachricht in die Queue gestellt werden und es wird **`k_can_tx_ok`** zurückgegeben. Nur wenn die Queue auch voll ist, wird **`k_can_tx_failed`** zurückgegeben.

***Abgabe**: Do 09.06.2005*

Sie können Ihre Lösung zum obigen Termin in elektronischer Form per eMail an ada@i3.informatik.rwth-aachen.de abgeben.

Bitte vermerken Sie in jedem Fall die Namen und Matrikelnummern aller beteiligten Personen (maximal 3), beim Versenden per eMail auch alle eMail-Adressen. Abgaben in elektronischer Form können ausschließlich in den Dateiformaten *Plain-Text oder PDF* erfolgen.