Einarbeitung/Speicher

1 SensorTag

Machen Sie sich mit dem SensorTag vertraut. Laden Sie sich dafür die SensorTag-App für ihr Smartphone herunter. Welche Sensoren stehen Ihnen zu Verfügung und welche Werte liefern diese?

2 Funktionen und Speicherverbrauch

Erstellen Sie auf dem Laborrechner und dem Raspberry Pi ein Programm, welches die in Abbildung 1 dargestellten Klassen implementiert. Nutzen Sie Dafür wahlweise die Code::Blocks-IDE oder ein Makefile-Projekt. Die Klassen SensorConfiguration und SensorCommunication simulieren eine Kommunikation mit dem SensorTag und werden in den späteren Terminen durch eine echte Implementierung ersetzt.

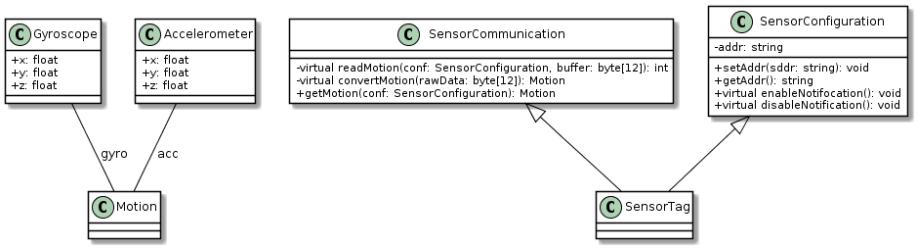

Figure 1: Klassendiagramm

1. Erzeugen Sie eine Instanz der Klasse SensorConfiguration und implementieren Sie zunächst nur die Methoden *getAddr* und *setAddr*. Geben Sie den Speicherbedarf des Objektes aus. Implementiern sie nun die virtuelle Methode *enableNotification*. Geben Sie erneut den Speicherbedarf des Objekts aus. Schauen Sie in dem Executable mittels nm, welche Symbole aus dem C++ Programm Sie dort wiederfinden. Fügen Sie nun auch die virtuelle Methode *disableNotification* hinzu. Ändert sich etwas am Speicherbedarf pro Objekt?

2. Erzeugen Sie nun auch eine Instanz der Klasse SensorCommunication. Der reale Sensor wird später die Werte des Accelerometers und Gyroscopes als Rohdaten in einem 12-Byte-Array liefern. Simulieren Sie das Empfangen der Rohdaten durch die Methode *readMotion* und implementieren sie mit der Methode *convertMotion* einen Parser, der die Rohdaten umrechnet und die jeweiligen Strukturen *Accelerometer* und *Gyroscope* füllt. Das Format der Rohdaten ist wie folgt:

   ```
   GyroX[0:7], GyroX[8:15], GyroY[0:7], GyroY[8:15],
   GyroZ[0:7], GyroZ[8:15], AccX[0:7], AccX[8:15],
   AccY[0:7], AccY[8:15], AccZ[0:7], AccZ[8:15]

   Bit-Breite: 16
   Gyro range: -250 bis +250
   Acc range: -2 bis +2

   Testdaten: 0d 01 6c fc d9 fc f4 fd 85 00 d2 0f
   Erwartetes Ergebnis:
   gyro: 2.05344, -6.99237, -6.16031
   acc: -0.0319824, 0.00811768, 0.247192
   ```

   Für die Umrechnung finden sie hilfreiche Beispiele im User's Guide [1], achten sie außerdem auf die Endianness ihres Systems. Die Methode *getMotion* soll die Struktur *Motion* mit den umgerechneten Werten füllen und zurückgeben.

3. Implementieren Sie die Klasse SensorTag, die von SensorConfiguration und SensorCommunication erbt. Welcher Speicherbedarf entsteht pro Objekt? Wie sehen die Symbole im Executable aus?

References

[1] TI SencorTag CC2650 User's Guide, Abschnitt 5.2
`http://processors.wiki.ti.com/index.php/CC2650_SensorTag_User's_Guide`