Übung Sensornetze – (für 18. November 2004)

Vorlesung 1: Motivation

Aufgabe 1.1: Abschätzung der Lebenszeit eines Knotens

Folgende Daten seien für einen Knoten gegeben:

Grundverbrauch im Sleep-Modus:	50 uA = 0,05mA [1u = 10^(-6)]
Grundverbrauch während Prozessor aktiv ist (z. B. für Berechnungen):	8mA
Zusätzlicher Verbrauch für das Senden:	10mA
Zusätzlicher Verbrauch für das Empfangen:	6mA

Die Batterie liefert 1800 mAh
Pro Tag verliere die Batterie 8% ihrer noch verfügbaren Ladung (zur Vereinfachung erfolgt der Verlust nur 1x pro Tag zu genau einem Zeitpunkt, also nicht kontinuierlich).

Wie lange kann ein Knoten betrieben werden, wenn er alle 200 ms eine Messung aufnehmen soll und diese 1x pro Sekunde versendet? Man nehme pro Sendevorgang nur einen Empfangsvorgang an.

Dabei müssen pro Sendevorgang 200 Bytes an Daten verschickt werden. Die Funkverbindung ermöglicht eine Datenrate von 9600 Bits/s. Der Mess- und Verarbeitungsvorgang dauere je 5ms.

(1) Wie lange kann der Knoten betrieben werden?

(2) Auf welche Zeit verkürzt sich die Betriebszeit, wenn ein Knoten nicht weiß wann ein Paket eintrifft und daher ständig den Kanal abhören muss?

Lösung:

(siehe nächste Seite)

Übung Sensornetze – (für 18. November 2004)

Vorlesung 1: Motivation

Aufgabe 1.1: Lösung

Energie für Messung und Verarbeitung

5 Messungen/Sekunde x 0,005 Sekunden x 8mA = 0,2mAs

Energie für Datenübertragung (Senden und Empfangen)

(200 Bytes x 8 Bit)/(9600 Bits/s) x (8mA Grundverbrauch + 10 mA Sendeverbrauch) +
(200 Bytes x 8 Bit)/(9600 Bits/s) x (8mA Grundverbrauch + 6mA Empfangsverbrauch) =
2/12s x18 + 2/12x14mA = 5+1/3 mAs

Energieverbrauch Idle-Zeit

Aktive Zeit: 0,025s Messungen u. Verarbeitung
0,333s Datenübertragung

Idle Zeit d.h. (1-0,025-0,333) = 0,641. Verbrauch: 0,641s x 0,05 mA = 0,03208 mAs

Energie pro Sekunde

0,2mAs + 5,33 mAs + 0,03208 mAs = ca. 5,56208 mAs

Batterie hat 1800mAh = 1800x60x60 mAs / 5,56208 = ca. 1.165.031 Sekunden / (60x60x24) = 13,48 Tage

x + 0,08x = 12,79 d. h. x = 12,49

Übung Sensornetze – (für 18. November 2004)

Vorlesung 1: Motivation

Aufgabe 1.1: Lösung (Fortsetzung)

Variante, in der der Kanal immer abgehört wird:

Energie für Messung und Verarbeitung: 5 Messungen/Sekunde x 0,005 Sekunden x 20mA = 0,5mAs

Energie für Datenübertragung (Senden und Empfangen): Hier entfällt lediglich der Empfangsverbrauch

(200 Bytes x 8 Bit)/(9600 Bits/s) x [(8mA Grundverbrauch + 10 mA Sendeverbrauch) = 2/12 x18=3 mAs

Energieverbrauch Idle-Zeit: (Hier wird der sleep-Verbrauch ersetzt durch Grundverbrauch + Sendeverbr.)

Aktive Zeit: 0,025s Messungen u. Verarbeitung
0,167s Datenübertragung

Idle Zeit pro Sekunde (jed. ohne sleep): 0,808 s x (8 mA + 6 mA) = 11,32 mAs

Energie pro Sekunde

0,5mAs + 3 mAs + 11,32 mAs = ca. 14,82 mAs

Batterie hat 1800mAh = 1800x60x60 mAs / 14,82 = ca. 437246 Sekunden / (60x60x24x30) = 5 Tage

Übung Sensornetze – (für 18. November 2004)

Vorlesung 1: Motivation

Aufgabe 1.2: Die Länge einer Schwingung werde mit lambda bezeichnet. Aus der Nachrichtentechnik ist bekannt, dass Antennen der Länger lambda/4 Signale besonders effektiv abstrahlen. Weiterhin ist bekannt, dass sich ein Funksignal mit etwa 300.000 km/Sekunde bewegt. Wie lang sollte die Antenne eines Sensorknotens sein, wenn auf dem 2,4 GHz Band gesendet werden soll?

Lösung:

2,4 GHz bedeutet $2{,}4 \times 10^9$ Schwingungen/Sekunde.

Eine Schwingung dauert also $1/2{,}4 \times 10^9$ Sekunden. In dieser Zeit kommt das Funksignal $1/2{,}4 \times 10^9 \times 300000$ km weit = 12,5 cm

Wenn die Antenne bei ¼ der Wellenlänge besonders effektiv ist, beträgt sie optimal 3,125 cm.

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.1:

Bei einem Codewort eines Hamming-Codes kippt statt eines Datenbits ein Prüfbit? Kann der Fehler detektiert und behoben werden?

Lösung:

Bei der Prüfung des fehlerhaften Prüfbits wird eine falsche Parität entdeckt und der Zähler um die Nummer des Prüfbits inkrementiert. Weitere Prüfungen werden nicht fehlschlagen, denn alle anderen Daten- und Prüfbits sind korrekt. Somit enthält der Zähler auch bereits die Stelle des fehlerhaften Prüfbits. Der Fall wird also durch den Algorithmus bereits abgedeckt.

Aufgabe 2.2:

Gegeben sei das Hamming-Codewort 01111001111. Konstruieren Sie durch minimale Veränderung einen Übertragungsfehler, der nicht detektiert werden kann.

Lösung:

Prüfbits sind türkis
gekippte Bits sind rot

```
12345678901       12345678901
01111001111       11111000011
```

Für wenig Änderungen sollte man ein Datenbit wählen, das von möglichst wenigen Prüfbits abhängig ist. Einige Datenbits sind nur von zwei Prüfbits abhängig, so z. B. das Bit 9. Da fehlerkorrigierende Codes einen Abstand von 2d+1 besitzen, kommt man mit weniger Änderungen (als hier 3) auch nicht aus.

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.2: (Fortsetzung)

Konstruieren Sie durch maximale Veränderung einen Bitfehler, der nicht detektiert werden kann. Wie sieht der Algorithmus aus, mit dem dies bewerkstelligt werden kann.

Lösung:

Man iteriert über alle Prüfbits, also 1, 2, 4 und 8. Zu Anfang betrachtet man das Prüfbit 1 und alle darauf wirkende Datenbits, hier also 3, 5, 7, 9 und 11. Um die Parität nicht zu verändern, darf man nun eine gerade Menge dieser Bits kippen (evtl. inkl. des Prüfbits). Danach fährt man in gleicher Weise mit Prüfbit 2, 4 und 8 fort. Es ist jedoch darauf zu achten, dass ein bereits manipuliertes Bit nicht nocheinmal invertiert wird, da man damit sowohl die Anzahl der zuvor veränderten Bits verringern und gleichzeitig die vorherigen Pritäten gefährden würde.

```
12345678901      12345678901
01111001111      01111001111
```

Aufgabe 2.3:

Man möchte d Bitfehler korrigieren. Mit welcher Begründung reicht nicht bereits ein Abstand von 2d statt 2d+1 aus?

Lösung:

Ein ungerader Abstand ist nötig, damit bei d invertierten Bits ein kurzer und ein langer Weg zum nächsten gültigen Wort führt. Der kurze Weg ist dabei die erneute Invertierung der fehlerhaften d Bits, der lange Weg wäre ist Invertierung der verbleibenden d+1 Bits, um das nächste gültige Wort zu erreichen. Mit einem Abstand von nur 2d würde die Unterscheidung zwischen kurzem und langem Weg verloren gehen.

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.4:

In der Übung haben wir eine Abschätzung gesehen, wieviele redundante Bits einem Code mindestens hinzugefügt werden müssen, um einen 1-Bit Fehler zu detektieren und zu korrigieren. Führen Sie die gleiche Abschätzung (Sie müssen nicht den Code angeben) für 2-Bit Fehler aus, d. h. es sollen maximal bis zu zwei gekippte Bits wiederhergestellt werden können. Wieviele Bits braucht man mindestens, um so einen 7-Bit ASCII-Code zu sichern?

Lösung:

Der Code soll 2^m Datenbits haben und wird eine noch unbekannte Anzahl von r Redundanzbits benötigen. Bemerkung: Der Code aus 2^{m+r} Bits wird 2^m Datenwörter enthalten und $2^{m+r}-2^m$ ungültige Wörter.

Es können 1 und 2 Bit Fehler auftreten.

1 Bit Fehler: Für jedes gültige Codewort können durch Kippen eines der n Bits auch n ungültige Varianten entstehen.

2 Bit Fehler: Zuerst kann eines von n Bits gekippt werden. Danach kann eines der verbleibenden (n-1) Bits umfallen, so dass auf den ersten Blick n(n-1) Fehlermöglichkeiten entstehen. Jedoch ist die Reihenfolge gleich, in der die beiden Bits kippen. Es verbleiben daher nur 0.5 x n(n-1) Varianten.

Kein Bitfehler: Natürlich möchte man neben den vielen ungültigen Codewörtern auch jeweils ein gültiges kodieren können.

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.4: (Fortsetzung)

Lösung:

$$\left(\underbrace{n}_{\text{1-Bit Fehler}}+\underbrace{\frac{n(n-1)}{2}}_{\text{2-Bit Fehler}}+\underbrace{1}_{\text{Gültiges Codewort}}\right)2^{m}\leq 2^{r+m}$$

$$\left((m+r)+\frac{(m+r)(m+r-1)}{2}+1\right)2^{m}\leq 2^{r+m}$$

$$\frac{m+r+(m+r)^{2}}{2}+1\leq 2^{r}$$

$$\frac{7+8+(7+8)^{2}}{2}+1\leq 2^{8}$$

$$121\leq 256$$

Für die Sicherung eines 7-Bit ASCII Codes für bis zu 2 Bitfehler sind mindestens 8 Redundanzbits nötig.

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.5: Fehlerkorrektur vs. Neuübertragung

a)

Bei einer Datenübertragung ist im Mittel eines von 4000 Bits fehlerhaft, wobei die Bits statistisch unabhängig voneinander kippen. Ein (atomares) Datenpaket bestehe aus 128 Bytes. Es kann nur ganz oder gar nicht verschickt werden. Der Empfänger kann ohne Kosten nachprüfen, ob ein Paket korrekt übertragen wurde oder nicht. Wenn nicht, so wird beim Sender genau 1x um eine erneute Übertragung gebeten. Auch die Nachfrage ist wieder ein Paket der Länge 128 Bytes, das fehlerfrei ankommen muß. Ein fehlerhaftes Nachfragepaket wird behandelt, als wäre es nicht verschickt worden. Wie hoch ist der Datendurchsatz bei diesem Szenario? (in % des max. möglichen Durchsatzes).

Lösung Teil a)

Ein Bit wird korrekt übertragen, mit einer W' von $\frac{3999}{4000}$

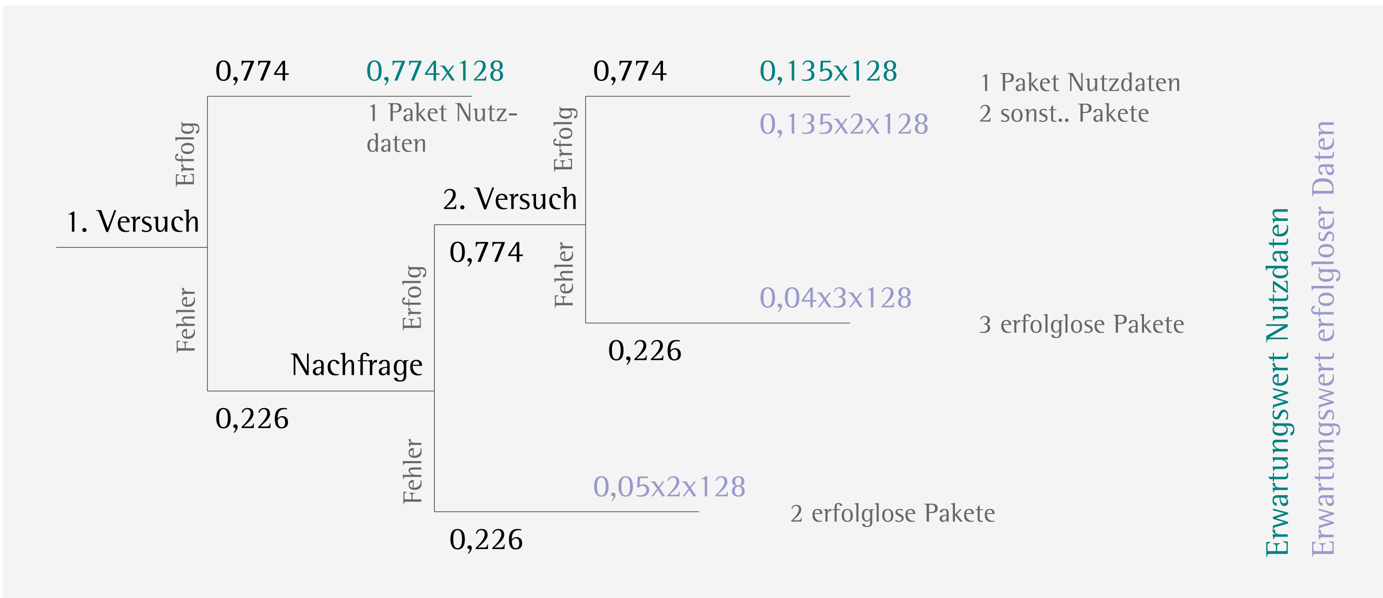
128x8=1024 Bit mit einer W' von $\left(\frac{3999}{4000}\right)^{1024} \approx 77{,}4\,\%$

Ein Fehler tritt also komplementär auf, mit $1-\left(\frac{3999}{4000}\right)^{1024} \approx 22{,}6\,\%$

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.5: Lösung Teil a)



Insgesamt sind also zu erwarten:

0,774x128 + 0,135x128 = ca. 116 Bytes Nutzdaten (Erwartungswert)
0,135x2x128 + 0,04x3x128 + 0,05x2x128 = ca. 63 Bytes verlorene Daten (Erwartungswert)

Der Durchsatz mit Nutzdaten beträgt also ca. $\frac{116}{116+63} \approx 64{,}8\,\%$

Übung Sensornetze - (für 18. November 2004)

Vorlesung 2: Kommunikation (MAC und Fehlersicherung)

Aufgabe 2.5:

b)

Zur Vereinfachung sei angenommen, dass nur einzelne Bitfehler pro Paket auftreten. Statt der Nachfrage entscheidet man sich nun dafür forward-error-correction zu betreiben und verwendet dafür den fehler-korrigierenden Huffman-Code aus der Vorlesung. Wie hoch ist der Datendurchsatz nun in %? Hat sich die Redundanz ausgezahlt?

Lösung Teil b)

Um 1024 Bit auf 1-Bit Fehler zu sichern muss folgende Ungleichung erfüllt sein:

$1024+r+1 <= 2^r$

ab r =11 gilt

$1024+11+1 <= 2048$

Der Durchsatz mit Nutzdaten beträgt hier $\frac{1024}{1024+11} \approx 98{,}9\,\%$

Fazit: Vor allem bei gleichmäßig verteilten Fehlern, also z. B. bei Funkübertragung in Sensornetzen kann forward-error-correction sinnvoller sein als ein Protokoll zur Fehlerbehebung.