Wireless Communication and Networked Embedded Systems

Exam - August 27, 2021
10:00 - 13:00

Submission format: one PDF file, to be uploaded on Studium > Exam - Part B.
Typeset your answers. Place everything belonging to the same question together in the document, and start a new page for every new Problem.

Include your exam code in the document. Grading is anonymous and your name should not be included.

By submitting your solution, you confirm that you solved all the questions by yourself and did not collaborate or use help from anyone else.

Instructions:
In some of the problems, you may not be given all the details needed to answer the problem. In these cases, you should make realistic assumptions that are explicitly stated and motivated.

Good luck with the exam!

Problem 1: Embedded Operating Systems (6p)

Contiki: Study the provided TinyOS code snippet and consider the following questions.

a. Briefly but concisely describe in your own words what the code does from an application point of view. (2p)
Note: Be specific about how things are done (e.g., ”Broadcast the temperature every 3s”). Explaining line by line or function by function will not give any points. Two sentences should be enough.

b. The code uses the boolean variables `locked`. What is it used for in a system context and what would be the consequence of not testing this variable? (1p)

c. If you would write the code in Contiki, what are the conditions c in PROCESS_WAIT_UNTIL(c) you would have to implement for the same functionality? (1p)

Checkpointing in intermittent computing: Consider code that writes a checkpoint every *N*-th instruction. A node harvests energy to run *M* instructions before booting.

d. Chose $N \in [10,50]$, $M \in [20,100]$ ($N \neq M$): At what instruction will the node resume its operation after having run out of energy three times? (2p)
Advice: draw a timeline to get an overview of the problem.

```
typedef nx_struct exam_msg {
       nx_uint16_t counter;
} exam_msg_t;

enum {
       AM_EXAM_MSG = 6,
};

------Configuration---------

#include "ExamMsg.h"
configuration ExamClientAppC {}
implementation {
       components MainC, ExamClientC as App, LedsC;
       components new AMSenderC(AM_EXAM_MSG);
       components new AMReceiverC(AM_EXAM_MSG);
       components new TimerMilliC();
       components ActiveMessageC;

       App.Boot -> MainC.Boot;

       App.Receive -> AMReceiverC;
       App.AMSend -> AMSenderC;
       App.AMControl -> ActiveMessageC;
       App.Leds -> LedsC;
       App.MilliTimer -> TimerMilliC;
       App.Packet -> AMSenderC;
}
```

```
------Implementation---------

#include "Timer.h"
#include "ExamMsg.h"
module ExamClientC @safe() {
	uses {
		interface Leds;
		interface Boot;
		interface Receive;
		interface AMSend;
		interface Timer<TMilli> as MilliTimer;
		interface SplitControl as AMControl;
		interface Packet;
	}
}

implementation {
	message_t packet;

	bool locked;
	uint16_t counter = 0;

	event void Boot.booted() {
		call AMControl.start();
	}

	event void AMControl.startDone(error_t err) {
		if (err == SUCCESS) {
			call MilliTimer.startPeriodic(250);
		} else {
			call AMControl.start();
		}
	}

	event void AMControl.stopDone(error_t err) {
		// do nothing
	}

	event void MilliTimer.fired() {
		counter++;
		if (locked) {return;}
		else {
			exam_msg_t* rcm =  (exam_msg_t*)call
				Packet.getPayload(&packet, sizeof(exam_msg_t));
			if (rcm == NULL) {return;}

			rcm->counter = counter;
			if (call AMSend.send(AM_BROADCAST_ADDR, &packet,
				sizeof(exam_msg_t)) == SUCCESS) {
				locked = TRUE;
			}
		}
	}

	event message_t* Receive.receive(message_t* bufPtr, void* payload, uint8_t len) {
		if (len != sizeof(exam_msg_t)) {
				return bufPtr;
			} else {
				exam_msg_t* rcm = (exam_msg_t*)payload;
				if (rcm->counter & 0x1) {call Leds.led0On();}
				else {call Leds.led0Off();}
				if (rcm->counter & 0x2) {call Leds.led1On();}
				else {call Leds.led1Off();}
				if (rcm->counter & 0x4) {call Leds.led2On();}
				else {call Leds.led2Off();}
				return bufPtr;
			}
		}

		event void AMSend.sendDone(message_t* bufPtr, error_t error) {
			if (&packet == bufPtr) {
				locked = FALSE;
			}
		}
	}
```

Problem 2: Energy Efficiency (6p)

Preparation:
Draw a graph/network with 6 nodes and 11 edges. Mark one node as base station "BS", enumerate the others with A..F, and assign weights between 1 and 5 to each edge in a non-trival way. One edge connecting the base station to another node shall have a weight corresponding to the last digit in your exam code modulo 5 plus 1 (e.g., AB-0057-CDE -> weight = (7 mod 5) + 1 = 2 + 1 = 3).

In the following, the edge weights represent the expected number of transmissions (ETX) for that edge.

Spanning tree

a. Give the minimum cost spanning tree with root at the base station for your network. (mark the edges belonging to the spanning tree with a different color or make them thicker) (1p)

Energy, Data rate and error coding
In the following task, we vary the data rate and error coding in the given network. We assume that transmission takes the same amount of power for all data rates and coding, and that one transmission with a data rate of 200kbit/s costs 1*E* Joule, and that the edge weights in your network correspond to the case of 200kbit/s (ETX_{200}). We consider the following two additional cases:

- 100kbit/s without error correction (edge weights ETX_{100})
- 200kbit/s with forward error correction coding (FEC). The coding adds 50% to the packet length. (edge weights ETX_{200FEC})

b. Suggest a relation between ETX_{100}, ETX_{200} and ETX_{200FEC}. Motivate briefly your choice and present the relations in form of a table. (2p)

Draw your network for the two additional cases and remove edges with ETX>5. Don't mind if nodes get unconnected.

c. Compute the expected energy consumed per (connected) node with data aggregation for the 200kbit/s and 200kbit/s+FEC networks. (2p)

d. Assume that you can configure the data rate and error coding individually per link. Assign data rate and error coding for the links along the spanning tree. Argue for your assignment. (1p)

Problem 3: Digital Modulation (10p)

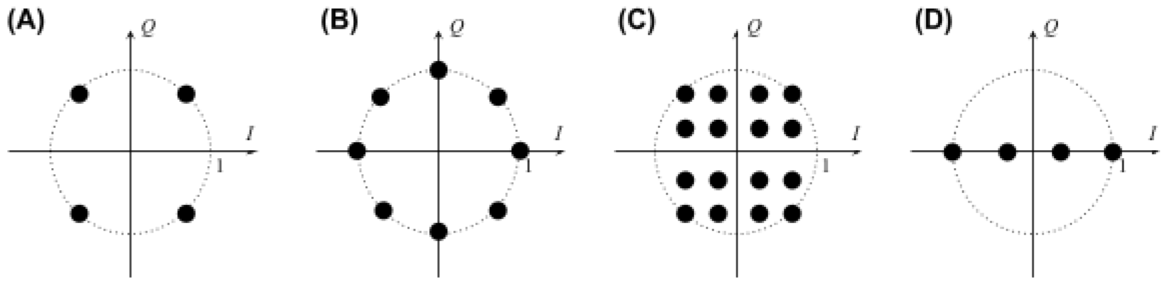

Figure 3.1

Consider the four modulation constellations (A..D) illustrated in Figure 3.1. Assume that all the modulations have the same symbol rate $1/T$ and that we have a constant noise floor for all modulations (Noise energy *N0*).

	(A)	(B)	(C)	(D)	yours
Modulation order [number]					-
Type of digital modulation [amplitude, phase, frequency, etc]					-
Bit rate [ranking↘ with order number 1,2,3,4]					> min
Average Symbol energy [ranking↘]					< max
Symbol error rate [ranking↘]					-

a. Compare the four modulations (A..D) by filling the table above. If an answer depends on some assumptions, state them! (3p)
Note: You sometimes have to give an absolute number, sometimes a ranking in decreasing order.

b. Suggest one your own modulation constellation with (1p)
- bit rate higher than the lowest in (A..D)
- average symbol energy lower than the maximum in (A..D)
To get full points motivates your choice!

c. Draw the decision boundaries for your constellation. (1p)
If you have symmetries in your constellation you only have to show the boundaries for a representative part of the constellation.

d. Assume random noise in both the I and Q components. The noise is uniformly distributed in the range [-0.66, +0.66]. What are the expected symbol error rates for each of the symbols in modulation A and D? (2p)

e. For one of the modulations (A..D), assign bits to the symbols and plot the time signal of the bit-sequence 100111. Mark clearly where a new symbol starts. (1p)

f. Consider two different bitmaps for modulation (A): the Gray coded bitmap and the Lexicographic bitmap, which are visualized in Figure 3.2 and 3.3. Derive the average bit error rate for each bitmap in relation to the given symbol error probabilities p_1 (horizontal and vertical) and p_2 (diagonal). Which bitmap has a lower average bit error rate? Motivate your answer. (2p)

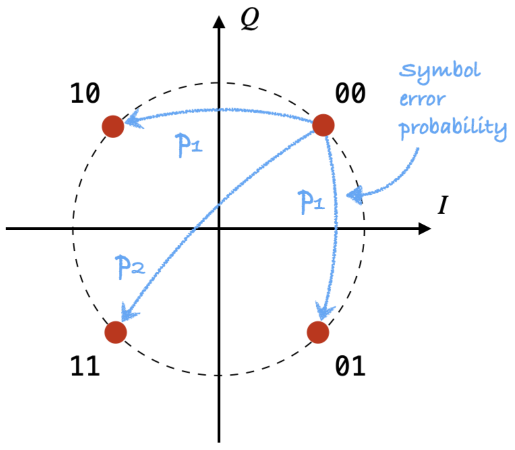

Figure 3.2: Gray bitmap

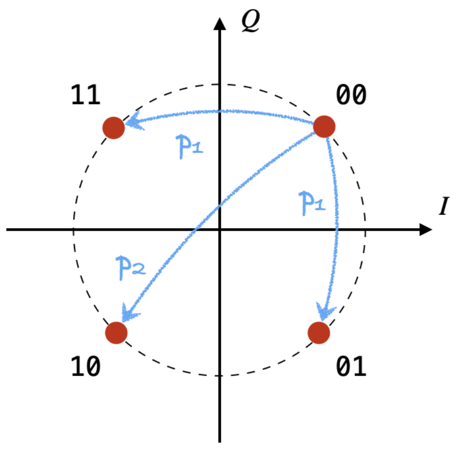

Figure 3.3: Lexicographic bitmap