Electronics and Computer Science

UNIVERSITY OF Southampton

ELEC3222 Computer Networks

Coursework 2015/16

Assignment Set:	Thursday 29th October 2015, during lecture
Assignment Type:	Individual coursework, but working in teams
Submission Deadline(s):	Tuesday 10th November 2015, during lab sot (*oral*) Friday 11th December 2015, during lectures (*oral*) Monday 04th January 2016, by 4pm (*online hand-in*)
Feedback:	You will receive oral feedback on your design at the Design Review Meeting, and at the Practical Demonstration session. You will receive written feedback and a mark for your report within three weeks of submission.
Mark Contribution:	This assignment is worth 30% of your mark for ELEC3222
Required Effort:	You should expect to spend up to 45 hours on this coursework
Examiners:	Dr Geoff Merrett and Dr Jeff Reeve

Learning Outcomes

Having successfully completed this coursework, you will be able to:

- interpret standardisation documents
- design and analyse embedded networked devices and systems

Individual or Team?

This coursework is an individual piece of coursework, but you will be working in *teams*. The *team* allocations are listed below:

Team A1	**Team A2**
Yubo Zhi Jiayang Sun Jiuxi Meng	Tu Lan Fuxin Zhou Alaa Khoja
Team B1	**Team B2**
Dominic Maskell Nathan Ruttley Huw Percival	Samuel Presley Vincent Chan Anthony Lau

Team C1	**Team C2**
Dongyu Wang Zhanwen Pan Robert Davis	Dimitrios Prapas James Prance Terra Barber
Team D1	**Team D2**
Tobias Fok Lawrence Harlow Marjan Krsteski	Timothy Furlong Henry Wilson Grisha Gevorkyan

Some **important** definitions:

- Individuals that are in the same *team* (e.g. everyone in Team A1) are *teammates*.
- *Teams* that share the same letter (e.g. B1 and B2) are *peer-teams*.
- Individuals working on the same layer(s) in *peer-teams* are *peer-teammates*.

Your Task

A full network architecture supporting 'chat' using the Il Matto + RFM12B

Your *team's* task is to design and implement a full, working, network architecture capable of allowing multiple users to 'chat' using it. Jabber, MSN Messenger, Blackberry Messenger, Facebook Messaging, Skype, iMessage etc could all be considered as implementing this kind of functionality. How you interpret this, and the services you offer are up to the *teams* and *teammates* to decide. You are not being asked to implement an existing network architecture (e.g. TCP/IP or ZigBee); in fact we want you to create your own. That said, there is no requirement to invent new algorithms (e.g. you can implement existing error control algorithms, routing algorithms, etc). You should implement your architecture using the Il Matto + RFM12B hardware. Your *team* should already have at least 3x Il Matto boards and 3x RFM12B radio modules, and you should demonstrate your network using all of these.

Note the wording of the assignment: you are being asked to implement a network architecture, not a protocol stack. This means that your *team* could choose to implement multiple protocols at a particular layer (for example both a reliable and an unreliable DLL). However, there is no necessity to do this, and your network architecture may be a single protocol stack.

Although you are working in *teams*, you must each work on unique and clearly defined tasks. The recommended way to implement this would be to give each *teammate* different layer(s) from the 5-layer reference model considered on this module (PHY, DLL, NET, TRAN, APP), where everyone should have at least one layer to work on. However, in some cases, it might be necessary to split a layer into sub-layers (e.g. the MAC and LLC sub-layers of the DLL) in order to appropriately divide tasks. Whatever you choose, it should be clearly defined, agreed amongst *teammates*, and in-line with the spirit/principles of protocol layering.

When you settle on your division of labour, one of the first things that you need to do is agree on the (initial) functionality of each layer and then define the software interfaces. That is, you need to agree on a prototype for the function to call when one layer needs the services of another. You have to keep in mind that the end product is one integrated program.

As mentioned above, although you will be working in *teams*, this is an individual coursework. You should only interact with your *teammates* through clearly defined and agreed interfaces (although you may also need to work together to manage the limited hardware resources, e.g. RAM, CPU cycles, etc).

TRAN:

Control [2]	SRC Port [1]	DEST Port [1]	Length [1]	APP Data [1-114]	Checksum [2]

NET:

Control [2]	SRC Address [1]	DEST Address [1]	Length [1]	TRAN Segment [8-121]	Checksum [2]

DLL:

Header [1]	Control [2]	Addressing [2]	Length [1]	NET Packet (or part of) [1-23]	Checksum [2]	Footer [1]

Figure 1: Segment/packet/frame field structures defined by the coursework

As a starting point for your protocols, the packet/frame field structures in Figure 1 are defined by the coursework and are non-negotiable. There are multiple ways to use some of these fields (e.g. the Checksum field: this does not have to be a true 'checksum', and could be a parity bit, interleaved parity, a checksum, a CRC, etc; the bits in the Control fields can be used how you like, etc). The numbers in square brackets refer to the field size in bytes; however, if you do not want to use a particular field (or some of the bits within a field), just replace them with zeros.

As can be seen in Figure 1, If any NET packet is >23 bytes, it will need to be split into multiple frames. Likewise, if the APP data is >114 bytes, it will need to be split into multiple TRAN segments. You may wish to start by designing your system to work with small amounts of data (<9 bytes, so that it all gets encapsulated into a single DLL frame), and then expand it later.

Standardising your Network Architecture

A proprietary network architecture is fine for communicating within your own network, but limits the potential of a network. For this reason, you should also agree a standard (defining your network architecture) with your *peer-team*, such that you can also demonstrate successful communication with the nodes/stations in their network. Remember, a protocol only defines how communication between peer-layers should function, NOT how it should be implemented.

As mentioned above, although you will be working in *teams*, this is an individual coursework. You should interact with your *peer-teammate* only to define and agree protocol(s). No other collaboration should be necessary (in fact, you will be penalised if it occurs). You do not need to subdivide the tasks in the same way as your *peer-team* (i.e. there does not have to be a 1:1 mapping between the tasks being undertaken by *peer-teammates*).

Your agreed standard may include some optional functionality that is not necessarily implemented by both *peer-teammates*. For example, your DLL might include a bit in the Control field which indicates whether the Checksum field is using a single even parity bit (core functionality) or a 32-bit CRC (optional functionality). However, the presence or absence of these optional functionalities should not stop the 'core' (non-optional) functionality from working. This ensures that no-one is limited by the capabilities and aspirations of their *peer-teammate*.

If you are not sure what a standard looks like, look at http://standards.ieee.org/about/get/ (however, yours doesn't have to be – and shouldn't be – hundreds of pages long)!

Disclaimer: *We are trying to make this engaging, practical and useful (and fun), rather than having an individual or theoretical coursework. However, bear in mind that this makes it quite intricate and complex, and this is the first year that the coursework has run. As such, we may need to make minor tweaks and modifications to the specification and rules as the coursework develops (while ensuring fairness). Apologies if this happens, but please be patient with us!*

Deliverables and Marking Scheme

This coursework is marked out of 100, and contributes 30% of the credit for this module. More marks will be awarded for ambitious implementations if, and only if, the ambitious functionality works. Hence, there is no point aiming for something really complicated if you know you won't be able to get it to work! The coursework has the following deliverables:

- **Design Review Meeting [total of 0 marks]**
  (held during the lab slot on Tuesday 10th November 2015)
  - The examiners will meet with all *teams* in the lab. *Teams* (and all of the *teammates* in them) should explain their proposed network architecture and the protocols and services of each layer. Any progress on standardisation with the *peer-team* should also be explained. Printed hand-outs are encouraged to aid discussion. This deliverable is not assessed, but provides feedback.

- **Practical Demonstration [total of 30 marks]**
  (assessed in lecture slots on Friday 11th December 2015)
  If an individual does not attend their scheduled demonstration session, they will receive a mark of zero for this element.
  - **Demonstration of functioning layer(s) in isolation** [15 marks]: Individual *teammates* should demonstrate the functionality of their layer(s) and explain what is happening and what services are provided. Think about how you are going to demonstrate this to us, even if your *teammates* do not get their parts working – printed hand-outs are encouraged.
  - **Demo of layer(s) functioning as part of stack** [15 marks]: *Teams* should demonstrate their network architecture permitting communication across their network. Individual *teammates* should explain the relevance of their part.
  - **Demo of networking between *peer-teams*** [0 marks]: *Peer-teams* should demonstrate their networks communicating with each other. No marks are allocated to this, for fairness, but it will help the examiners interpret reports!

- **Individual Report [total of 70 marks]**
  (due by 4pm on Monday 04th January 2016, online hand-in)
  Your report should be structured to include ALL of the following:
  - **Introduction** [0 marks]**:**
    - Your report should begin with your name, student ID, and team letter/number. It should then include a diagram of your team's network architecture (protocol stack(s)), highlighting the layers/sub-layers that you worked on. This should be accompanied by a paragraph or two explaining the network architecture your team adopted.
    - The introduction carries no marks of its own. However, failing to include this will mean that we cannot fully understand the remainder of the report, and hence will lose you marks in those sections.
  - This should be followed by a section for each layer/sub-layer you worked on. For each (note, you may have only worked on one), include subsections for:
    - **Standard Document** [20 marks]:
      - Agreed and collaborated on between *peer-teammate(s)*.
      - It is assumed that this is a joint-effort between *peer-teammates*, and hence all will get the same mark for this part.

- **Design** [20 marks]:
  - A clear account of how *you* chose to implement different parts of the standard, and the design and implementation of the algorithms. What existing algorithms did you implement or adapt? How did they work? How did you structure your code to ensure that you followed the principles of a layered architecture (i.e. interfaces, services and protocols)?
- **Testing, Results and Analysis** [20 marks]:
  - How do you know your implementation worked and implemented the protocol? How well did it work? Did it work on its own (i.e. in a test-harness) and as part of your team's network architecture? What evidence do you have to support all of this?
- **Critical Reflection and Evaluation** [10 marks]:
  - What would you do differently if you were to repeat it? What are the weaknesses of your design? What problems did you encounter working in a *team*, agreeing on interfaces between layers, and agreeing on the standard with your *peer-teammate*?

Academic Integrity

Please ensure that you are aware of the University's regulations on academic integrity, particularly on collaboration vs collusion. While you will be working in *teams*, this is an individual coursework and the report that you submit must be your own work. Any collaboration must be clearly declared. While you are expected to discuss (and standardise) the protocol for your layer(s) with your *peer-teammate*, these discussions should only be at the abstract level, i.e. not discussing the implementation, code, etc. *Note: it is however expected that the 'standard' document in your individual report will be the same for peer-teammates, as it is inherently collaborative and should be identical for it to be a standard!*