KTH Electrical Engineering

Wireless Water Tanks

with LabVIEW GUI

ANSER AHMED, JAMES WIEMER AND JOSE ARAUJO

Stockholm September 28, 2012

TRITA-EE 2011:XXX
Version: 0.1

Contents

Chapter 1

Overview

1.1 Introduction

In the Automatic Control lab a Coupled Water Tanks is extensible used for academic purposes, either in a Basic Control course or in the Advance Control course. In this document we present the LabVIEW GUI along with IEEE 802.15.4 based Wireless Sensor Actuator Network which will be used to do experiment on Coupled Water Tank.

To implement the Wireless Sensor Actuator Network we use the motes from TmoteSky [2], Telosb [12] or MAXFOR [1] using the TinyOS [14] operating system. For the wireless communications we used the extended IEEE 802.15.4 [7] protocol that can support a maximum of 32 Guaranteed Time Slot (GTS). For details about IEEE 802.15.4 protocol, its GTS implementation and extended IEEE 802.15.4 protocal we recommend following references [3],[6],[8].

For information regarding the modelling, and simple control laws we recommend the following references from Quanser [9, 10, 11] . This document is focused on the implementation and communication between the motes, not on the control perspective.

The code and software necessary to run the experiments explained below is available on the following URL:

http://code.google.com/p/kth-wsn/source/browse/trunk/kth-wsn/apps.water-tank/water-tank-GTS.

To compile the applications we recommend the use of the latest version of TinyOS that you could find on the following URL:

http://code.google.com/p/tinyos-main/

1.2 Setup

In this section we show how we need to setup the water tanks, Universal Power Module (UPM) and the new board where we need to plug our motes.

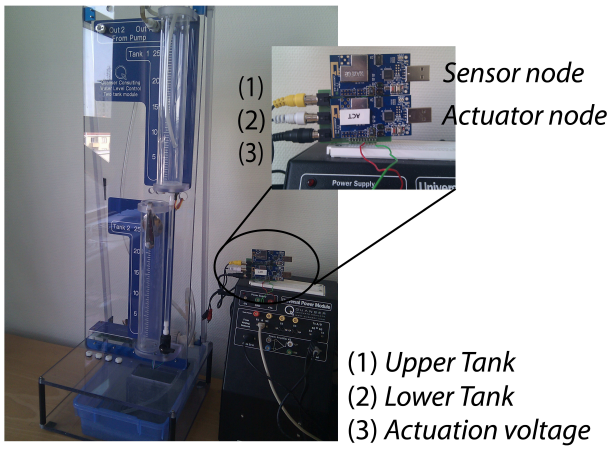

Figure 1.1: Water tank with the UPM and the PCB board connected

Figure 1.1 show a Coupled water tank with the UPM and a Printed Circuit Board (PCB) connected to it. The cables are connected following Quanser configuration specification in [11, p. 12 - 14]. However in our case instead of connect the sensor values to the Q8 Terminal Board, we connect them directly to our PCB board.

The PCB has connected three RCA cables: white cable is for the Upper tank level, yellow cable for the lower tank level and the black cable is for the actuator voltage. Moreover the PCB needs to be supplied by +12 V. For this reason, we connect the Ground of the UPM to the first PIN of the socket (starting from the right) and the 12 V (+Vs) to the PIN number 5.

Figure 1.2 shows the connection that the motes have with the PCB board. The PIN V_{out} is exclusive for the actuator mote.

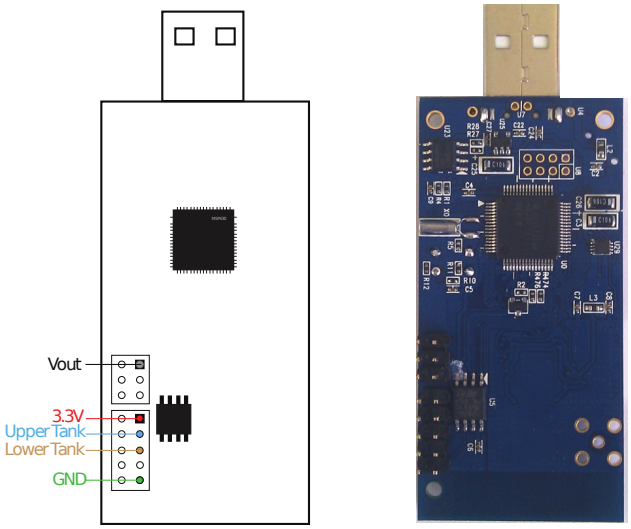

Figure 1.2: Motes connections. View from the bottom

Chapter 2

Scenario

The scenario for our application is based on a Centralized Controller were the sensors data is transmitted from the sensors to the coordinator, the coordinator is attached to PC where control algorithum will be run.

2.1 Centralized Controller

This application consists on having a centralized Controller. Table 2.1 shows the different programs that are needed to deploy this scenario.

Application	Description
`Coordinator`	a coordinator for network also used as a base station
`SensorApp`	sensor connected to the PCB board in the water tank
`ActuatorApp`	actuator connected to the PCB board in the water tank

Table 2.1: Centralized Controller applications

Figure 2.1 shows the scenario with three water tanks and the coordinator. The coordinator is connected to the computer through serialforward. On the PC we are running LabVIEW with mathscript node. The mathscript is responsible for all control and scheduler algorithm. The `SensorApp` samples the water tank levels and sends the values to the coordinator using IEEE 802.15.4 protocol. The coordinator forward this value to the LabVIEW to calculate actuation voltage, which will be send back to the coordinator.The coordinator send actuator voltage to `Actuator` using IEEE 802.15.4 protocol. In the next sections we explain it with more detail.

The current LabVIEW application can support a maximum of six water tanks. The number of water tank can be further increased by slightly modifying the LabVIEW application.

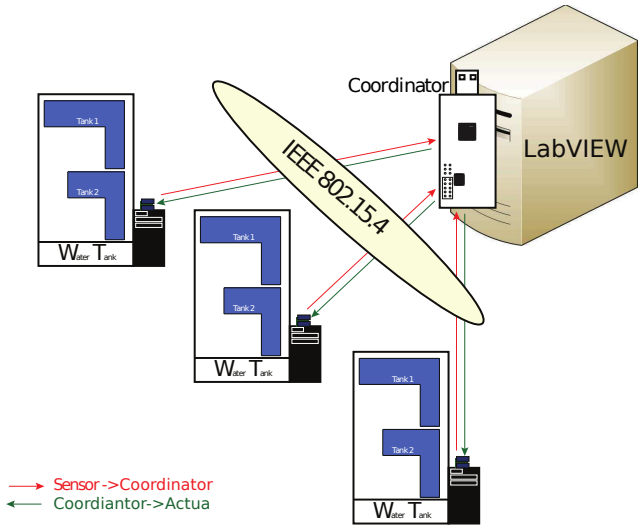

Figure 2.1: Scenario with a coordinator mote connected to the computer and working as a bridge between the controller and scheduler in the PC and the three water tanks

2.1.1 Requirements

First of all, it is mandatory that at this point, you have read the following documents:

- *Getting started with TinyOS at the Automatic Control Lab* [5]
- *Communication between PC and motes in TinyOS* [4]

Below we show a list of requirements that you need to run this example.

- N[1] Quanser Coupled Water Tanks
- N PCB board to connect the motes with the water tanks
- $2 * N + 1$ Telosb, TmoteSky or MAXFOR motes
- TinyOS properly installed [14]
- Source code for the GTS implementation [6] http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/kth/tkn154-gts
- Source code for the modified version of the GTS implementation [7] http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/kth/tkn154-gts-mod

[1] Where $N <= 6$

- Source code for the motes http://code.google.com/p/kth-wsn/source/browse/trunk/kth-wsn/apps.water-tank/water-tank-GTS
- Scripts to program the motes http://code.google.com/p/kth-wsn/source/browse/trunk/kth-wsn/apps.water-tank/water-tank-GTS
- LabVIEW with mathscript node
- Matlab (optional)
- Serial-Forwarder from TinyOS, C version. [4]

2.1.2 Communication

For communication between motes, we use extended IEEE 802.15.4 protocol [7] configured in the beacon-enable mode. In the IEEE 802.15.4 standard protocol the maximum number of slots in the Contention-Free Period (CFP) is limited to 7. So, we are using extended IEEE 802.15.4 protocol that can support upto 32 GTS slots. The mote connected to PC act as a coordinator of the network.

In the beacon-enabled mode, the coordinator sends beacons periodically and the motes are synchronized between each other. With this mode, we have two ways of transmission, we could transmit during the Contention Access Period (CAP) or CFP. During CAP the sensors request GTS slots these request is forward by the coordinator mote to PC(LabVIEW + mathscript). All the sensor and actuator are schedule by PC. After sensor and actuator get GTS slots these will transmit and receive within the certain slot assigned by the scheduler.

Figure 2.2 shows the example of superframe structure for the experiment.In summary in CAP period motes request GTS slots, in the CFP period the scheduled motes start communication with the coordinator and in the inactive period the LabVIEW will be operate.

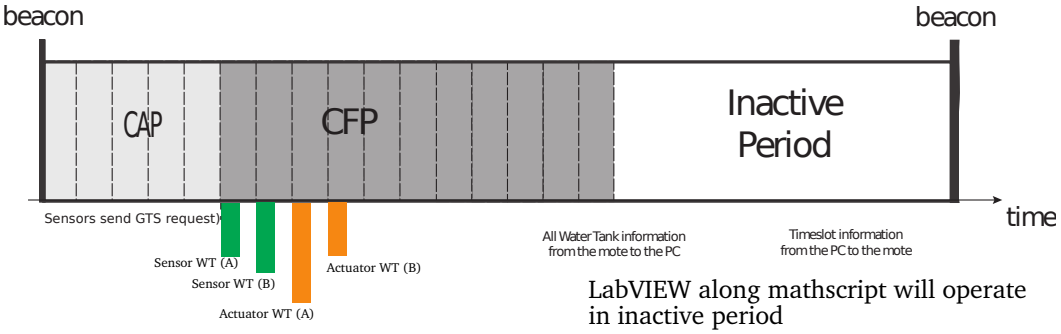

Figure 2.2: Superframe structure with $aNumSuperframeSlots - 1$ GTS slots

2.1.3 Installation

We omit the installation methods assuming that the reader already knows how to program the motes. If not, please before continue with this document read the following references [13, 5]

Number of water tanks

The maximum number of water tanks is given by the constant `NUMBER_WT` founded in the `app_profile.h`. As explained earlier the maximum number of water tank should be less or equal to 6.

The controller assumes that each water tank is defined with the following IDs:

Sensor	1 to 6
Acutator	9 to 14

An example for three water tanks would be:

Controller		`ID = 0`
Water Tank 1	Sensor	`ID = 1`
	Actuator	`ID = 9`
Water Tank 2	Sensor	`ID = 2`
	Actuator	`ID = 10`
Water Tank 3	Sensor	`ID = 3`
	Actuator	`ID = 11`

2.2 LabVIEW GUI with supporting functions

In this section we will explain about the developed LabVIEW GUI and matlab function that are necessary for running application.

2.2.1 Graphical User Interface

We have developed the LabVIEW Graphical User Interface (GUI). The LabVIEW is connected to coordinator through serialforward. Figure 2.3 shows an screenshot of how the GUI looks like. The GUI will operate on inactive period of Superframe. At the end of CFP period the coordinator send all received data to the GUI. Depending upon data,LabVIEW's matscript node will calculate scheduler or/and actuation voltages for the Actuators.

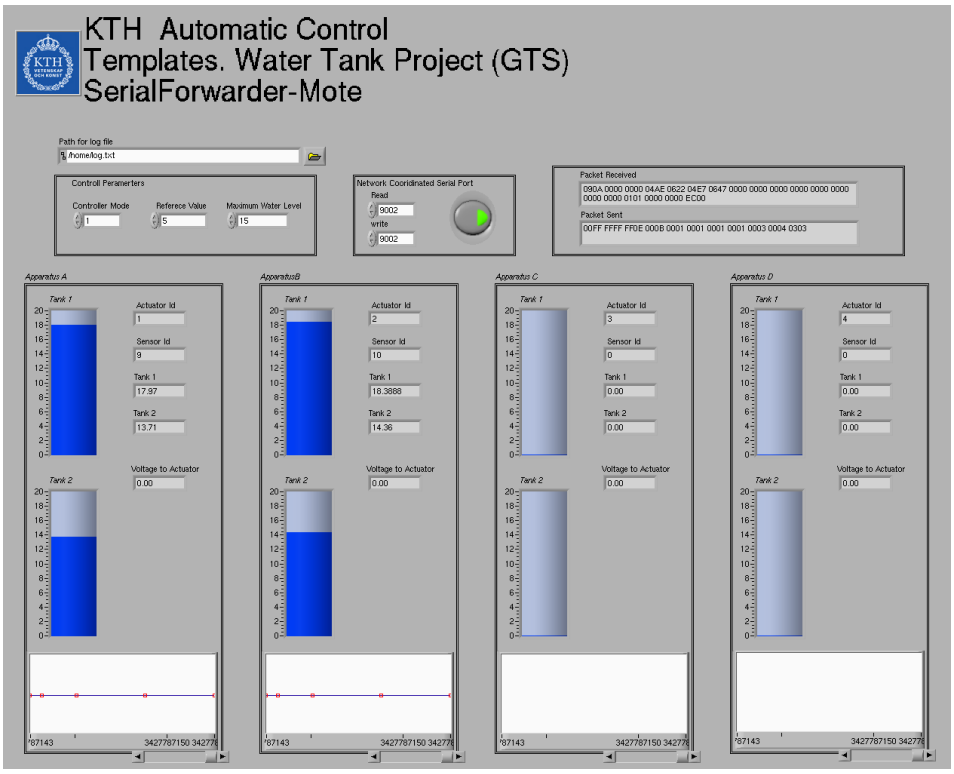

Figure 2.3: Screenshot of the GUI for the Water Tanks

Water Tank Panel

Figure 2.4 shows main part of GUI, the water tank. On the left half of figure the water level is showing graphically. The blue colour indicate the presence of water on the tanks. On the right side we have the mote Id of sensor and Actuator to which these level belongs. The water levels and voltage for the Actuator is also shown by numeric indicator.

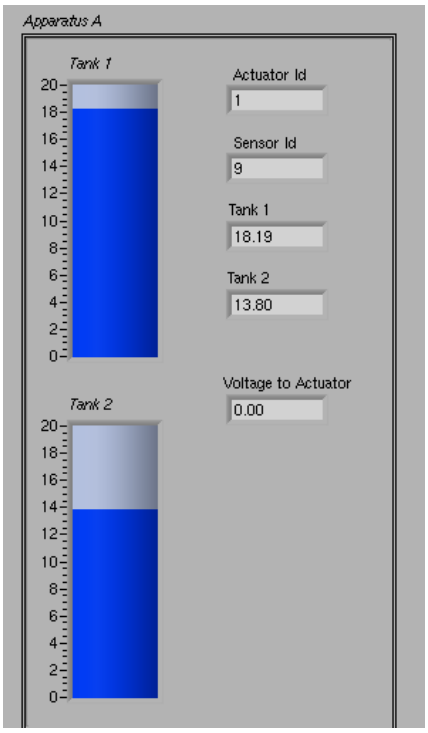

Figure 2.4: Water Tank and other parameters

Control Parameters

Figure 2.5 shows some of the control parameters that we can modify from GUI. The *Controller Mode* will select different control algorithm i.e or constant voltage Controller. The *Reference Value* is the water level that we want our tank to go and finally the *Maximum Water Level* will limit the maximum water level that a tank can have. If the water try to exceed the level the Actuator will be stopped to prevent overflow.

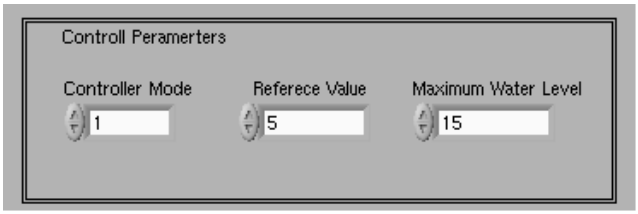

Figure 2.5: Control Panel

Communication Indicator Graph

To indicate the communication on GUI a time graph is placed below every Apparatus. As an example see figure 2.6. The blue square indicate the time when GTS slot

has been requested. The red square indicate the time at which GTS slot is assigned by the Coordinator. The solid blue line indicate the the start of communication.

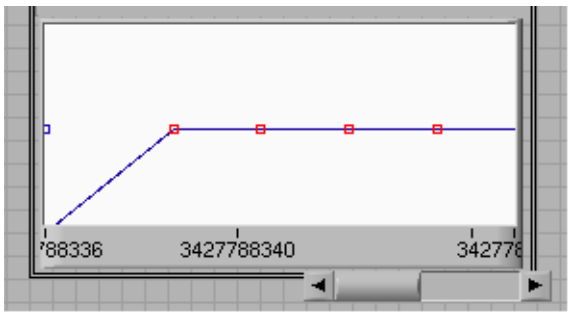

Figure 2.6: Communication Indicator Graph,

Network Coordinator Serial Port

To be able to read and write data from Coordinator, LabVIEW use C based serialforward. Figure 2.7 shows network coordinator serial port panel. It will be used to select the serial port for read and write. Since we are using same mote to read and write from the LabVIEW, The port number in both *read* and *write* should be same. It also important to enable the *Toggle button*, other wise no data will flow from LabVIEW to Coordinator.

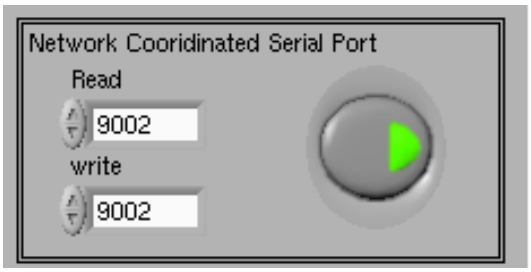

Figure 2.7: Network coordinator serila port panel,

Plotting Option

In the GUI, we log all data from LabVIEW (received and sent data) which include all water tanks levels and Actuator Voltages. Beside that we are also logging communication data which includes, GTS requests and GTS assignments. The data can be logged by specifying the file name in the data logging panel. The data parsing can be done off-line by using `paraseData.m` matlabfile function.Figure 2.8 shows the screen shot for data log panel.

Figure 2.8: Panel to save log data ,

2.2.2 Libraries

The purpose of this document is not to have a extensive explanation about the code implemented and libraries. However, it is important to give an guidance where to start looking at. Below we have a table with the main files that are involved on this implementation.

File	Description
`mathscript_contol.m`	The files includes the core of the water tanks experiment. It will be uploaded on the mathscript node. All the sheduling will be done from this file.
`contoller.m`	It contains control algorithm for calculating actuating voltage. This function is called from `mathscript_contol.m` file.
`praseData.m`	We use this function to parse the logged data for plotting purpose.

2.2.3 Conclusion

The LabVIEW is an excellent tool for creating GUI for embedded application with much lower complexity as compare to Matlab. For implementing control algorithm, its much easier to use matlab, also most academic people are familiar with Matlab. The LabVIEW with mathscript node have combine the advantages of Matlab and LabVIEW, giving us ability to create nice GUI and to implement control algorithm on matlab efficiently.

Appendices

Appendix A

Description of system and implementation

There are two Tmote-sky wireless nodes in each double tank setup. They perform the following functions:

Sensor Node: A T-mote sky situated on the UPM is connected to the sensors from the tank process, and collects and transmits this information to the Coordinator node.

Actuator Node: A T-mote sky situated on the UPM is connected to the actuator of the tank process. This node receives the control signal from the coordinator and applies this voltage to the pump in the tank setup.

Beside that there is another T-mote sky, acting as a coordinator node of the network.

A.1 Important parameters in the header file

The following brief list introduces some of the parameters that is important to know for this application.

`RADIO_CHANNEL`: The frequency channel number that motes will used in the network.

`PAN_ID`: The PAN Id of the network.

`BEACON_ORDER` : The beacon order.

`SUPERFRAME_ORDER`: The super frame order.

Since we are operating LabVIEW on inactive period of the frame, the `BEACON_ORDER` must be greater than `SUPERFRAME_ORDER`.

A.2 Steps to run the experiment

Step 1: Programme the motes according to the section [**?**].

Step 2: Open the terminal window and open serialforward for the coordinator,from following command `sf 9002 /dev/ttyUSBX 115200`.

Step 3: Open LabVIEW VI `WT_main.vi`.

Step 4: Enable the *Toggle button* on the Network Coordinator Serial port panel of the GUI.

Step 5: Select the file name with the extension of *.txt* to save the logging data.

Step 6: Run the LabVIEW VI.

Step 7: Press *user button* on the coordinator mote to start the experiment.

References

[1] Advantic. Maxfor data sheet. Technical report, Madrid, 2010. URL http://www.advanticsys.com/files/CM5000.pdf.

[2] Moteiv Corporation. Tmote sky data sheet. Technical report, San Francisco, June 2006. URL http://www.bandwavetech.com/download/tmote-sky-datasheet.pdf.

[3] Jan-Hinrich Hauer and Adam Wolisz. TKN15.4: An IEEE 802.15.4 MAC Implementation for TinyOS 2. Technical report, Technical University Berlin - Telecommunication Networks Group, March 2009. URL http://www.tkn.tu-berlin.de/publications/papers/TKN154.pdf.

[4] Aitor Hernandez. Communication between pc and motes in tinyos. Technical report, Royal Institute of Technology (KTH), July 2011.

[5] Aitor Hernandez. Getting started with tinyos at the automatic control lab. Technical report, Royal Institute of Technology (KTH), July 2011.

[6] Aitor Hernandez. IEEE 802.15.4 implementation for TinyOs based on TKN15.4. GTS implementation. Technical report, TinyOS Contribution, January 2011. URL http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/kth/tkn154-gts/doc/pdf/gts_implementation.pdf.

[7] Aitor Hernandez. Modification of the ieee 802.15.4 implementation. extended gts implementation. Technical report, Royal Institute of Technology (KTH), July 2011.

[8] IEEE Std 802.15.4, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs) , September 2006. URL http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf.

[9] Quanser. Coupled water tanks. instructor manual. Technical report, Quanser.

[10] Quanser. Coupled water tanks. student handout. Technical report.

[11] Quanser. Coupled water tanks. user manual. Technical report, Quanser.

[12] Crossbow Technology. Crossbow telosb. Technical report, San Jose, California. URL http://www.willow.co.uk/TelosB_Datasheet.pdf.

[13] TinyOS. Tinyos tutorials. Technical report. URL http://docs.tinyos.net/tinywiki/index.php/TinyOS_Tutorials.

[14] TinyOS. Tinyos. Technical report, 2010. URL http://www.tinyos.net/.