

ESE 519: Group 10 ISA 100.11a Final Project Report

Azriel Samson, Jonathan Balloch, Vignesh Anantha Subramanian

Contents

Introduction:	2
Project Motivation:	2
Hardware and Operating System.....	2
ISA Data Link Layer Overview:	3
Time Slots.....	3
Channel Hopping:.....	4
Slotted, Slow and Hybrid Channel Hopping:.....	4
Super Frame:.....	5
Links:	6
Basic Communication between nodes.....	6
Synchronization	7
Time Correction:	8
Links, Graphs and Neighbors	9
Introduction:	9
Configuring Links and Graphs:	9
Graph Preferences:	10
Advertisements and Neighbor Discovery	13
Advertisements.....	13
Neighbor Table Report.....	14
Flushing Neighbor Table Entries	14
Packet Loss Analysis.....	14
Future Work:	16
Software Correction Required:	16
Conclusion:.....	17

Introduction:

ISA is a wireless networking technology standard developed by the Industrial Society of Automation. It is intended to provide reliable and secure wireless operation for non-critical monitoring, , supervisory control, open-loop control and closed loop applications. The standard addresses coexistence with other wireless devices anticipated in the industrial workspace. This standard is built upon radios compliant with IEEE Std 802.15.4 (channels 11-26) operating in the 2.4 GHz ISM band, which is available and license-exempt in most countries worldwide.

Robustness in the presence of interference and with non-wireless industrial sensor networks

This standard supports channel hopping to provide a level of immunity against interference from other RF devices operating in the same band, as well as robustness to mitigate multipath interference effects. In addition, this standard facilitates coexistence with other RF systems with the use of selective channel utilization to detect and avoid using occupied channels within the spectrum. Selective channel utilization can also enhance reliability by avoiding the use of channels with consistently poor performance.

The architecture supports wireless systems that span the physical range from a single, small, isolated network, such as might be found in the vicinity of a gas or oil well or a very small machine shop, to integrated systems of many thousands of devices and multiple networks that can cover a multi-square-km plant.

Project Motivation:

The motivation of this project is to produce an open source implementation of ISA 100.11a. The project aims at iteratively developing features of the wireless protocol.

Hardware and Operating System

The project has been implemented using Nano-RK which is a fully preemptive reservation based real time operating system with multi-hop networking support for use in wireless sensor networks. It is implemented on FireFly Sensor Networking Platform which is a low-cost low-power hardware platform for wireless sensor networking. The main Firefly board uses an Atmel ATmega1281 8-bit micro-controller with 8KB of RAM and 128KB of ROM along with Chipcon's CC2420 IEEE 802.15.4 standard-compliant radio transceiver for communication.



Figure 1: Firefly Node

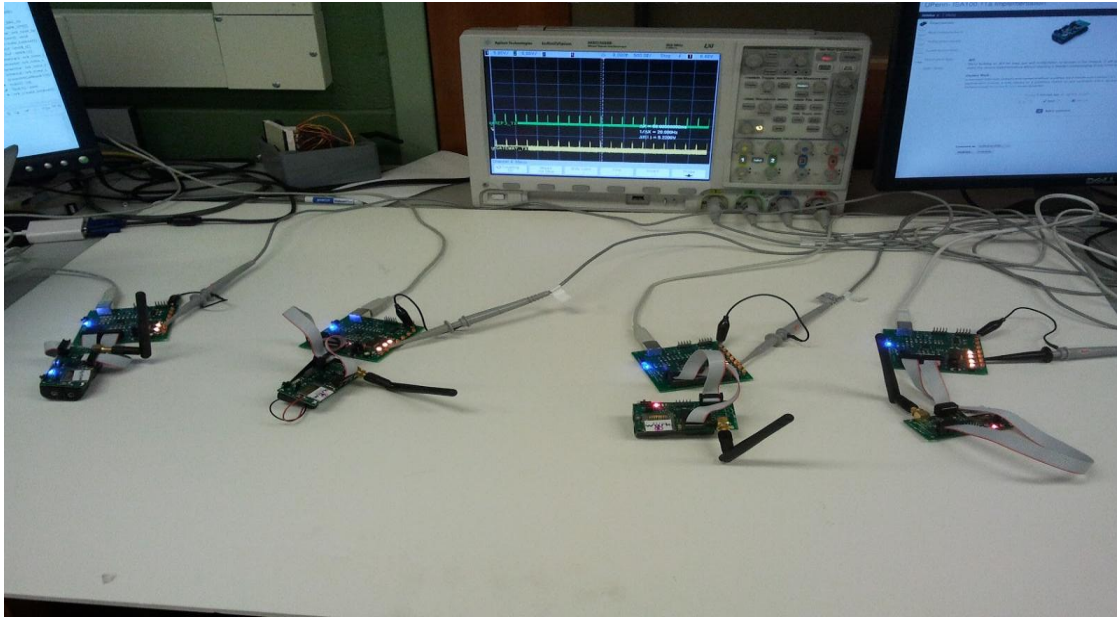


Figure 2: A small network consisting of a gateway node(pictured far left) and three repeater nodes

ISA Data Link Layer Overview:

The data link layer (DL) in this standard is designed with the general goal of constraining the range of build options for a field device, while enabling flexible and innovative system solutions. The DLs building blocks include timeslots, superframes, links, and graphs. The system manager may assemble these building blocks to configure a device in one of three general operational alternatives, slotted channel hopping, slow channel hopping, and hybrid slotted/slow hopping.

Time Slots

A timeslot is a single, non-repeating period of time. The timeslot durations in this standard are configurable to a fixed value such as 10 ms or 12 ms. Once a timeslot duration is selected, all timeslots generally have the same duration and they are re-aligned to a 4 Hz cycle at each 250 ms clock interval.

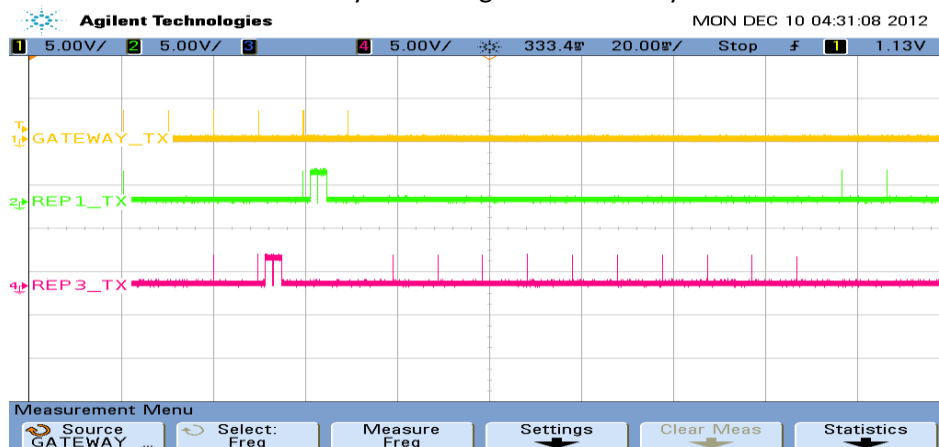


Figure 3: Each single vertical line represents the start of a timeslot for that device

DL communications are intended to be distributed across multiple radio channels(11-26). The system uses channel hopping patterns which provide a specific sequence of channels for communication among collections of devices. Channel hopping begins at a designated offset in the hopping pattern, continuing through the pattern in order until the end, and then repeats the pattern.



At least three channels separate each consecutive hop in pattern 1, resulting in frequency shifts of at least 20 MHz. When retries occur in consecutive hops, they will not encounter or cause interference in the same IEEE 802.11 channel.

In slotted hopping, channel-hopping timeslots of equal duration are used. Each timeslot uses a different radio channel in a hopping pattern. In slotted hopping, each timeslot is intended to accommodate a single transaction including one DPDU and its acknowledgement(s). In slow hopping, a collection of contiguous timeslots is grouped on a single radio channel. Each collection of timeslots is treated as a single slow hopping period; however, as shown in Figure 57, timeslots still underlie slow hopping. Slow-hopping periods are configurable, typically on a scale of about 100 ms–400 ms per hop. Hybrid operation uses slotted hopping and slow-hopping periods in a configured combination.

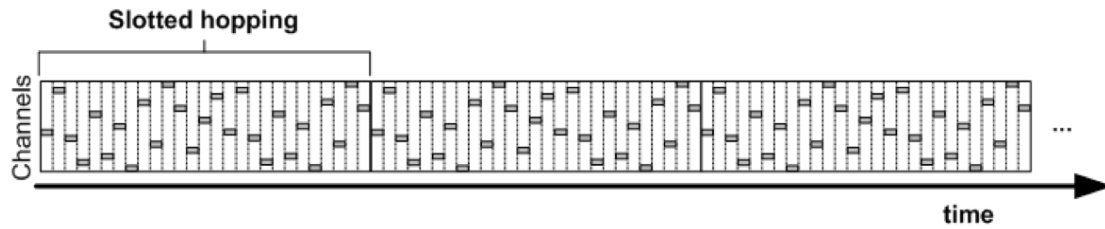


Figure 5: Slotted Hopping

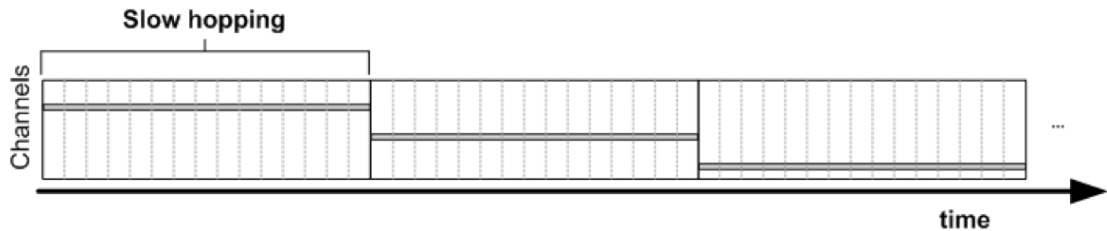


Figure 6: Slow Hopping

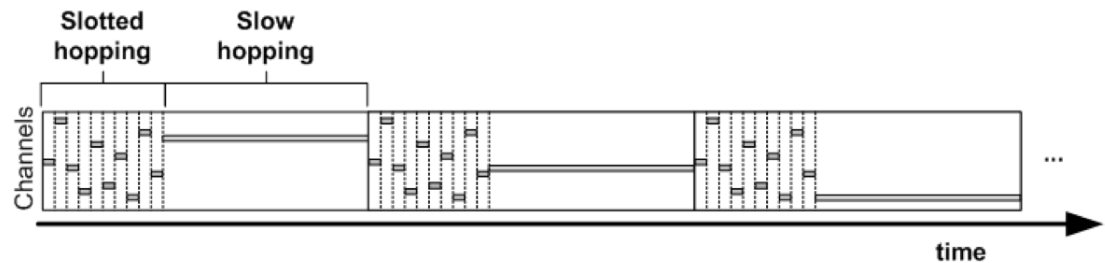


Figure 7: Hybrid Hopping

Super Frame:

A super frame is a collection of timeslots repeating on a cyclic schedule. The number of timeslots in a given super frame determines how frequently each timeslot repeats, thus setting a communication cycle for devices that use the super frame. The super frame also has an associated reference channel hopping pattern.

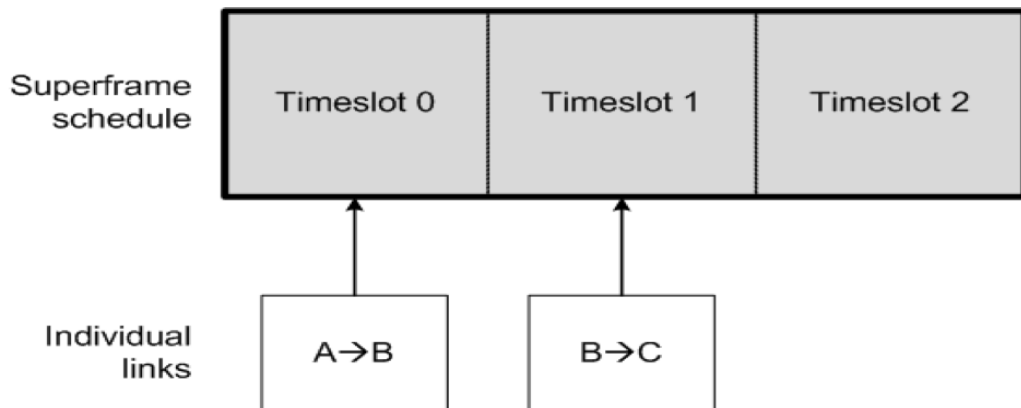


Figure 8: A 3 time slot superframe. This superframe will repeat every 3 timeslots

Links:

Links refer to the use of superframe timeslots for communication between a specific pair of devices. A timeslot is a period of time. A superframe is a cyclic schedule of timeslots. A link describes a specific activity that repeats within a superframes cyclic schedule.

The system manager configures matching sets of links among a collection of devices that communicate with each other. For example, a link on Device A may be configured to transmit DPDU's to Device B on a particular superframe cycle. On the same cycle, Device B should have a link that is configured to listen for an incoming DPDU. These two links are matched in the sense that the system manager has configured these two related operations to occur in tandem.

Basic Communication between nodes

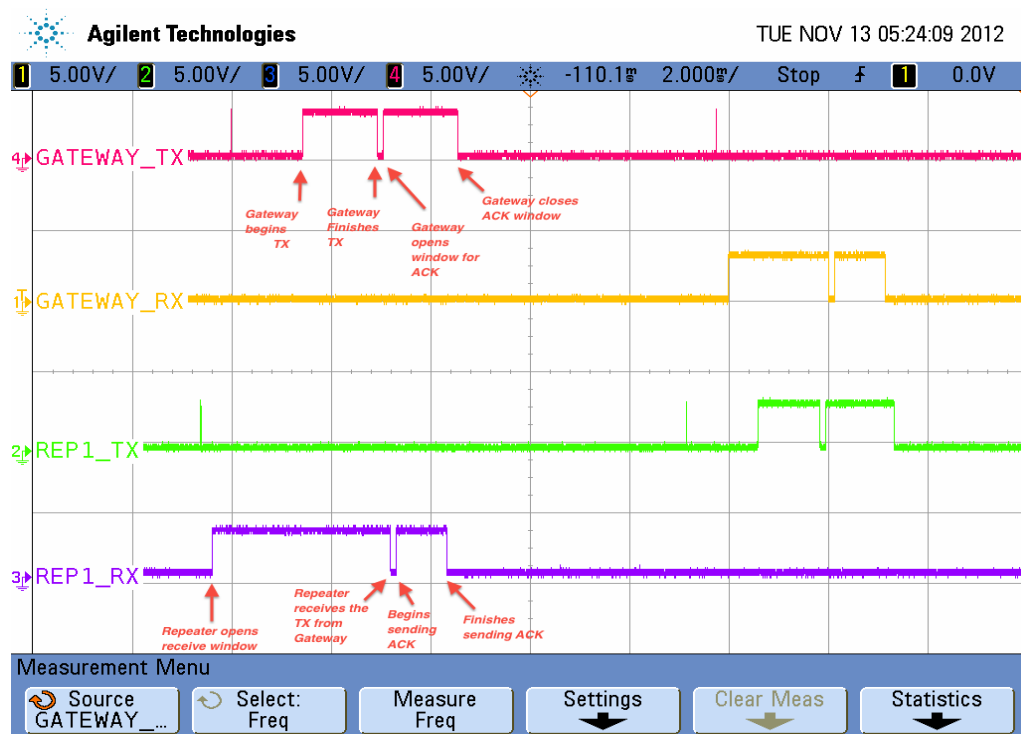


Figure 9: Basic Communication mechanism between two devices

The above diagram shows communication between the Gateway and Repeater 1. The Gateway_TX line and Rep1_TX line show the start of timeslot. This is indicated by the single vertical line. The Gateway waits a while before it starts transmitting. This is to give any receiving device time to open its receiving window. The receiver opens its receiving window immediately at the start of the timeslot. (the repeater here is configured to receive on this timeslot)

After the receiver receives the packet from the gateway, it will send an ACK back. This ACK is sent back only if the repeater successfully receives the packet. After the gateway finishes transmitting, it opens a

window and expects an ACK.

Similarly, in the next instance, Repeater 1 transmits (Green line) and then opens its receive window for an ACK and the Gateway receives (Yellow line) and then sends an ACK back.

Synchronization

In the protocol, there are different ways in which a device can join the network. There is a provisioning and join step which facilitates the process. Provisioning inserts the required configuration data into a device to allow a device to join a specific network. In our implementation, we are programming this required information into the device.

When a device is looking to join a network, the device listens on a pre defined channel for an advertisement from any other device. Using the information received, the device aligns its super frame slots according to its position within global super frame.

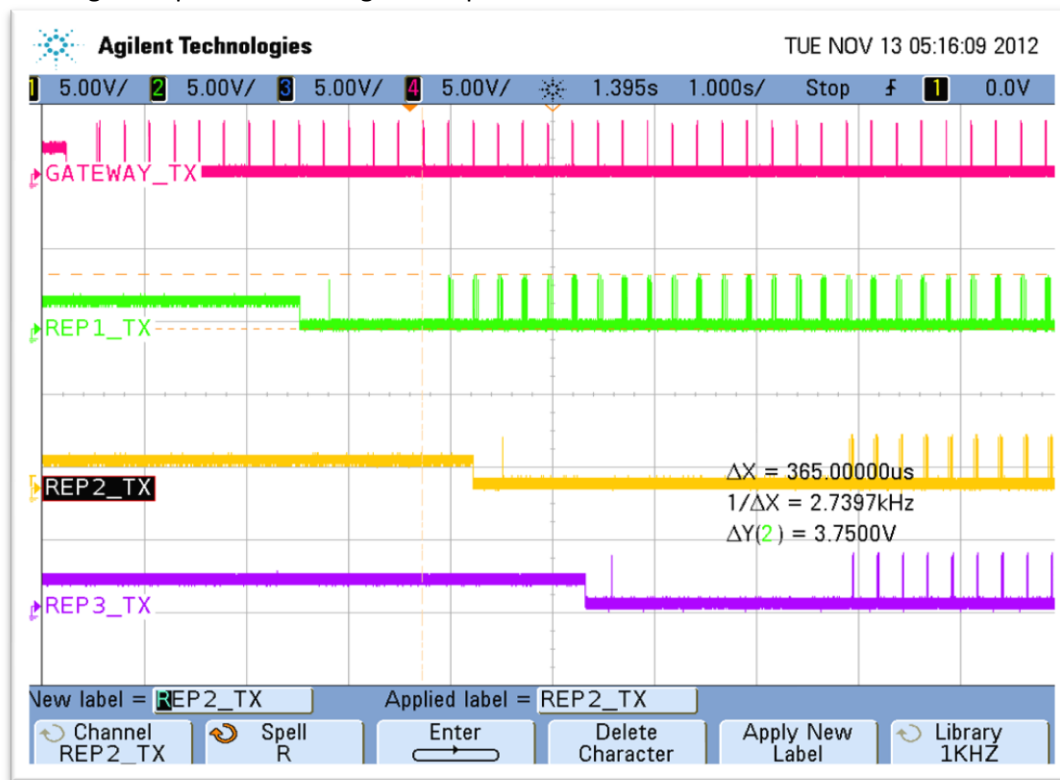


Figure 10: Synchronization

In the image shown, we can see three devices from the time they are switched on, to the time they synchronize to the network. During this time, each of these devices listen on a predefined channel for slot information. Upon receiving this information, each device aligns its slot such that its slotted channel hopping pattern is identical to the one on the other device, thereby letting them communicate.

Time Correction:

Clock updates are propagated through the Data Link network in the course of normal communications within a timeslot or slow hopping period. These building blocks are leveraged by the system manager to arrange the propagation of network time.

Clock sources use the three following mechanisms to propagate clock updates to clock recipients:

1. A clock source sends the network time as a DPDU to its recipients.
2. A clock recipient sends a dummy acknowledges a received DPDU in a timeslot. The time is conveyed by measuring when the clock source detects the start of the DPDU and echoing the result of this measurement in the acknowledgement.
3. A DL clock source acknowledges a DPDU within a slow hopping period. The process is similar to acknowledgement within a timeslot, with the addition of an octet to identify the timeslot uniquely if needed.

We have implemented the 2nd method for time correction. When a device sends a message to it's clock source and has indicated a request for time correction using the appropriate header, the clock source shall send back it's offset in the timeslot, and the device calculates the change in it's time required to match that offset.

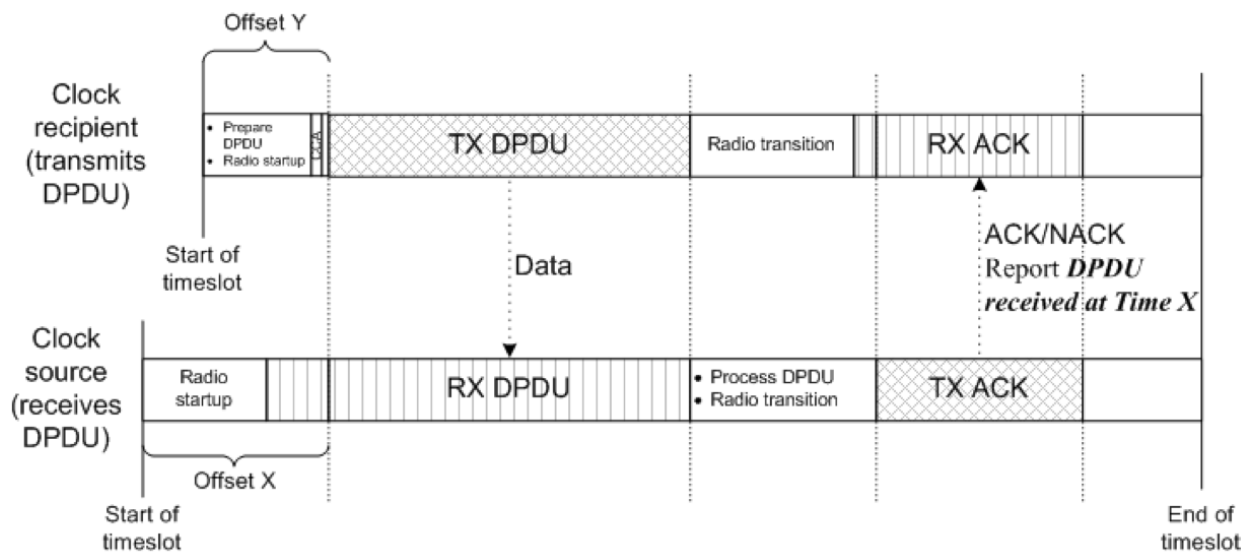


Figure 11: Time Correction on acknowledgement

In accordance with the protocol, time correction occurs when as a result of a transmission between a device and its clock source. A device sends a message to its clock source using the appropriate header, the clock source measures the time period from its wake up until it receives the packet, and sends back this "offset" in the acknowledgement to the device. The device calculates its time correction as the difference between the wake up-to-transmission offset on the device and the wake up-to-reception offset on the clock source.

Links, Graphs and Neighbors

Introduction:

Links are connections between devices. When the system manager defines paths between devices, the devices receive link assignments. A link assignment repeats on a cyclic schedule, through its connection to an underlying superframe. Each link refers to one timeslot or a group of timeslots within a superframe, its type (transmit and/or receive), information about the device's neighbor (the device on the other end of the link), a channel offset from the superframe's underlying hopping pattern, and transmit/receive alternatives.

This standard supports graph routing as well as source routing. A directed graph is a set of directed links that is used for routing DPDUs within a DL subnet. Each directed graph within the DL subnet is identified by a graph ID. In source routing, the originating device designates the hop-by-hop route that a DPDU takes through a DL subnet. Graph routing and source routing may be mixed.

Configuring Links and Graphs:

Now that we have seen what links are, we shall see how links are used to implement the underlying structure of the data link layer. Initially, all devices have exactly two links configured: One to transmit their Clock Source device and one to receive from their clock Source device. All other links have to be configured for each device.

There are essentially 3 ways in which device configuration information is sent: Configure a Link, Configure a Neighbor and Configure a Graph. We will discuss each one of these configurations in detail. To do that, we evaluate how a link is configured and how it is associated with a neighbor and a Graph.

There are many different kinds of links that can be configured according to the standard. We have essentially implemented 3 types:

1. Receive Links: To receive information from another device
2. Transmit Links: To send information to other devices.
3. Advertise Links: To send advertisement packets, which will be discussed later in detail.

if a receive link is to be configured, the slot is simply configured as a receive Link and is not associated to any neighbor or graph. If the incoming Link is a transmit Link, it has to be associated with a Neighbor. Therefore, when a Transmit Link configuration is sent, it is stored and associated with the respective Neighbor data structure. A Link may or may not be associated with a Graph. If a Link is related to a particular graph, the link is associated to that particular graph data structure.

Graph Preferences:

A graph is a set of directed links that is used for routing messages within a DL subnet. Each graph designated by the system manager for routing within a DL subnet is identified by a graph ID. The links associated with each graph are configured by the system manager. A DL subnet may have multiple graphs, some of which may overlap. Each device may have multiple graphs going through it, even to the same neighbors.

When a graph is to be configured, a graph can be associated with up to 3 neighbors with preferences. When a config Graph message is sent, a particular graph ID is associated with up to 3 neighbor structures with the first neighbor having a higher preference over the others.

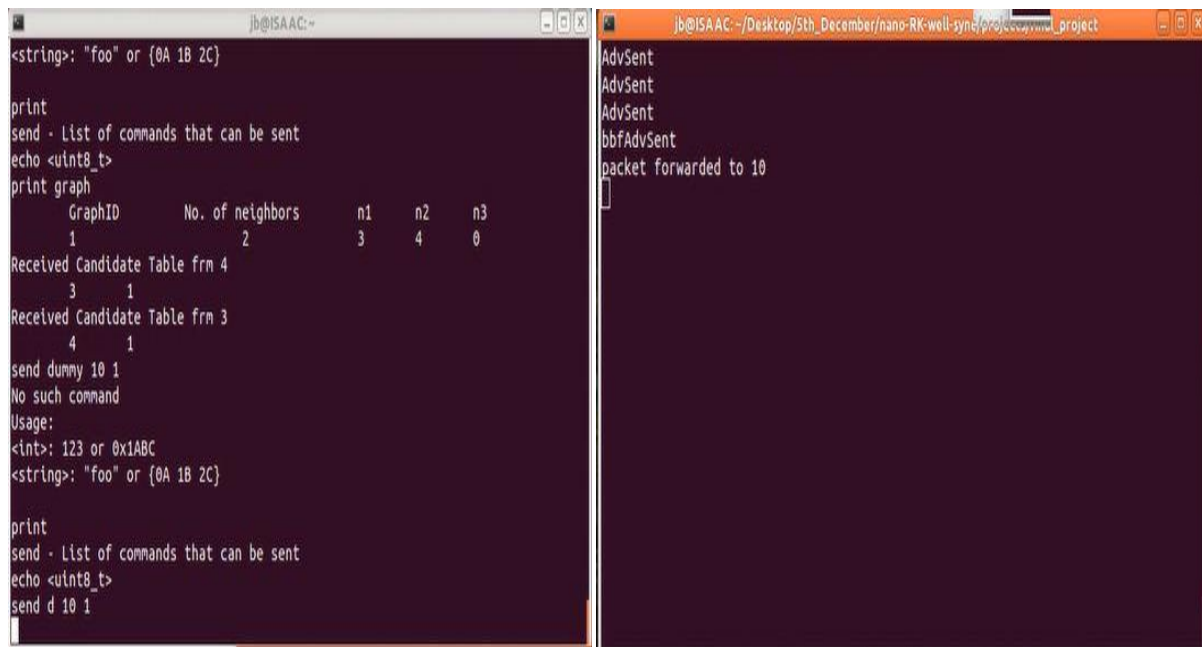
When a message is to be forwarded on a graph, a device will attempt to transmit the message to the neighbor with the first preference on the respective transmit link. If it does not receive an acknowledge, the device will retransmit the message to any one of the other two neighbors present in its link.

The example is displayed below.



```
jb@ISAAC:~  
Received Candidate Table frm 4  
  3      1  
Received Candidate Table frm 3  
  4      1  
print graph  
No such command  
Usage:  
<int>: 123 or 0x1ABC  
<string>: "foo" or {0A 1B 2C}  
print  
send - List of commands that can be sent  
echo <uint8_t>  
print graph  
  GraphID      No. of neighbors      n1      n2      n3  
  1              2              3       4       0
```

Figure 12: List of Neighbors for Gateway



```
jb@ISAAC: ~  
<string>: "foo" or {0A 1B 2C}  
  
print  
send - List of commands that can be sent  
echo <uint8_t>  
print graph  
  GraphID      No. of neighbors      n1      n2      n3  
  1              2              3       4       0  
Received Candidate Table frm 4  
  3          1  
Received Candidate Table frm 3  
  4          1  
send dummy 10 1  
No such command  
Usage:  
<int>: 123 or 0x1ABC  
<string>: "foo" or {0A 1B 2C}  
  
print  
send - List of commands that can be sent  
echo <uint8_t>  
send d 10 1  
  
jb@ISAAC: ~/Desktop/5th_December/nano-RK-well-syns/project/terminal_project  
AdvSent  
AdvSent  
AdvSent  
bbfAdvSent  
packet forwarded to 10  
[
```

Figure 13: Gateway(left) sends a message over Graph 1. It is successfully received and transmitted by 3 which is the gateway's first preference

```
print
send - List of commands that can be sent
echo <uint8_t>
print graph
      GraphID      No. of neighbors      n1      n2      n3
      1            2            3      4      0
Received Candidate Table frm 4
      3            1
Received Candidate Table frm 3
      4            1
send dummy 10 1
No such command
Usage:
<int>: 123 or 0x1ABC
<string>: "foo" or {0A 1B 2C}

print
send - List of commands that can be sent
echo <uint8_t>
send d 10 1
send dummy 10 1

AdvSent
bAdvSent
AdvSent
AdvSent
packet forwarded to 10

AdvSent
AdvSent
AdvSent
bAdvSent
packet forwarded to 10

+-----+
| Cannot open /dev/ttyUSB2! |
+-----+
```

Figure 14: Once Repeater 3 is down(pictured top right), the gateway then forwards the same message to repeater 4, which is second on its preference list

Advertisements and Neighbor Discovery

Advertisements

Wireless subnets compliant with this standard are detected by the DL. A new device may hear advertisements from neighboring devices that have already joined a DL subnet of interest. This is called neighbor discovery.

Devices discover DL subnets through DL advertisements received from one or more advertising devices that periodically announce their presence. After joining the network, the device receives advertisements from a series of neighbors in a DL subnet and builds internally a history of candidate neighbors that are in range. This list of candidates is reported to the system manager through the attribute `dlmo.Candidates`. The system manager uses this information to determine how the device fits into the network topology and thereby establishes communication relationships between a device and its neighbors.

Advertising devices may be discovered using passive scanning, active scanning, or a combination of passive and active scanning.

Our network uses passive scanning for a device to identify its neighbors. In passive scanning, the DL periodically listens for advertisements on a series of channels. Generally, a battery-powered passive scanning device will listen frequently when first powered on. To support passive scanning by new devices, active scanning hosts may also be configured to transmit advertisements periodically. If a DL subnet is not discovered quickly, the device may preserve battery life by scanning less frequently, on fewer channels, and/or for shorter periods of time.

When a device receives an advertisement message from a neighbor, it adds this neighbor to its neighbor table. Each device builds its neighbor table over time.

```
Added Neighbor at Index 0
Added Neighbor at Index 1
Added Neighbor at Index 2
Added graph at Index 0
Added link at Index 0
Added link at Index 1
Added link at Index 2
Added link at Index 3
Added link at Index 4
Added link at Index 5
Added link at Index 6
Added link at Index 7
Task1 PID=2
Gateway
Isa start!
Added 2 to Candidate table at 0
echo 1
Received Candidate Table frn 2
3 1
Received Candidate Table frn 3
1 2 4
Received Candidate Table frn 4
2 3 1
n

Added graph at Index 0
Added link at Index 0
Added link at Index 1
Added link at Index 2
Added link at Index 3
Added link at Index 4
Added link at Index 5
Added link at Index 6
Added link at Index 7
Added link at Index 8
Added link at Index 9
Added link at Index 10
Added link at Index 11
Added link at Index 12
Create Done
Task1 PID=2
Isa start!
Added 2 to Candidate table at 0
Added 3 to Candidate table at 1
Added 1 to Candidate table at 2
bChannel 21
bChannel 14
AdvSent
```

Figure 15: Repeater 3 sends a Neighbor Discovery Report Message back to Gateway with a list of neighbors

Neighbor Table Report

Each device may be configured to periodically transmit their candidate neighbor table, along with other neighbor diagnostics, to the system manager.

The dlmo.Candidates attribute is used to provide the system manager with a list of candidate neighbors. The DL autonomously populates this attribute as it receives advertisements from a number of candidate neighbors. This attribute is then forwarded to the system manager so that routing decisions can be made. The system manager may reset dlmo.Candidates.N=0, thus signaling to the device to clear its history of received advertisements and resume the neighbor discovery process.

Flushing Neighbor Table Entries

To detect a dropped neighbor, a node has to be manually told by the gateway to flush its candidate table information and rebuild it.

```
Added Neighbor at Index 2
Added graph at Index 0
Added link at Index 0
Added link at Index 1
Added link at Index 2
Added link at Index 3
Added link at Index 4
Added link at Index 5
Added link at Index 6
Added link at Index 7
Task1 PID=2
Gateway
lsa start!
Added 2 to Candidate table at 0
echo 1
Received Candidate Table frn 2
  3 1
nReceived Candidate Table frn 3
  1 2 4
nReceived Candidate Table frn 4
  2 3 1
n send flush 3 0
send flush 4 0

Added link at Index 3
Added link at Index 4
Added link at Index 5
Added link at Index 6
Added link at Index 7
Added link at Index 8
Added link at Index 9
Added link at Index 10
Added link at Index 11
Added link at Index 12
Create Done
Task1 PID=2
lsa start!
Added 2 to Candidate table at 0
Added 3 to Candidate table at 1
Added 1 to Candidate table at 2
bChannel 21
bChannel 14
AdvSent
Received flush candidate table 1
Added 1 to Candidate table at 0
Added 2 to Candidate table at 1
Added 3 to Candidate table at 2
```

Figure 16: Repeater 4 flushes and rebuilds it's candidate table from advertisements

Packet Loss Analysis

We tested packet forwarding for 4 nodes over a period of about 12 hours. The set up, results and analysis are presented below.

Configuration : Gateway <--> Repeater 1 <--> Repeater 2 <--> Repeater 3

Gateway	Repeater 1	Repeater 2	Repeater 3
TX : 2	RX : 2		
RX : 3	TX : 3		
	TX : 5	RX : 5	
	RX : 7	TX : 7	
		TX : 8	RX : 8
		RX : 9	TX : 9

25 time-slots of 10ms each are used with a the standard ISA100.11a channel hopping pattern. The slot used to transmit or receive is given beside TX or RX above. We print the packet loss statistics every 1000 transmits (this also includes ACK's that are transmitted after a packet is received).

Time correction information is present in the ACK sent from the node on the left to the node on the right. Repeater 1 is time corrected on slot 3. The time correction information is present in the ACK sent by the gateway. Similarly time correction information is present on TX slot 7 for Repeater 2 and Tx slot 9 for Repeater 3.

The packet received from the gateway is forwarded on TX : 5 by Repeater 1 and TX : 8 by Repeater 2. towards Repeater 3.

Repeater 1 and Repeater 2 also transmit a self generated packet on TX : 3 and TX : 7 respectively. The waveform is shown below:



Figure 17: Testing with 4 nodes

We monitored packet loss for approximately 9 hours. The file below is a link to statistics printed by Repeater 1.

Packet loss count is incremented when a packet is not received in the receive slot or an ACK is not received back after a transmit.

Packets transmitted : 579987

Packets received : 579969

Total lost : 7 + 18

18 packets were lost during startup on purpose
 Percentage loss : 0.0012

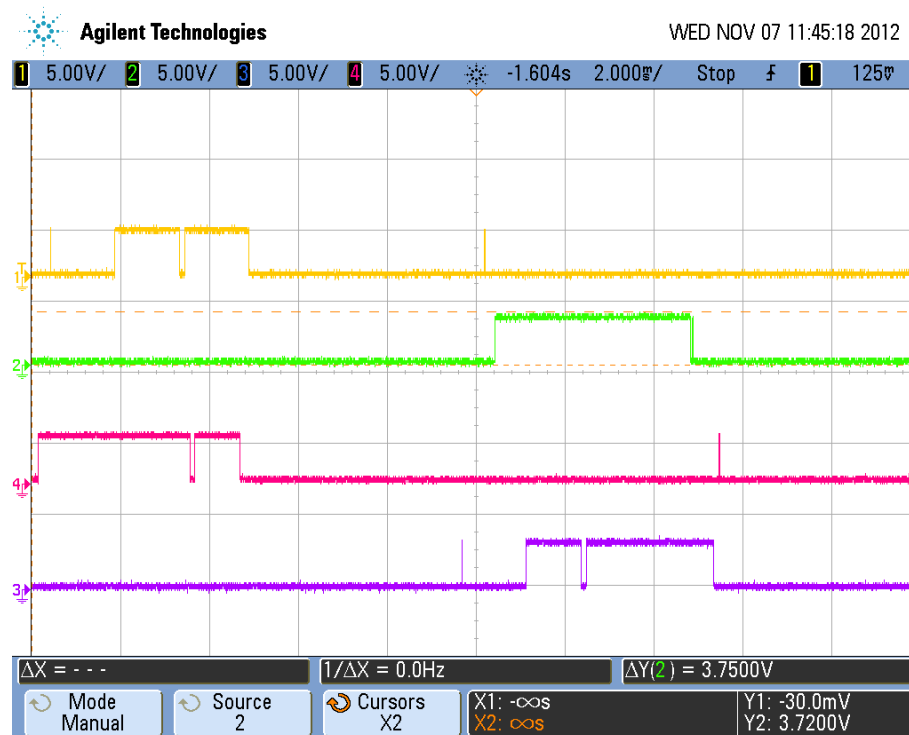


Figure 18: Packet Loss Scenario

Why doesn't the gateway transmit an ACK (green line) after it receives a packet from the Receiver (Purple line) ? We see that even though the gateway opens its receive window before the Receiver starts sending, it does not send an ACK back. In this case, we noticed that the gateway does not receive the SFD of the Receiver.

Possible causes : Packet loss through medium, Receiver did not send because of Collision Avoidance

Future Work:

Software Correction Required:

1. Time correction is reliant on Nano-RK's response

In the function `isa_rx()` in `isa.c`, the following line is used to calculate the offset for time correction in one case. The 1 works well in most cases and gives small values for time correction and keeps nodes well in sync. However, on occasional cases, when nodes were configured with very few links, adding 2 to the offset worked better.

offsetSec = (time_correction/7325)+1;

2. Destination Address and PAN should be included in the appropriate header. At the moment, this information is sent in the payload.

Conclusion:

The ISA100.11a protocol represents a comprehensive industrial wireless approach that serves a broad, facility-wide wireless network for both process and discrete manufacturing. In this project, we have implemented a significant portion of the data link layer.

Our next step is to implement this protocol on a platform to demonstrate it's working, preferably in a process control setting.