Team 37

Open-Source Prototyping of Advanced Wireless System for Smart Agriculture and Connected Rural Communities

TEAM MEMBERS - ROLES:

ZEQUN WANG - MEETING SCRIBE

JIAWEI DENG - CHIEF ENGINEER

YULIN SONG - TEST ENGINEER

SHAOHANG HU - TEST ENGINEER

ZHENGWEI SU - REPORT MANAGER

DYLAN SHARP - MEETING FACILITATOR

CLIENT:

IOWA STATE UNIVERSITY

ADVISORS:

PROF. HONGWEI ZHANG

DR. MATTHIAS SANDER-FRIGAU

Problem & Need Statement:

Rural regions are home to many industries such as agriculture, renewable energy and manufacturing, and they are major sources of food and energy for our society. But 39% of the rural US lacks broadband access, and most farms are not connected at all. Through this project, we are going to develop and prototype advanced wireless solutions for smart agriculture and connected rural communities.

Solution:

We are going to use cutting-edge TV white space (TVWS) wireless platforms and advanced wireless algorithms. At the end of this project, we hope to accomplish a spiral evaluation and refinement of the aforementioned novel 5G wireless solution for smart agriculture and connected rural communities.

Functional Requirements

- Protocol must store a buffer of nodes
- Each node in the exclusive region must update every 1 s
- Every node must monitor the signal strength
- Every node must store their local signal map

Non-Functional Requirements

- Package loss rate should below 10%
- Link reliability should be higher than 90%
- Real time latency should be below 50 ms
- Network throughput should be higher than 10 Mbs

Operating Environment:

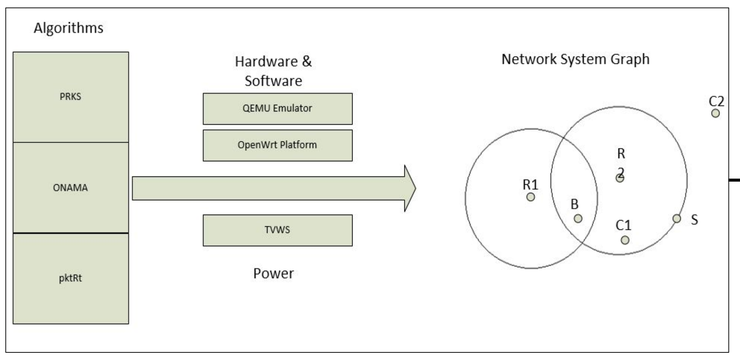
- Based on OpenWRT platform, powered by TVWS Spectrum

Intended Users:

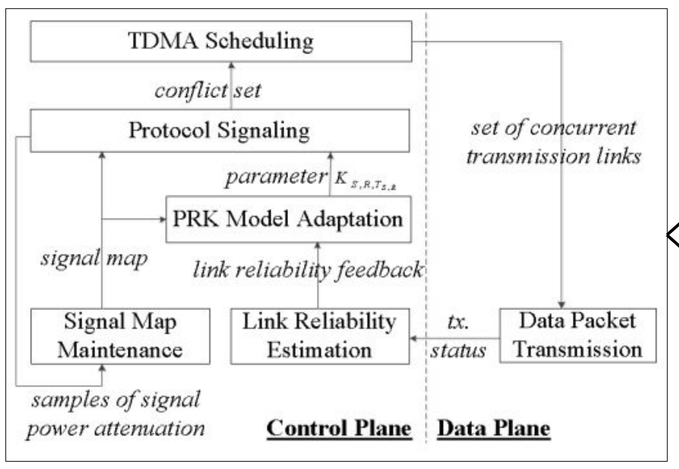
engineers and developers who work in the industry of wireless communications

Uses:

The product will benefit customers/users in the smart agriculture industry and rural communities. In the US, tons of industries are based in rural areas, as well as agriculture. A big number of them especially farms have a lack of broadband connectivity or have no reception at all. So if these regions can have a reliable wireless network setup, there will be a big progress towards smart industry/agriculture.



Concept Sketch shows what our project will create. Based on OpenWrt platform and using QEMU as an emulator, combined with the power supply of TWWS spectrum. We are building a network system of nodes that are transmitting information with each nearby nodes. Each node represented a device in the network, they can broadcast itself and transmit data within its sensing range to other nodes. After other nodes receive the data they will do selfcheck to see whether they are in the same Exclusion Region or nor. Then they will repeat the broadcast function again in their sensing range until all of the nodes have been updated in the signal map.



Block Diagram displays the different modules within the PRKS algorithm scheduling and what relational arguments are passed from one module to another.

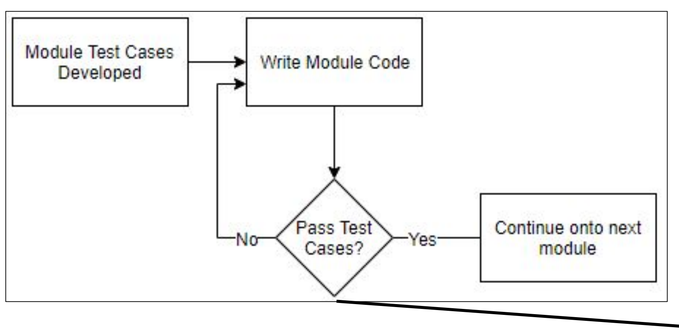
Functional Module Design is based on the PRKS Scheduling. We implement our pseudo code corresponds to each sections in the PRKS Scheduling.

```
/*
This function describes how nodes rebroadcast itself and deliver the value of K within its sensing range.
And how other nodes react after receiving the signal.
*/

Main(exclusive_region E)
{

    While(true)
    {
        For Receiver R in exclusive_region E
        {
            Broadcast(R, K)
        }
    }
}

//node broadcast in their carrying sense range
broadcast(node R, K)
{
    For every node C in sensing range of node R
    {
        If P(C, R) >= P(S, R) / K
            C.broadcast(C, K, signal_map)
    }
}
```



Testing environment: Linux

Tool: Linux strace tool

Programming Language: C
Library: nl80211, mac80211, cfg80211, etc.
Development tools: OpenWrt, QEMU
Development environment: Linux

Testing Strategy (unit testing):

- Develop test cases for each module
- Implementation for each module will be developed
- After Implementation has been developed then the test cases will serve as acceptance tests to determine whether the module behaves as it should or not.
  - -If the tests fail, then go back and develop and try again.
  - -If tests pass, then continue to the next module.