|  |
| --- |
|  |
| Wireless Protocol Design Documentation |
|  |
|  |
| **Andrew Burian Chris Holisky Shane Spoor Ashley Tham** |
| **11/13/2013** |

Table of Contents

**Task Assignments1**

**State Chart Diagram2**

**Pseudo Code3**

**Gantt Chart9**

|  |
| --- |
|  |

**Wireless protocol project task assignments:**

Documentation:  
Gantt chart: Shane Spoor/Ashley Tham  
State Chart: Andrew Burian  
Pseudocode: Chris Holisky/Andrew Burian  
Tasks Document: Chris Holisky  
Compile documentation: Chris Holisky

Coding:  
Connect to Comm Port:  
Start Protocol Engine:  
File Loader Thread Start:  
Load file to output queue:

Clear Queue:

Reading Thread Start: Andrew Burian

Data Received: Andrew Burian  
Validate Data: Chris Holisky  
Trigger event:

Idle:

TxAck(1): Andrew Burian  
Wait for Ack(1):

Create Packet: Andrew Burian

Wait for Ack(2):

Done Sending:

Wait for data:

TxAck(2):   
Do Nothing:

File Write Start:

Write to File:

**Wireless protocol project state diagram:**



**Wireless protocol project base pseudo code:**

Connect to Comm Port

{

Open Comm config dialogue

check valid

set handle

}

Start Protocol Engine

{

Start Input reading thread

Start Protocol Driver thread

}

-----------------------------------------------------------

File Loader Thread Start

{

Open file selection dialogue

Verify file exists

}

Load file to output queue

{

Load the file byte by byte into the output cue

trigger the Output available event

if program end event

go to [Clear Queue]

}

Clear Queue

{

Empty the output queue

}

-----------------------------------------------------------

Reading Thread Start [Andrew]

{

Start listening to comm port for input

}

Data Received [Andrew]

{

if SYN

get next byte

if Ctrl

go to trigger event (ctrl)

if SOT

get next 1020 bytes

if not duplicate

go to validate data

go to trigger event (dup)

}

Validate Data [Chris]

{

Check crc section of data agaisnt the data packet

return true or false for valid data

}

Trigger event [Andrew]

{

If ctrl

trigger corresponding ctrl event

If dup

trigger data recieved event

if data-invalid

trigger baddata event

if data-valid

trigger data event

read data to input queue

return to Thread Start

}

-----------------------------------------------------------

Idle

{

wait for (Output available | ENQ | End of Program) events

if Output

Send ENQ

go to Wait for Ack(1)

if ENQ

go to TxAck(1)

if End of Program

end thread

}

TxAck(1) [Andrew]

{

send Ack

go to wait for data

}

Wait for Ack(1)

{

wait for Ack Event for TIMEOUT

if ack

go to Create Packet(new)

if timeout

return to idle

}

Create Packet [Andrew]

{

if new

add SYN

add correct SOT char

add data from queue

(pad if nessesary)

add CRC [Chris]

send packet

go to Wait for Ack(2)

if resend

send last packet

}

Wait for Ack(2)

{

wait for ack for TIMEOUT

if ack

if more data

Create Packet (new)

if no more data

go to Done Sending

if timeout

if attempted 5 times

give up (return to idle)

else

go to Create Packet (resend)

}

Done Sending

{

Send EOT

return to idle

}

Wait for data

{

wait for (data | bad data | EOT) for TIMEOUT

if data

go to txAck(2)

if bad data

do nothing

if EOT

return to idle

if TIMEOUT

return to idle

}

TxAck(2)

{

send ack

return to wait for data

}

Do Nothing

{

return to wait for data

// alt: send NAK then wait for data

}

-----------------------------------------------------------

File Write Start Start

{

wait for (input available | End of Program)

if input available

go to write to file

if end of program

end thread

}

Write to File

{

write input queue to file until queue empty

if EOF mid queue

start new file

continue

return to thread start

}