

# Appendix 4.

## Software

4.1 Protocol Specification

4.2 FreeRTOS and microcontroller test report

4.3 Communication protocol test report

**reel.SMRT**  
**Final report**  
**17 January 2010**





## **Appendix 4.1 Protocol specification rev 3**

The reel.SMRT protocol is designed to maintain communication and data transmission of reel.SMRT payload, FISH and ground station in reel.SMRT project. The protocol consists of 2 links: ground station – reel.SMRT payload and reel.SMRT payload and FISH. Both of the links have to initiate the connection, detect the connection loss and recovery the connection in the case of connection loss. The data have to be transmitted from FISH to reel.SMRT payload then to ground station with lossless schema.

### ***Revision update***

rev 1 : initial version, setup most of command and acknowledgement procedure

rev 2 : modified during first implementation on the evaluation board, a lot more of clarification.

rev 2.5 : modified during launch campaign, add more command and some parameter

rev 3 : as built specification

### ***Abbreviations***

TCP – Transmission Control Protocol

UART – Universal Asynchronous Receiver/Transmitter

RF – Radio Module

IP – Internet Protocol

### ***Protocol Definition***

The connection and data transmission are different depend on operation status which can be divided in 2 modes in reel.SMRT payload and FISH. The ground station has only one mode which is waiting for data or heart beat. The operations in each mode are described here.

### **Mode**

Reel.SMRT payload and FISH can be operated in 2 modes:

- Normal mode
  - o Wait for command to change mode
  - o Heart beat packet has been sent to maintain the connection checking
- Data mode
  - o Data is sent and data packets are also checking the connection status

Note – Data mode is changed to normal mode automatically when there is no data packet left to send.

### **Link 1: Ground Station (GS) – (E-link) – reel.SMRT payload**

- Using TCP internet protocol via E-link of the balloon in client-server architecture
- Reel.SMRT payload is client
- Ground station is server

- reel.SMRT payload tries to init the connection every estimated round trip timeout , if no connection is established
- When connection is established, reel.SMRT payload sent heart beat packet to ground station. Ground station do not send acknowledgement to reel.SMRT payload because the TCP protocol have already provided acknowledgement mechanism. The ground station computer operating system has been modified the kernel to disable the delayed ACK function to increase the data rate that has been limited by uIP. uIP is a network module implemented in microcontroller.
- Ground station can sent command to reel.SMRT payload, reel.SMRT payload has to reply back with the result status of the command
- When the ground station sends the command to start the experiment, reel.SMRT payload sends the command to the FISH and determines the status. Then reel.SMRT payload replies the result to ground station. If the start experiment is success, reel.SMRT payload changes its mode to data mode. The example can be seen in example section.
- reel.SMRT payload sends data packet that it got from FISH to ground station. The packet has to have correct length and structure.

#### Connection loss detection

Ground station side – wait for certain time, if there is no packet from reel.SMRT payload, consider connection loss

reel.SMRT payload side – if there is no acknowledgement in TCP protocol process, consider connection loss

### **Link 2: reel.SMRT payload – (RF module) – FISH**

- The RF module is connected to the UART of microcontroller, so no connection state is required. The data can be sent directly to the UART. The RF module has no packet control feature, so the protocol has to handle the data loss itself.
- FISH sends heart beat packet every 1 second
- reel.SMRT payload sends acknowledgement packet to the heart beat packet
- reel.SMRT payload can sent command to FISH, FISH has to reply back with the result status of the command
- When the reel.SMRT payload receives the command to start the experiment from ground station, reel.SMRT payload sends the command to the FISH to change it to data mode. FISH reply with the status, reel.SMRT payload sends result back to ground station. Then reel.SMRT payload starts the experiment. The example can be seen in example section.
- FISH sent data packet in data mode
- reel.SMRT payload sends acknowledgement packet to the data packet if length and packet structure are correct.

#### Connection loss detection

reel.SMRT payload side – wait for certain time, if there is no packet from FISH, consider connection loss

FISH side – if there is no acknowledgement in from reel.SMRT payload for certain time, consider connection loss

### Example operation

1. Drop command from ground station to reel.SMRT payload.
2. reel.SMRT payload sends change to data mode command to FISH, if fail go to number 4.
3. FISH send status to reel.SMRT payload and FISH change to drop mode. if fail go to number 4.
4. reel.SMRT payload replies to drop command with status to ground station
  - o If reel.SMRT payload cannot send the status packet to ground station, try to send again until successful.
  - o If it cannot send for certain time, the connection loss is considered. The packet will be sent when the connection is established.
5. If everything success, reel.SMRT payload changes to drop mode and starts drop

### Action when link failure

#### Ground station

- Wait for client to connect again

#### reel.SMRT payload(Ground station side)

- Reconnect the connection until the connection is established.
- When connection is established, send packet that has not been seen again. If no packet left, start send heart beat

#### reel.SMRT payload (FISH side)

- No action will be taken

#### FISH

- No action will be taken

### **Packet structure**

#### **Ground station → reel.SMRT payload**

#### Command

1            1 byte                    1 byte                    2 bytes                    1 byte

C(67)	Packet Number	Command Number	Parameter	LF (0x10)
-------	---------------	----------------	-----------	-----------

\*no checksum because TCP already do it.

*Packet Number (unsigned char)* – running number of packet

*Command Number (unsigned char)* – number represent command

48 – (0 in ascii) Start slow reel ,

Parameter

- round - How long to reel down and up e.g. 2 round for 2 lap of hall sensor detect

49 – (1 in ascii) Start drop ,

Parameter

- Time(ms) - how much time to free fall + additional drop time e.g. 1000 ms

– (D in ascii) Enter Data mode

Parameter

- Time(ms) - how much time to be in Data mode e.g. 1000 ms

(N in ascii) Enter Normal mode

\* Not implemented

(M in ascii) Turn reel motor for x \* 10 degree

Parameter

- x - (signed int) turn in 10 degree, can be plus or minus integer.

(P in ascii) Turn reel up until proximity sensor detect

Parameter – max reel up rounds

(S in ascii) Set reel turning speed in to x digital number

Parameter

- x – (unsigned int) designed turning speed in digital number from 0..1023

(I in ascii) Turn reel into initial position ( 0 degree )

\* Not implemented

(O in ascii) Open bail (no implementation at the moment)

\* Not implemented

(C in ascii) Close bail (no implementation at the moment)

\* Not implemented

(T in ascii) Synchronize time between ReelSMRT payload and FISH

\* Implemented but no use in the flight

(L in ascii) Reel line guide for x lap

Parameter

- x - (signed int) turn in lap, can be plus or minus integer.

(H in ascii) Heating mode setting / Heat specific heater

\* Not implemented in flight version. This feature is coded after the interference test.

Parameter

- mode – first 3 bits of the parameter field
  - 000 – Turn off all heater on reel.smrt Payload
  - 001 – Automatically process on heating on reel.smrt Payload
    - Consider control segment
  - 010 – Turn on heater specified by control bit on reel.smrt Payload
    - Consider force control segment
  - 011 – Turn on all heater on reel.smrt Payload

- 100 – Turn off all heater on FISH
- 101 – Automatically process on heating on FISH
- 110 – Turn on heater specified by control bit on FISH
- 111 – Turn on all heater on FISH
- Force Control – next 5 bits of the parameter field each bit corresponded to each heater all over the experiment ( 0 for turn off, 1 for turn on)
- Control – last 5 bits of the parameter field each bit corresponded to each heater all over the experiment ( 0 for turn off, 1 for turn on automatic control system)

3 bits	5 bit	3 bit	5 bit
mode	Force Control	No use	Control

- (X or x in ascii) Emergency Stop ,  
No parameter (just ignore the parameter field)
- (Y in ascii) Change time delay for predefine additional drop time,  
- Parameter – new time delay in ms (default is 300 ms)
- (W in ascii) Switch direction  
- Parameter –  
0 – switch can reel for both up and down  
1 – switch can only reel up

### reel.SMRT payload → Ground station

#### Reply to command

1      1                                  1 byte                                  1 byte

C	Packet Number	result	LF (0x10)
---	---------------	--------	-----------

\*no checksum because TCP already do it.

*Packet Number (unsigned char)* – packet number of command packet

*Result (1-byte)* – result status to command

- 0, if command cannot execute successfully
- 1, if command execute successfully
- >1, error with error code (To be defined)

#### Reply to time sync command

1      1                                  4 byte                                  1 byte

T	Packet Number	Time diff	LF (0x10)
---	---------------	-----------	-----------

- *Packet Number (unsigned char)* – packet number of command packet

- *Time diff (long)* – time different between reel.smrt payload and FISH in ms = payload\_time-fish\_time

Data packet of reel.SMRT payload

1                    1                                    5 bytes                                    1 byte

M(77)	Packet Number	Data	LF (0x10)
-------	---------------	------	-----------

*Packet Number (unsigned char)* – running number of packet

*Data (5 byte)* – data of the reel.SMRT payload, divided into segment

Data	Type	Size	Description
Timestamp	1 x unsigned long	4 bytes	In ms from system start
Hall Sensor	1 x unsigned short	1 bytes	Each bit (0/1) show status of each hall sensor  * hall sensor data is not implemented to put the data into this field. This is just blank data.
Total		5 bytes	

Heart beat of reel.SMRT payload

1                    1                                    34 byte                                    1 byte

H(72)	Packet Number	Data	LF (0x10)
-------	---------------	------	-----------

*Packet Number (unsigned char)* – running number of packet

*Data (59 byte)* – data of the reel.SMRT payload, divided into segment

Data	Type	Size	Description
Timestamp	1 x unsigned long	4 bytes	In ms from system start
Hall Sensor	1 x byte	1 byte	Each bit (0/1) show status of each hall sensor  * hall sensor data is not implemented to put the data into this field. This is just blank data.
GPS			
- Time	unsigned long	4 bytes	HHMMSS
- Latitude	long	4 bytes	in micro degree (10e-6)
- Longitude	long	4 bytes	in micro degree (10e-6)
- Altitude	short	2 bytes	in m
- Course angle	short	2 bytes	in degree
- Course vel	short	2 bytes	in mm/s

			* GPS is not implemented. This is just blank data.
Temperature	6 x byte	6 bytes	In degree
Battery voltage	5 x byte	5 bytes	Digital number * Voltage measurement is not implemented. This is just blank data.
		34 bytes	

Data packet of FISH

1      1                                  22 byte                                  1 byte

B/F(70)	Packet Number	Data	LF (0x10)
---------	---------------	------	-----------

\* when get the packet from FISH, the reel.SMRT payload puts data into the memory for send to ground station

*Packet Number (unsigned char)* – running number of packet

*Data (22-byte)* – data of the FISH, divided into segment (the same as data from FISH)

Debug message

1                                  1 byte                                  n byte                                  1 byte

D	Number of byte	Message	LF (0x10)
---	----------------	---------	-----------

**reel.SMRT payload → FISH**

Acknowledgement to Heart beat

1 byte

A(65)
-------

Acknowledgement to Data

1                                  1 byte                                  1 byte                                  1 byte

F(70)	Packet Number	checksum	LF (0x10)
-------	---------------	----------	-----------

*Packet Number (unsigned char)* – running number of packet

Command

1                                  1 byte                                  1 byte                                  2 byte                                  1 byte                                  1 byte

C(67)	Packet Number	Command Number	Parameter	checksum	LF (0x10)
-------	---------------	----------------	-----------	----------	-----------

*Packet Number (unsigned char)* – running number of packet

*Command Number (unsigned char)* – number represent command

68 – (D in ascii) change to data mode

(N in ascii) Enter Normal mode  
No parameter

(T in ascii) Query the time of the fish

*Checksum( 1 byte)* – checksum, calculated by all character before checksum XOR with each other

### **FISH → reel.SMRT payload**

#### Heartbeat(B)/Data(F) of FISH

1	1	26 byte	1 byte	1 byte
B/F	Packet Number	Data	checksum	LF (0x10)

*Packet Number (unsigned char)* – running number of packet

*Data (n-byte)* – data of the FISH, divided into segment

Data	Type	Size	Description
Timestamp	1 x long	4 bytes	In ms from system start
Accelerometers	3 x long	12 bytes	Digital number
Gyroscopes	3 x short	6 bytes	Digital number
Accelerometers Temperature	1 x long	4 bytes	Digital number
	Total	26 bytes	

*Checksum( 1 byte)* – checksum, calculated by all character before checksum XOR each other

#### Reply to command

1	1	1 byte	1 byte	1 byte
C	Packet Number	result	checksum	LF (0x10)

*Packet Number (unsigned char)* – packet number of command packet

*Result (signed char)* – result status to command

- 0, if command cannot execute successfully
- 1, if command execute successfully
- >1, error with error code (To be defined)

*Checksum( 1 byte)* – checksum, calculated by all character before checksum XOR each other

1	1	4 byte	1 byte	1 byte
---	---	--------	--------	--------

T	Packet Number	Fish time	checksum	LF (0x10)
---	---------------	-----------	----------	-----------

*Packet Number (unsigned char)* – packet number of command packet

*Fish time (long)* – FISH local time in ms from the start of microcontroller

*Checksum( 1 byte)* – checksum, calculated by all character before checksum XOR each other

End of data

For SD card file synchronization.

1      1                                      1                                      1 byte

N	0	checksum	LF (0x10)
---	---	----------	-----------

No acknowledgement from reel.smrt payload

## Appendix 4.2 Microcontroller and FreeRTOS Test Report

### Objective

1. Setup the development environment for microcontroller
2. Test basic functionality of microcontroller that available on evaluation board that related to project
3. Test functionality of FreeRTOS
4. Test functionality of FreeRTOS simulator

### Tool

- Eclipse C++
- Yagarto arm compiler
- Flash Magic
- Visual studio
- NXP LPC2368 evaluation board kit
- USB-to-serial cable
- FreeRTOS source packet

### FreeRTOS and Microcontroller Procedure

1. Download FreeRTOS source packet into your computer from <http://www.freertos.org>.
2. Follow the instruction at <http://www.freertos.org/Eclipse.html> and <http://www.yagarto.de/howto/yagarto2/index.html#download> to install development tool need to compile the FreeRTOS source code with Eclipse.
  - o Now we have Eclipse C++ with Yagarto
3. Open the demo project for LCP2368 for Eclipse in the demo directory in source packet and follow <http://www.freertos.org/Eclipse.html> in section"Opening and using a FreeRTOS.org demo project".
  - o Now we have the demo project in Eclipse with correct reference directory
4. Try to compile the project and we got the problem of compilation error.
  - 4.1 Standard IO problem.

Because of new version of Yagarto was built to support newlib stub. We have to add the stub by adding "syscalls.c" from <http://www.yagarto.de/download/yagarto/syscalls.c> to the project.  
Then include syscalls.c to the makefile.
  - 4.2 Uncomment this line in FreeRTOSConfig.h

```
/* Value to use on rev 'A' and newer devices. */  
#define configPINSEL2_VALUE 0x50150105
```
5. Finish compilation and get FreeRTOSDemo.hex
6. Download FlashMagic from [www.flashmagictool.com](http://www.flashmagictool.com)
7. Connect USB-to-serial cable to the computer and install the driver.
8. Configure jumper according to this instruction [http://www.keil.com/support/man/docs/mcb2300/mcb2300\\_fp\\_flash\\_alone.htm](http://www.keil.com/support/man/docs/mcb2300/mcb2300_fp_flash_alone.htm)
9. Burn the "FreeRTOSDemo.hex" into evaluation board using FlashMagic.

## FreeRTOS Simulation Procedure

1. Open “Unsupported demo” in the demo folder of FreeRTOS and unzip “x86\_VisualStudio8\_DJ.zip”.
2. Open the project file in the demo/WIN32 folder.
3. Try to compile and run.

## Result

The development environment has been setup in Eclipse as seen in Figure 1.

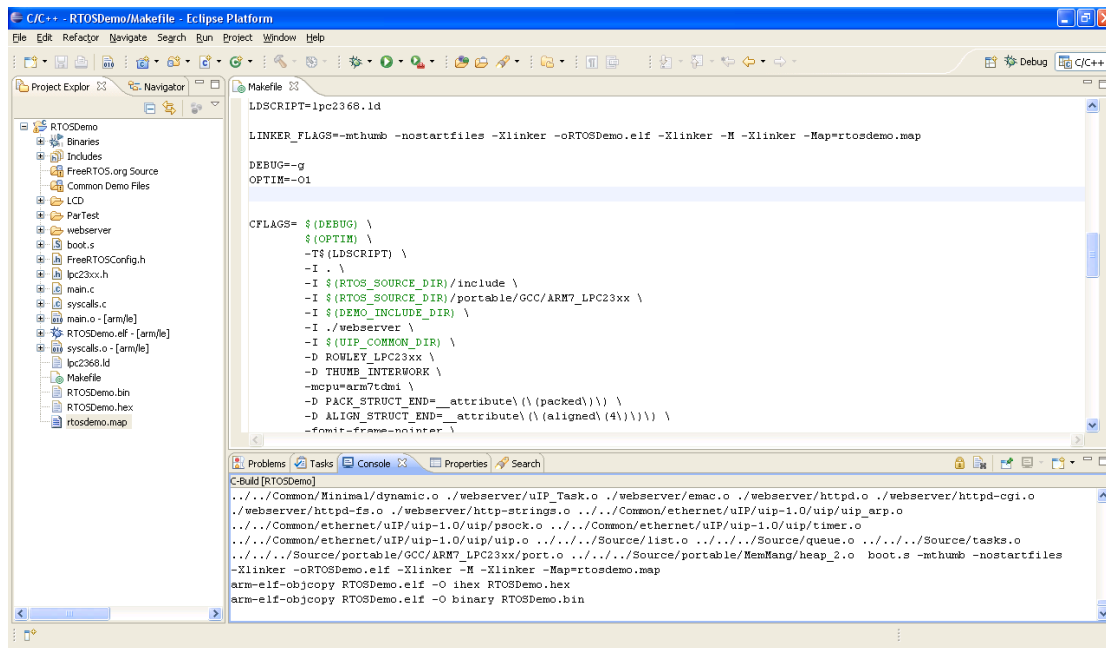
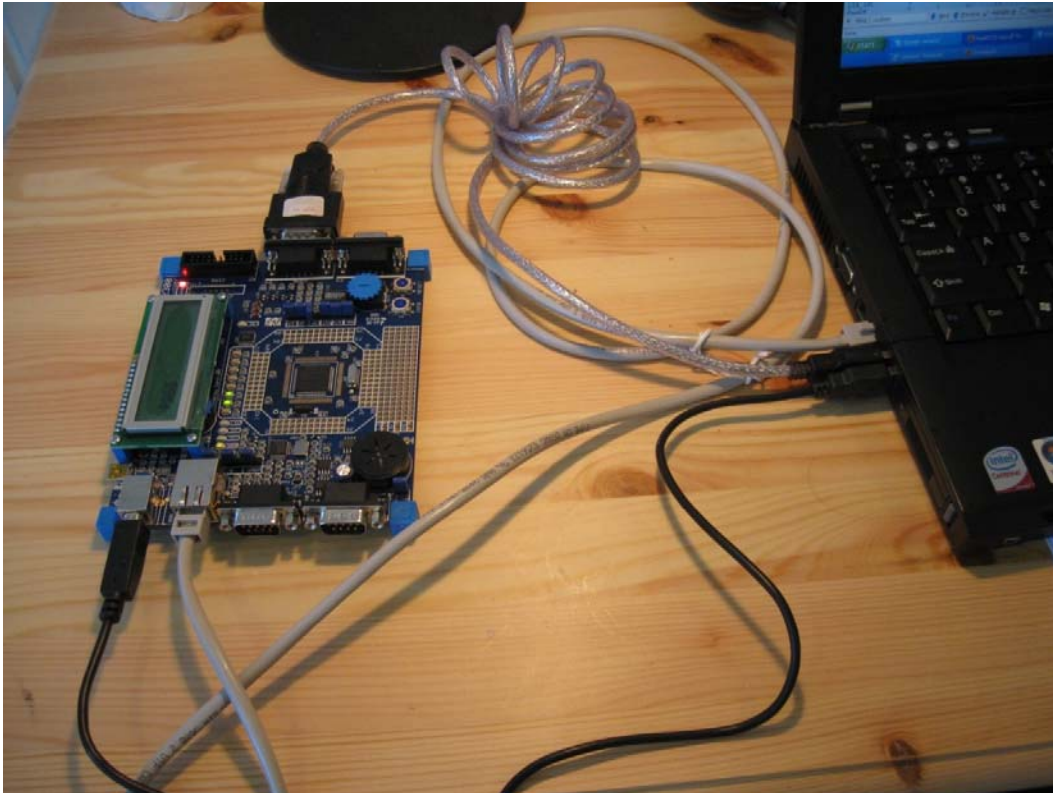


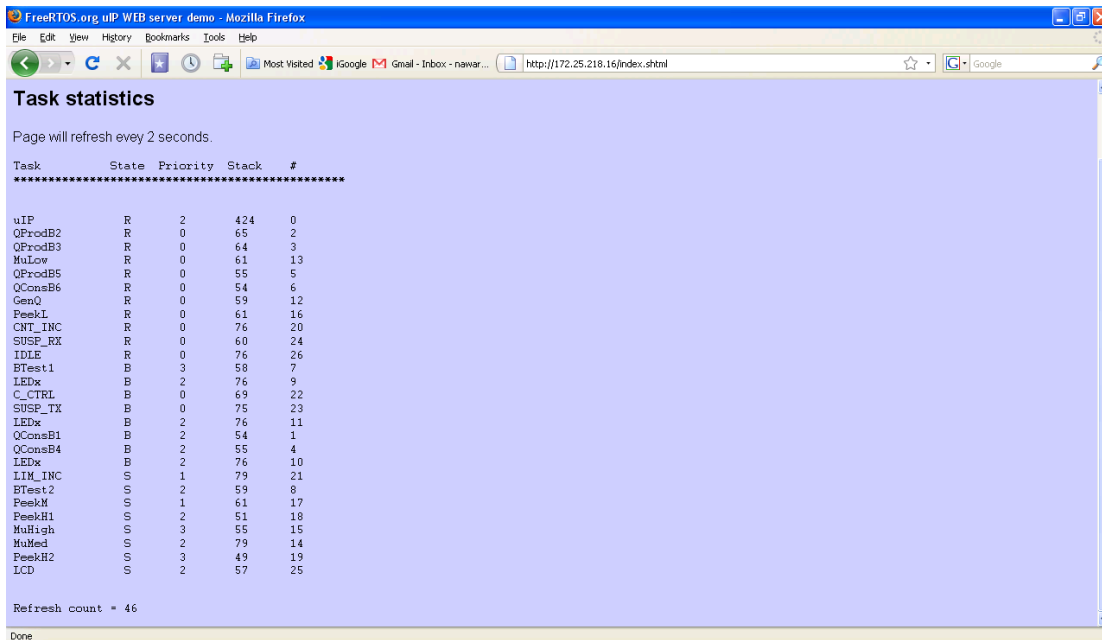
Figure 1 Eclipse development environment with demo project.

The evaluation board is running with FreeRTOS and successfully perform the LCD, LED and web server. The mutual excursion and real-time response also has been test inside the demo. The hardware connection with computer can be seen in Figure 2.



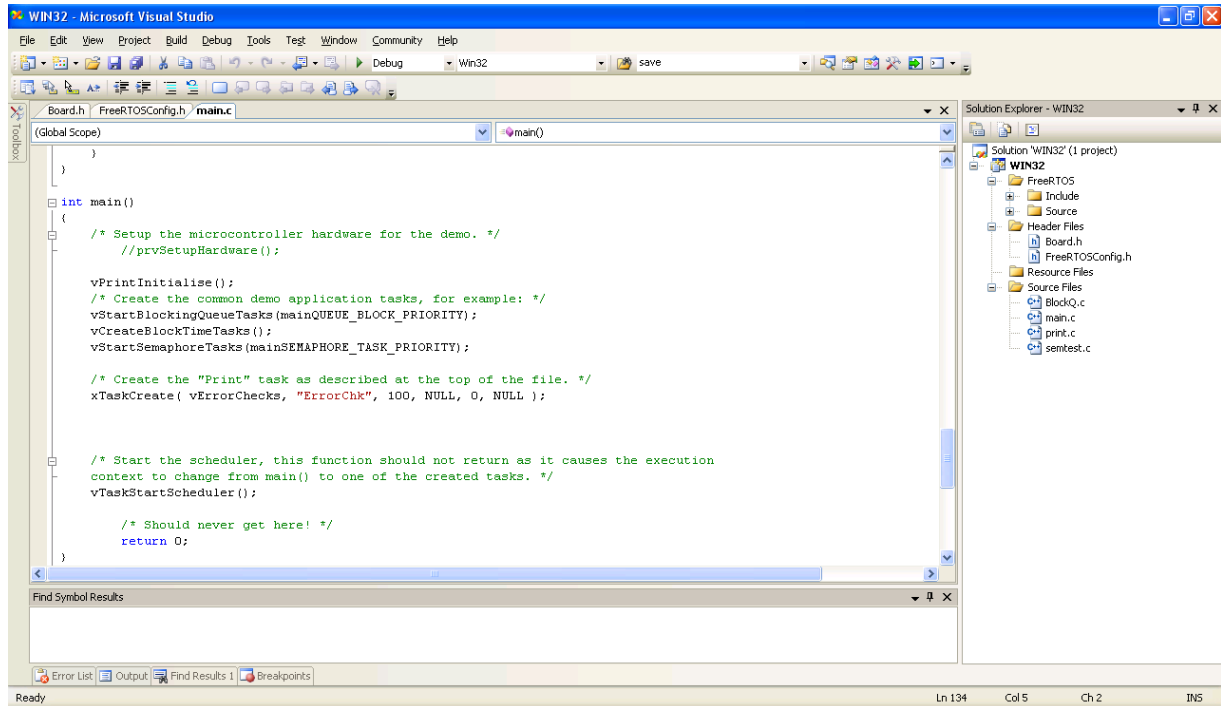
**Figure 2 Evaluation board with FreeRTOS demo.**

The web server has been test in computer. The response is quick and correct. The result page can be seen in Figure 3.



**Figure 3 Web server demo.**

Simulation environment is working fine, the task and FreeRTOS code is working in the simulation. The code in the Visual Studio 2005 development platform can be seen in Figure 4.



```
WIN32 - Microsoft Visual Studio
File Edit View Project Build Debug Tools Test Window Community Help
Debug Win32 save
Board.h FreeRTOSConfig.h main.c
(Global Scope) main()
}
}
int main()
{
    /* Setup the microcontroller hardware for the demo. */
    //prvSetupHardware();

    vPrintInitialize();
    /* Create the common demo application tasks, for example: */
    vStartBlockingQueueTasks(mainQUEUE_BLOCK_PRIORITY);
    vCreateBlockTimeTasks();
    vStartSemaphoreTasks(mainSEMAPHORE_TASK_PRIORITY);

    /* Create the "Print" task as described at the top of the file. */
    xTaskCreate( vErrorChecks, "ErrorChk", 100, NULL, 0, NULL );

    /* Start the scheduler, this function should not return as it causes the execution
    context to change from main() to one of the created tasks. */
    vTaskStartScheduler();

    /* Should never get here! */
    return 0;
}
Find Symbol Results
Error List Output Find Results 1 Breakpoints
Ready Ln 134 Col 5 Ch 2 INS
```

Figure 4 FreeRTOS simulation code.

## Conclusion

The development platform for FreeRTOS and simulation has been setup and tested. FreeRTOS has been tested which can provide good real-time deadline task. The Ethernet functionality has been tested.

To conclude, FreeRTOS and NXP LCP2368 have been proved in concept that can handle and control the experiment. The development phase can start.

## Appendix 4.3 Communication Protocol Test Report

First test - 31 July 2009

### **Objective**

1. Test the operation of the communication protocol

### **Tool**

- All software tool in appendix 4.2
- 2 evaluation boards
- 1 LAN cable
- 1 cross serial cable
- Wireshark network monitoring program
- Realterm terminal program

### **Introduction**

The test procedure is divided into 4 parts to test different functionalities of the communication protocol. In the test, the mock up sensors data have been generated to simulate equivalent amount of data in the real experiment. These data will be propagated from FISH and reel.SMRT payload to ground station. During the data propagation, buffer mechanism and acknowledgement have been tested. The commands have been sent from ground station to test system response to command.

### **Part 1: Ground station – reel.SMRT payload connection**

The tests are about client-server connection and TCP packet between reel.SMRT payload and ground station.

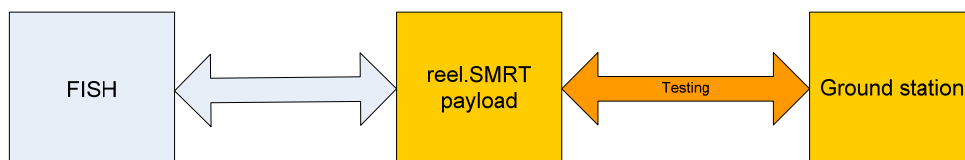


Figure 1 Testing part diagram for part 1.

### Pre-procedure

1. Connect LAN cable from evaluation board to host computer
2. Set IP of host computer to 172.25.218.1 with subnet mask of 255.255.255.0
3. Download the program of reel.SMRT payload to the evaluation board.

### **Test 1: Server and client init connection**

#### Procedure

1. Run ground station program. The program will start TCP server at port 8484.
2. Reset the evaluation board.

#### Expected result and test result

1. Ground station prints status that the connection is established - PASS

## **Test 2: Server and client reconnection**

### Procedure

1. Perform part 1 test 1.
2. Close ground station program.
3. Run the ground station program again.
4. Unplug LAN cable for 1 minute and plug the cable again.
5. Reset the evaluation board.

### Expected result and test result

1. Ground station prints status that the connection is established when the ground station is close and open again – PASS
2. Ground station prints status that the connection is established when the LAN cable unplug and plug again – PASS
3. Ground station prints status that the connection is established when reset the evaluation board – PASS

## **Test 3: TCP data transmission**

### Procedure

1. Open Wireshark and start capture the packet on the LAN interface.
2. Perform part 1 test 1.
3. Unplug the LAN cable and plug it again immediately.

### Expected result and test result

1. Ground station prints status that the reel.SMRT payload heartbeat data packets have been received continuously every 1 second – PASS
2. When the LAN cable unplug and plug again, Wireshark can capture TCP retransmission packet – PASS

## **Test 4: Command response and high data rate mode**

### Procedure

1. Open Wireshark and start capture the packet on the LAN interface.
2. Perform part 1 test 1.
3. Send change mode to data mode command from ground station.
4. Send change mode to normal mode command from ground station.

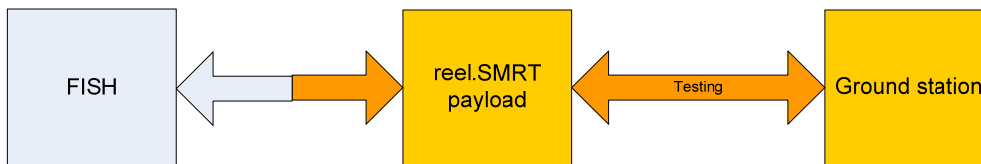
### Expected result and test result

1. Ground station shows that packet arrival rate is higher when sent “change to data mode” command – PASS
2. Ground station prints status that the reel.SMRT payload sensor data packets have been received continuously every 1 ms when sent “change to data mode” – **FAIL**, The data rate is quick but the data does not arrive continuously which can be seen from the discontinuous packet number. This shows that the send buffer is not enough for sending data in high rate.

3. Ground station prints the reply packet of “change to data mode” command – **FAIL**, the reply packet does not always arrive ground station. Sometimes, the packet arrives but sometimes there is no reply for the sent command packet. The reason may be because the send buffer in the microcontroller is not enough for the high data rate.
4. Ground station shows that packet arrival rate is slower down to 1 Hz when sent “change to normal mode” command – **PASS**
5. Ground station prints the reply packet of “change to normal mode” command – **FAIL**, the reply packet does not always arrive ground station. Sometimes, the packet arrives but sometimes there is no reply for the sent command packet. Even the high data rate mode is ended but the send buffer still full. So this may be the reason that the reply command is not always arrive.

## **Part 2: reel.SMRT payload individual response**

The test is mainly to test the response of reel.SMRT on the reel.SMRT-FISH connection.



**Figure 2 Testing part diagram for part 2 (reel.SMRT payload and ground station test).**

### Pre-procedure

1. Connect serial cable from computer to UART-2, the UART that connected to xBee.
2. Perform part 1 test 1.
3. Open realterm.

## **Test 1: Data propagation and acknowledgement**

### Procedure

1. Send heartbeat packet with checksum to the evaluation board.
2. Send FISH sensor data packet with checksum via realterm to the evaluation board.
3. Send heartbeat packet without checksum via realterm to the evaluation board.
4. Send FISH sensor data packet without checksum via realterm to the evaluation board.

### Expected result and test result

1. reel.SMRT payload sends ‘A’ back as acknowledgement to the serial connection when send heartbeat packet with checksum to it. – **PASS**
2. Ground station shows that it receives the heartbeat packet with checksum that has been sent from realterm via serial connection. – **PASS**
3. reel.SMRT payload sends reply packet with packet number and checksum back as acknowledgement to the serial connection when send FISH sensor data packet with checksum to it. – **PASS**

4. Ground station shows that it receives the FISH packet with checksum that has been sent from realterm via serial connection. – PASS
5. reel.SMRT payload does not send anything back to the serial connection when send heartbeat packet without checksum to it. – PASS
6. Ground station does not show anything that it has received any packet, when send a heartbeat packet without checksum to reel.SMRT payload. – PASS
7. reel.SMRT payload does not send anything back to the serial connection when send FISH sensor data packet without checksum to it. – PASS
8. Ground station does not show anything that it has received any packet, when send a FISH data packet without checksum to reel.SMRT payload. – PASS

## Test 2: Command propagation

### Procedure

1. Send “change to data mode” command from ground station.

### Expected result and test result

1. When send “change to data mode” command from ground station, reel.SMRT payload write “change to data mode” command with checksum to the serial connection. - PASS

## ***Part 3: FISH individual response***

FISH response according to protocol has been tested.

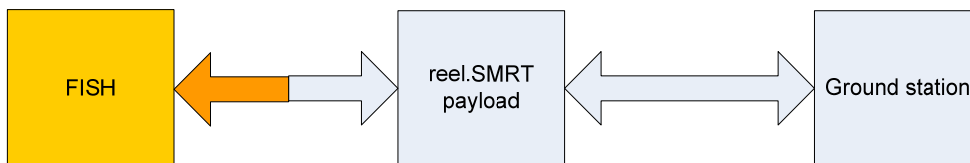


Figure 3 Testing part diagram for part 3 (FISH test).

### Pre-procedure

1. Download the program of FISH to the evaluation board.
2. Connect serial cable from computer to UART-2, the UART that connected to xBee.
3. Open realterm.

## Test: Data generation and command response

### Procedure

1. Send “change to data mode” command with checksum from realterm.
2. Send “change to normal mode” command with checksum from realterm.
3. Send “change to data mode” command without checksum from realterm.

### Expected result and test result

1. Reeterm shows that FISH send heartbeat packet out every one second when start FISH – PASS

2. When sending “change to data mode” command with checksum to FISH, FISH sends the reply packet to command and send the FISH sensors data packet out in high data rate. – PASS
3. When sending “change to normal mode” command with checksum to FISH, FISH sends the reply packet back and changes to send heartbeat packet every 1 second. – PASS
4. When sending “change to data mode” command without checksum to FISH, FISH does not send any reply back and still sends the heartbeat packet at one packet per second. – PASS

#### ***Part 4: protocol test in integrated system***

The communication protocol working in overall system is tested.

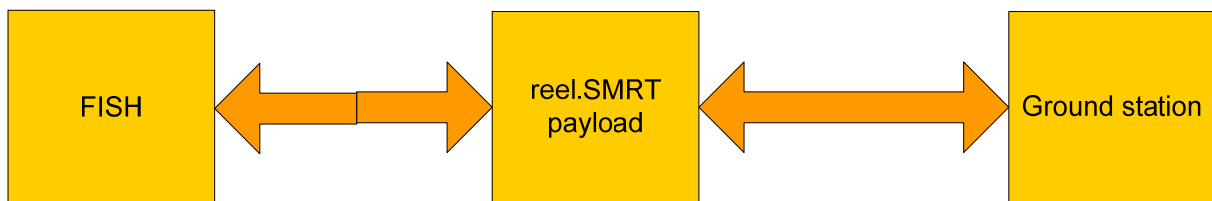


Figure 4 Testing part diagram for part 4 which is test all system.

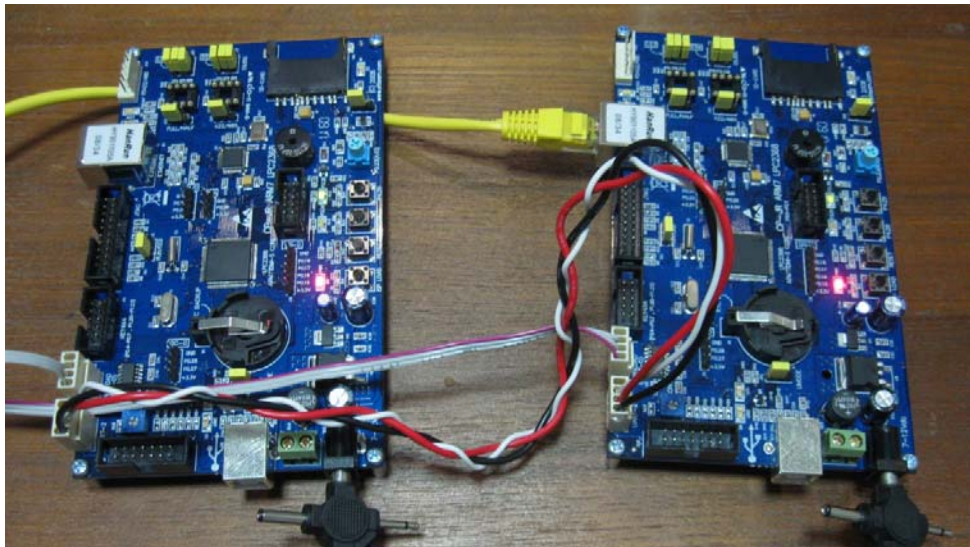


Figure 5 Overall system setup for communication protocol testing.

#### Pre-procedure

1. Connect cross serial cable to both evaluation boards UART-2 as in the Figure 5.
2. Download the program of reel.SMRT payload and FISH to each evaluation board.
3. Perform part 1 test 1 on the reel.SMRT payload board.

**Test: Perform some operations that will be perform in real experiment**

#### Procedure

1. Send “change to data mode” command from ground station.
2. Send “change to normal mode” command from ground station.

#### Expected result and test result

1. Ground station shows that it receives heartbeat packet from both FISH and reel.SMRT payload. – PASS
2. When sending “change to data mode” command, ground station shows that FISH and reel.SMRT payload change to send sensors data in higher rate. – **FAIL**, ground station receives packet less than in normal mode (less than 1 Hz), but it receives the sensors data packet. When checking FISH and reel.SMRT payload individually by disconnect the cross serial cable, FISH and reel.SMRT send the sensor data out at high rate. Reel.SMRT payload also sends the buffered packet to ground station at very high speed immediately after unplug the cross serial cable from FISH. This indicates that reel.SMRT payload has not enough CPU processing time when the data arrive from FISH in high rate.
3. When sending “change to normal mode” command, FISH and reel.SMRT change to send heartbeat packet at 1 Hz. – **FAIL**, the reel.SMRT does not response to the command. It still sends the sensors data and ground station still receives very less data. The reason can be that reel.SMRT payload does not have enough processor time to receive the command.

#### **Conclusion**

Most of the communication protocol is working fine except in the very high data rate. The send buffer is not enough because of small amount of RAM. SD card should be used for enlarging the send buffer. FISH and reel.SMRT can response to command when each performs individually but when they perform together in the system, reel.SMRT payload encounters problem in CPU processing time. The modification to the protocol should be done to solve this problem. One possible way to solve this problem is to decrease the FISH data sending rate and design the time expire for the “change to data mode” command.

#### **Remark**

The protocol specification has been changed to support the FAIL test case, the modification of the protocol specification rev 2. The further tests of the communication protocol are done during system test.