**CG1112 Engineering Principles and Practices**

**Week 7 Studio 2 – Communication Protocols**

**Answer Book**

**Student Number: A0214561M**

**Name: Zhuang Jianning**

**Pre-start Check (Check Fail: -5 mark penalty)**

**Did you name your file AxxxxxxY.docx, where AxxxxxxY is your student number? -2 marks if NO.**

**Did you fill in your student number and name above? -3 marks if NO.**

Question 1

In the output, the first character we see is the character representation of the size of TData when we Serial.write(theSize), which is 2 bytes(size of int) + 2 bytes = 4.

Then, we output the character representations of x and y. y = 10 is a newline character while x is incremented at each loop to show different characters.

Question 2

No, instead of x = 5 and y = 10, we get values like x = 655365 and y = 2123141636.

Question 3

Yes, x and y are printing correctly now. In the previous question, ints on the Arduino were 16 bits(2 bytes) wide while ints on the RPi were 32 bits(4 bytes) wide. When the data is sent over from Arduino to Pi, Pi expects 8 bytes but the original packet size is only 4 bytes. Pi reads x as the first 4 bytes while the value of y is corrupted. After replacing int with standardized integer types, both Arduino and Pi read int as 4 bytes.

Question 4

Yes, there is a difference in size. The size of TData on Arduino is 9 bytes while the size of TData on RPi is 12 bytes. This is because the character is compiled as 1 byte on the Arduino while the character is compiled as 1 byte + 3 bytes of padding on the RPi. This difference does affect the values for x and y as RPi reads the character as the first 4 bytes instead of just 1 byte. This causes the values of x and y to be different as well.

Question 5

Yes, the x, y and c fields are being printed correctly now. For architecture reasons, Pi moves data 4 bytes at a time and adds an extra 3 bytes of padding to load up 4 bytes for characters. Arduino has no padding. Hence, Pi compiles the data structure into 12 bytes while Arduino compiles data structure to 9 bytes. After actively adding in extra padding for character in Arduino, each field has 4 bytes and compiles with the correct values.

**For TA Use:**