Data write not performed correctly for SimpleTags in Series 40 NFC devices (Known Issue)

Writing a data block with SimpleTagConnection to a Mifare Ultralight tag is not done according to the JSR-257 Extensions specification.

Description

The SimpleTagConnection interface is an API extension, part of the Nokia extensions to the Contactless Communication API (JSR-257). The interface defines the necessary methods for reading and writing data and for other basic operations of the Simple Tag RFID ICs such as Mifare Ultralight tags.

The connection supports two addressing modes: physical and logical. When using the physical mode, one can read and write from any part of a tag and the offset to the memory area is specified using a block number. The first block number is 0 and the last is getBlockAmount() - 1. Physical addressing is used by the methods `read(int startBlock, int length)` and `write(int startBlock, byte[] data)`.

The write method writes the given data to the tag starting from the specified block. According to the JSR-257 API Extensions specification, if the length of the data array is not a multiple of the block size, then extra zeroes will be written to the end to fit the block size. When writing data to a Mifare Ultralight tag using the Nokia 6131 NFC and 6212 Classic NFC however, the extra zeroes are not written. In addition, with the Nokia 6131 NFC, the last block will not be written at all if the data does not fill the whole block.

How to reproduce

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import com.nokia.nfc.nxp.simpletag.SimpleTagConnection;
import javax.microedition.contactless.ContactlessException;
import javax.microedition.contactless.DiscoveryManager;
import javax.microedition.contactless.TargetListener;
import javax.microedition.contactless.TargetProperties;
import javax.microedition.contactless.TargetType;
import javax.microedition.io.Connector;


public class Test_NFC_SimpleTagConnection extends MIDlet implements TargetListener {

   int writeStartBlock = 4;
   int readNumberBlocks=12;

   /* Mifare Ultralight tag includes 16 memory blocks (one block = 4 bytes) as follows:
    -blocks 0 and 1 are read-only blocks
    -blocks 2 and 3 are one time programmable blocks
    -blocks from 4 to 15 are read and write enabled blocks
   */

   //test String for first four blocks starting from block 4 (with block 7 missing the last byte)

   String testStr = "123412341234123";
   byte[] writeBytes = testStr.getBytes();

   //String for reseting tag contents
   //String resetStr="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
   //byte[] resetBytes=resetStr.getBytes();

   SimpleTagConnection stConn;
   DiscoveryManager dm;
   Form f;
   Display d;

   public void startApp() {

      d=Display.getDisplay(this);

      f=new Form("SimpleTagConnection test");

      d.setCurrent(f);

      dm = DiscoveryManager.getInstance();

    try {

          dm.addTargetListener(this, TargetType.RFID_TAG);

      } catch (IllegalStateException e) {

          // Catch IllegalStateException

      } catch (ContactlessException e) {

          // Catch ContactlessException

      }

  }

   public void targetDetected(TargetProperties[] prop) {

       try {

       TargetProperties target = prop[0];

       stConn= (SimpleTagConnection) Connector.open(target.getUrl(SimpleTagConnection.class));

       } catch (Exception e) {

       f.append("1. Exception! :"+e.toString());}
```


(C) Copyright Nokia 2013. All rights reserved.

```
        try  {
        // Snippet for writing reset
        // stConn.write(writeStartBlock, resetBytes);
        // f.append("Reseting Mifare UL tag.\n");
        // Thread.sleep(2000);

            f.append("Writing data to a Mifare Ultralight tag. \n");

            // Let's write data into a Mifare UL tag.
            stConn.write(writeStartBlock, writeBytes);

            f.append("Write complete.\n");

            // Let's wait for a while before starting to read the written tag data.
            Thread.sleep(2000);

            f.append("Reading written data from the tag: \n");

            // Let's read the data from a Mifare UL tag starting from the 4th block to 16th block

            byte[] bytes=stConn.read(writeStartBlock, readNumberBlocks);

            String tagBytes=ByteToString(bytes);

            f.append(""+tagBytes);

  } catch (Exception e) {

     f.append("2. Exception!: "+e.toString()+"\n\n");

  e.printStackTrace();

  }
    }

    public void pauseApp() {

    }

    //Convert the byte array data from Mifare UL tag into String

    public String ByteToString(byte[] b){

    String t="";

    for(int i=0;i<b.length;i++)

        t=t+(char)b[i];

    return t;
              }
    public void destroyApp(boolean unconditional) {

    }

  }
```

1. Test the previous MIDlet using either the Nokia 6131 NFC SDK or Nokia 6212 NFC SDK Emulator.
2. Install the above MIDlet on the Nokia 6131 NFC or Nokia 6212 NFC and test it with a Mifare Ultralight tag.

Solution

Make sure that the length of the data array is a multiple of the block size by manually patching the last block with extra zeroes if necessary.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.