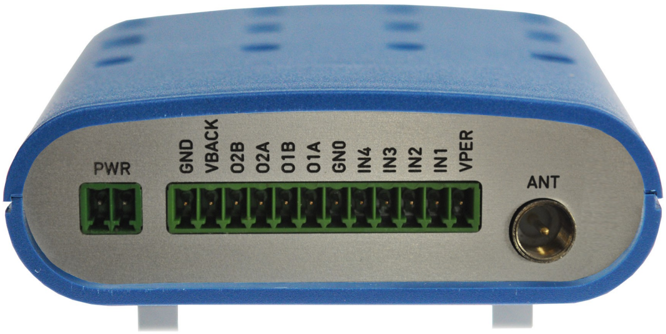
Conel

# CGK-5x

## PROGRAMMER GUIDE



www.tuv.com TÜV TÜV Rheinland Group ID:9105030435

ISO 9001:2001

CE

## Contents

## Figures list

## Symbols used

⚠ Danger – important notice, which may have an influence on the user's safety or the function of the device.

❗ Attention – notice on possible problems, which can arise in specific cases.

ℹ Information, notice – information, which contains useful advices or interest notice.

# 1. Requirements for use

## *1.1. Hardaware*

- PC with serial port or USB

## *1.2. Software*

- Windows XP or newer installed
- 40Mbytes free disk space for SMTK
- Administration privileges
- Cinterion TC65i-X Software development kit
- Cinterion The module exchange suite
- Java 2 SDK, Standard Edition 1.4. or newer

# 2. Installation of TC65i-X software development kit

All the information about installation of Cinterion TC65i-X software development kit are in Cinterion user guide **JAVA User's Guide** (version 17) for products *TC65i, TC65i-X, EGS5, EGS5-X* – JAVA$^{TM}$ Users Guide **[2]**.

# 3. Description of the CGK-5x device

## 3.1. General description

The CGK-5x is application which contain electronic device TC65i-X Communicator Java I/O with programmable GSM/GPRS module Cinterion, servants to control of electrical drive of gateways, pikes and gates by mobile telephone.

The CGK-5x module is fitted with a Java-based control software which is used to control two output ports with relays and four optically separated input ports.

## 3.2. Description of the individual parts of the CGK-5x

### 3.2.1. CGK-5x block diagram

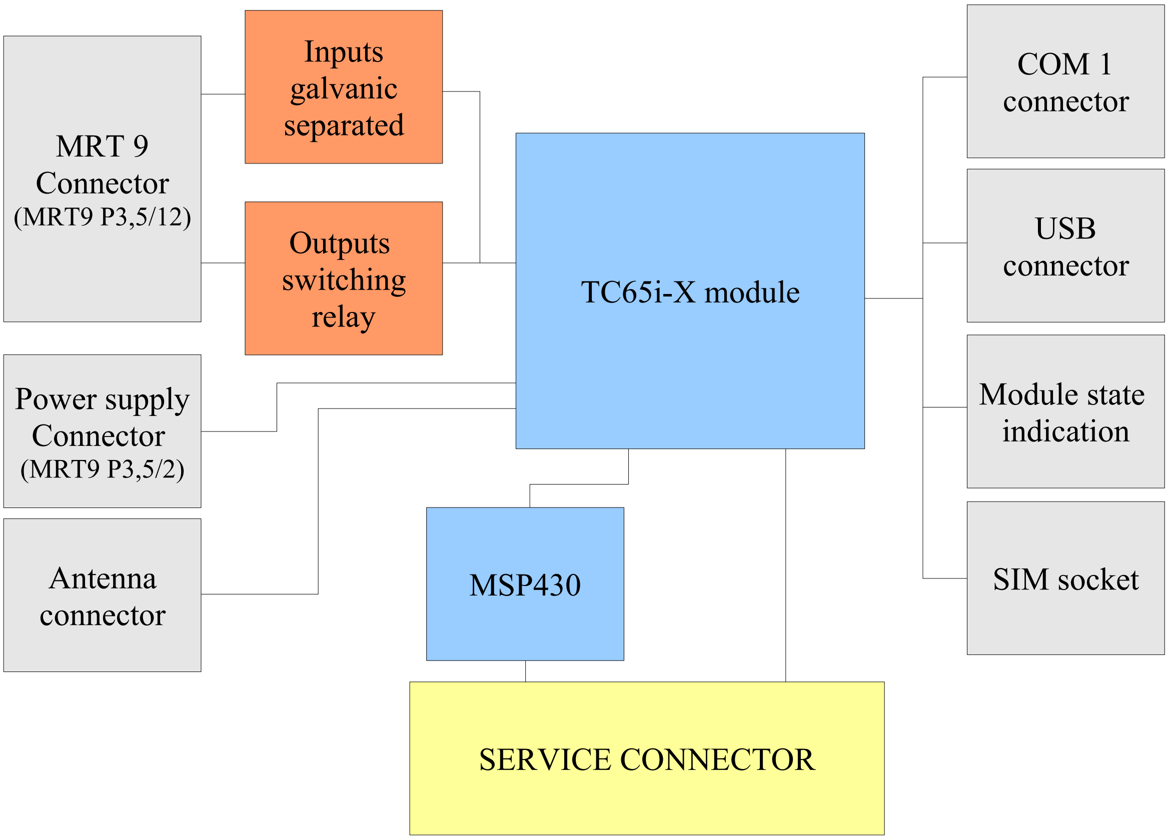


Fig. 1: CGK-5x block diagram

### 3.2.2. Programmable GSM-GPRS TC65i-X module

Wireless communication in the GSM network is carried out by means of the OEM module TC65i-X Java of the CINTERION company. It has been incorporated directly in the printed-circuit board. The push-out holder of the SIM card reader is accessible from the front panel. The antenna connector is accessible from the rear panel. The TC65i-X module is suitable for communication in both GSM bands 850/900/1800/1900 MHz.

The TC65i-X module is fitted with two serial interfaces, ASC0 and ASC1. The ASC1 interface has been brought out to the RJ45 connector, which is labelled COM. All the RS232 signals are protected against the over voltage coming through the data cable.

The module TC65i-X contain interface USB2.0 full speed too, which is take out on USB connector of type 'B' under marking USB. For this interface it is delivered driver. The USB interface is not exploited by Java application.

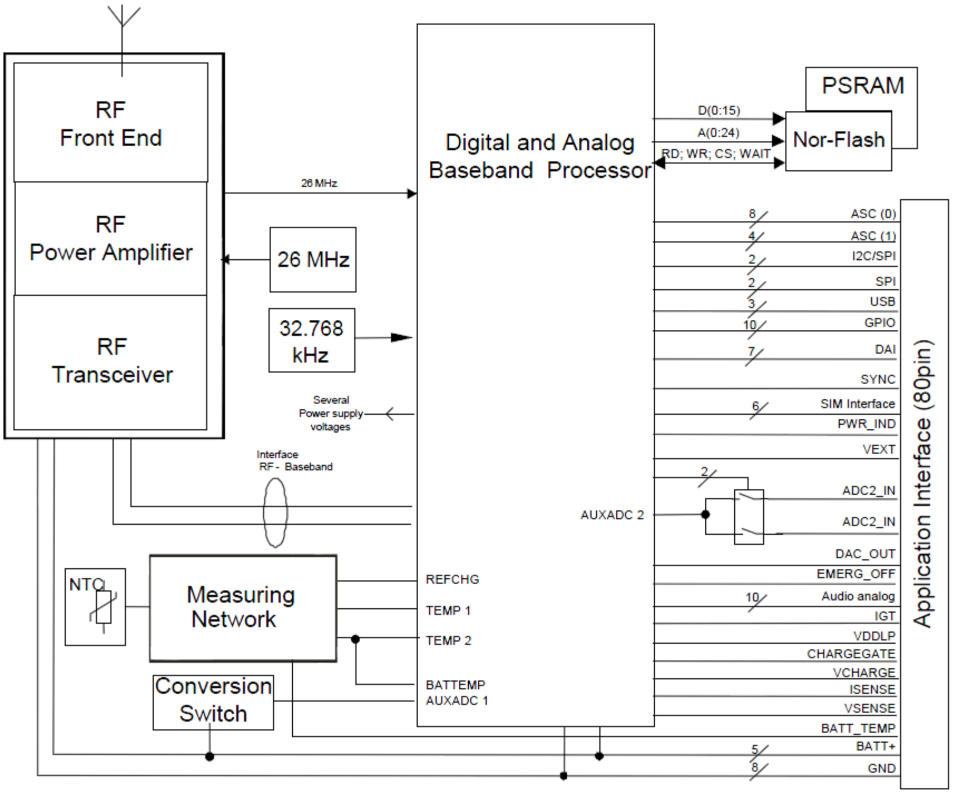


Fig. 2: TC65i-X module block diagram **[1]**

### 3.2.3. Control microcontroller

The CGK-5x communication module has been fitted with an 16-bit microcontroller that serves for starting and monitoring the TC65i-X module operation. The operating microcontroller is also used for monitoring the state of the supply voltage; if the supply voltage drops below 10,5 V or 21 V, the operating microcontroller automatically turns off the TC65i-X module. The automatic turn-off of the TC65i-X module also occurs in the event that there is a minimally power supply 10,8 V, or 21,7 V.

### 3.2.4. MSP430 - principle of WatchDog

Connection of jumper, pins in yellow rectangle in figure below, in 16-bit microcontroller deactivate hardware WatchDog function for module TC65i-X. In case of module TC65i-X working failure, the control microcontroller switch off module TC65i-X and after four seconds start him again.

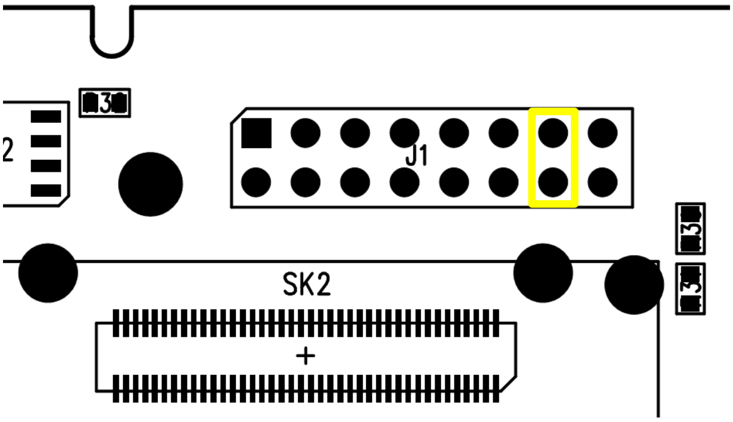


Fig. 3: Connection of jumper on CGK-5-SL desk

The WatchDog working is based on monitoring of status module TC65i-X signal GPIO1. The GPIO1 signal is possible control by the help of AT commands from running Java application. The GPIO1 signal serves to run indication of program in module TC65i-X and it is by control microcontroller carried on green LED on front panel.

After start of module TC65i-X is signal GPIO1 in level 1 and signal SYNC in level 0. The control microcontroller wait for falling edge of signal GPIO1, which the Java application signalized successful start. If it is not change of signal status to one minute, then the module TC65i-X will restart.

The Java application must in the future change to value of signal GPIO1 minimal with frequency 0,1 Hz after notice of successful start. The violation of this minimal frequency is evaluation as false and control microcontroller restart module TC65i-X.

### 3.2.5. Example of WatchDog service from Java application

```
import javax.microedition.midlet.MIDlet;
import com.cinterion.io.ATCommand;

public final class WDTest extends MIDlet {
    public void startApp() {
        try {
              // initialize of communication in AT Modem protocol
              ATCommand atcommand = new ATCommand(false);
              // I/O driver permit
              atcommand.send("AT^SPIO=1\r");
              // configuration of pin GPIO1 (LED PWR)
              atcommand.send("AT^SCPIN=1,0,1,1\r");

              while (true) {
                // LED PWR off
                atcommand.send("AT^SSIO=0,0\r");
                Thread.sleep(1000);
                // LED PWR on
                atcommand.send("AT^SSIO=0,1\r");
                Thread.sleep(1000);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    public void pauseApp() {
    }
    public void destroyApp(boolean cond) {
    }
}
```

### 3.3. Inputs and outputs

Besides the service data and USB interface, an IO interface has been created in the CGK-5x module. This is a case of two pairs of signals, where two pairs represent the relay outputs (O1A+O1B and O2A+O2B), and four represent the separated inputs (I1, I2, I3, I4) with common ground. The input I1 is possible use as counter input for counting with max. frequency 100 Hz and pulse ratio 10 to 50 %. Inputs and output circuits have been designed for voltage up to 30 V.

### 3.4. User interfaces (connectors)

At the rear panel of the CGK-5x there are situated two MRT9 connectors (12-pins IO, 2-pins PWR) and one connector FME (ANT). The IO-labelled connector has two relay outputs and four inputs. The PWR-labelled connector is used for connecting the power supply adapter and for monitoring the state of the main power supply. At the front panel of the module there is one RJ45 connector (COM).

#### 3.4.1. Connection of the COM 0 connector on board

| Pin no. | Signal identification | Description | Data flow direction |
|---|---|---|---|
| 1,2 | PWR | Output for feeding other circuits +3 V (connected directly to the feeding system of the modem) | |
| 3 | RXD0 | Receive Data | Output |
| 4,7,14 | GND | GROUND – signal ground | |
| 5 | TXD0 | Transmit Data | Input |
| 6 | TEST_MSP | MSP – test pin | |
| 8 | $\overline{\text{RST\_MSP}}$ | MSP – Request To Send | Input |
| 9 | $\overline{\text{CD0}}$ | Carrier Detect | Output |
| 10 | $\overline{\text{CTS0}}$ | Clear To Send | Output |
| 11 | $\overline{\text{DTR0}}$ | Data Terminal Ready | Input |
| 12 | $\overline{\text{RTS0}}$ | Request To Send | Input |
| 13 | WD | WatchDog | |
| 15,16 | PWR | Output for feeding other circuits +4 V (connected directly to the feeding system of the modem) | |

**Beware!** On COM 0 connector aren't RS232 levels.

#### 3.4.2. Connection of the COM 1 connector

The RJ45 panel socket. (RS232 – DCE – Data Communication Equipment)

| Pin no. | Signal identification | Description | Data flow direction |
|---|---|---|---|
| 1 | RTS | Request To Send | Input |
| 2 | CTS | Clear To Send | Output |
| 3 | DTR | Data Terminal Ready | Input |
| 4 | DSR | Data Set Ready | Output |
| 5 | GND | GROUND – signal ground | |
| 6 | RXD | Receive Data | Output |
| 7 | CD | Carrier Detect | Output |
| 8 | TXD | Transmit Data | Input |

### 3.4.3. Connection of the IO connector

Connector MRT9 P3,5/12.

| Pin no. | Signal identification | Description |
|---|---|---|
| 1 | GND | Signal and power supply ground |
| 2 | VBACK | Connection for the backup battery |
| 3 | O2B | Relay output |
| 4 | O2A | Relay output |
| 5 | O1B | Relay output |
| 6 | O1A | Relay output |
| 7 | GND | Signal and power supply ground |
| 8 | IN4 | Input (can to use as countig input) |
| 9 | IN3 | Input |
| 10 | IN2 | Input |
| 11 | IN1 | Input |
| 12 | VPER | Output for feeding other circuits (connected directly to the feeding system of the modem) |

### 3.4.4. Connection of the supply PWR connector

Connector MRT9 P3,5/2.

| Pin no. | Signal identification | Description |
|---|---|---|
| 1 | +UN | Positive pole of the DC supply voltage (10 to 30 V) |
| 2 | NC | Signal not connected |
| 3 | NC | Signal not connected |
| 4 | +UN | Positive pole of the DC supply voltage (10 to 30 V) |
| 5 | GND | Negative pole of the DC supply voltage |
| 6 | GND | Negative pole of the DC supply voltage |

### 3.4.5. Connection of the USB connector

The USB panel socket.

| Pin no. | Signal identification | Description |
|---|---|---|
| 1 | + UN | Positive pole of the DC supply voltage (5 VDC) |
| 2 | D - | Data - |
| 3 | D + | Data + |
| 4 | GND | Negative pole of the DC supply voltage |

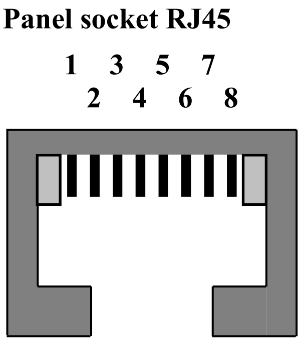


Fig. 4: Panel socket RJ45

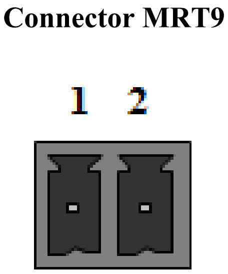


Fig. 5: Connector MRT9 P3,5/2

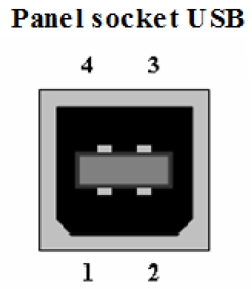


Fig. 6: USB connector

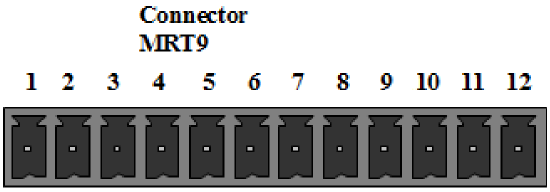


Fig. 7: Connector MRT9 P3,5/12

## 3.5. *Description of ports*

### 3.5.1. Description of IO circuitry

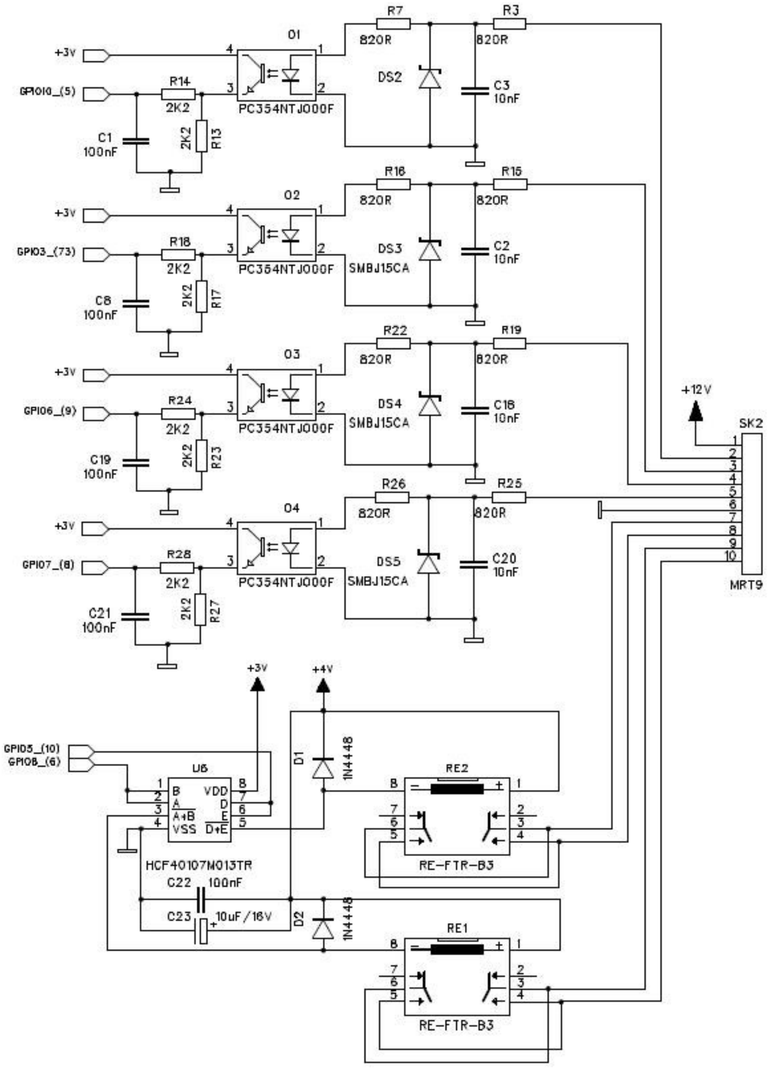


Fig. 8: IO circuitry

### 3.5.2. Description of COM 0 circuitry

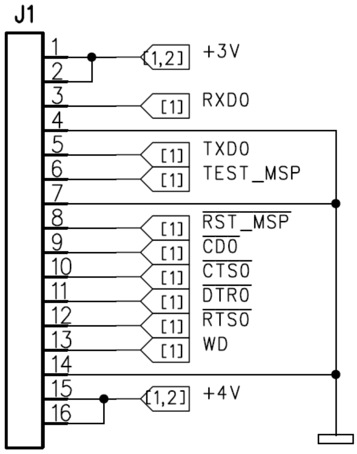


Fig. 9: COM 0 circuitry

### 3.5.3. Description of  COM 1 circuitry

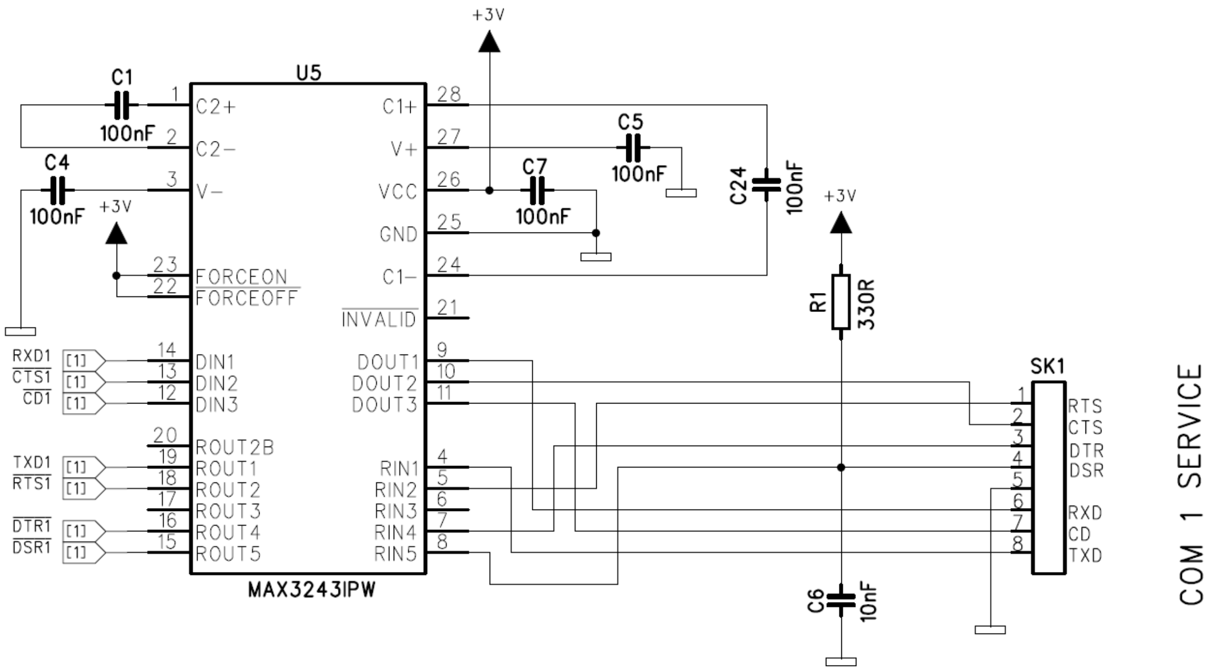


Fig. 10: COM 1 circuitry

# 4. Example of JAVA application

```
package demo;

import javax.microedition.midlet.MIDlet;
import com.cinterion.io.ATCommand;

// -------------------------------------------------------------------------------
// Demo class
public final class Demo extends MIDlet {

  // GPIO configuration
  private static int GPIO_LED  = 0;
  private static int GPIO_DSR  = 1;
  private static int GPIO_DTR  = 3;
  private static int GPIO_CD   = 8;
  private static int GPIO_IN1  = 9;
  private static int GPIO_IN2  = 2;
  private static int GPIO_IN3  = 5;
  private static int GPIO_IN4  = 6;
  private static int GPIO_OUT1 = 4;
  private static int GPIO_OUT2 = 7;

  // AT parser
  private ATCommand atcommand = null;

  // -----------
  // constructor
  public Demo() {
    System.out.println("Demo: init");
  }

  // ----------------------
  // signal stock taking
  private int getIO(int pin) {
    try {
      String response = atcommand.send("AT^SGIO=" + pin + '\r');
      if (response.indexOf("OK") >= 0) {
        return (response.indexOf('1') >= 0) ? 1 : 0;
      } else {
        return -1;
      }
    } catch (Exception e) {
      e.printStackTrace();
      return -1;
    }
  }
```

```
// -----------------------
// signal status setting
private synchronized void setIO(int pin, int state) {
  try {
    atcommand.send("AT^SSIO=" + pin + ',' +  state + '\r');
  } catch (Exception e) {
    e.printStackTrace();
  }
}

// --------------
// main program
public void startApp() {
  boolean ok;

  System.out.println("Demo: start");

   // initialization AT parsers
  do {
    System.out.print("Demo: initializing AT parser... ");
    try {
      atcommand = new ATCommand(false);
      if (atcommand != null) {
        if (atcommand.send("\rAT\r").indexOf("OK") < 0) {
          atcommand.release();
          atcommand = null;
        }
      }
      System.out.println(atcommand != null ? "ok" : "error");
    } catch (Exception e) {
      e.printStackTrace();
    }
  } while (atcommand == null);

   // input and output pins initialization
  do {
    System.out.print("Demo: initializing I/O pins... ");
    try {
        // I/O driver permit
      ok = atcommand.send("AT^SPIO=1\r").indexOf("OK") >= 0;
        // GPIO1 (POWER LED) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,0,1,1\r").indexOf("OK") >= 0;
        // GPIO2 (DSR on COM1) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,1,0\r").indexOf("OK") >= 0;
        // GPIO3 (IN2G) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,2,0\r").indexOf("OK") >= 0;
        // GPIO4 (DTR on COM1) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,3,0\r").indexOf("OK") >= 0;
        // GPIO5 (OUT1G) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,4,1,0\r").indexOf("OK") >= 0;
        // GPIO6 (IN3G) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,5,0\r").indexOf("OK") >= 0;
        // GPIO7 (IN4G) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,6,0\r").indexOf("OK") >= 0;
        // GPIO8 (OUT2G) pin configuration
      ok &= atcommand.send("AT^SCPIN=1,7,1,0\r").indexOf("OK") >= 0;
        // GPIO9 (CD on COM1) pin configuration
        ok &= atcommand.send("AT^SCPIN=1,8,1,1\r").indexOf("OK") >= 0;
          // GPIO10 (IN1G) pin configuration
        ok &= atcommand.send("AT^SCPIN=1,9,0\r").indexOf("OK") >= 0;
          // if initialization was successful
        if (ok) {
          System.out.println("ok");
          // if initialization was not successful
        } else {
          System.out.println("error");
          atcommand.send("AT^SPIO=0\r");
        }
      } catch (Exception e) {
        ok = false;
        e.printStackTrace();
      }
    } while (!ok);


     // main loop
    try {
      boolean state = false;
      while (true) {
        setIO(GPIO_LED, 1);
        Thread.sleep(100);
        setIO(GPIO_LED, 0);
        Thread.sleep(900);
        System.out.println("Demo: IN1=" + getIO(GPIO_IN1) + ", IN2=" + getIO(GPIO_IN2) +
                          ", IN3=" + getIO(GPIO_IN3) + ", IN4=" + getIO(GPIO_IN4));
        setIO(GPIO_OUT1, state ? 1 : 0);
        setIO(GPIO_OUT2, state ? 0 : 1);
        state = !state;
      }
    } catch (Exception e) {
      e.printStackTrace();
    }

     // applications termination
    destroyApp(true);
    notifyDestroyed();
  }

  // ------------------
  // applications stop
  public void pauseApp() {
    System.out.println("Demo: pause");
  }

  // -----------------
  // applications termination
  public void destroyApp(boolean cond) {
    System.out.println("Demo: destroy");
  }

}
```

More informations about JAVA programming in TC65i-X module are in reference **[2]** and about AT commands are informations in reference **[3]**.

# 5. Reference

**[1]** Cinterion**: TC65i-X_HD_v02.004 –** Hardware Interface Description**, 2012**

**[2]** Cinterion**: wm_java_usersguide_v17** – JAVA$^{TM}$ Users Guide**, 2011**

**[3]** Cinterion**: TC65i-X_ATC_V02.004** – AT command Set**, 2012**

# 6. Links to related products of the manufacturer

Related products and materials with a reference can be found on the manufacturer's website Conel company:

www.conel.cz

There are another links on Cinterion company website, TC65i-X module:

www.cinterion.com