AN11402

How to implement the ICODE ILT anti-collision

Rev. 1.0 — 23 October 2013
274910

Application note
COMPANY PUBLIC

Document information

Info	Content
Keywords	CLEV663B, ICODE ILT, anti-collision
Abstract	This application note describes how to implement and use the ICDOE ILT anti-collision on the CLEV663B.

Revision history

Rev	Date	Description
1.0	20131023	First release

Contact information

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

1.1 Scope

This application note will guide you through a step by step manual to implement the anti-collision feature for the ICODE ILT transponder IC on CLEV663 or CLEV610 Blueboard.

1.1.1 What you need

CLEV663B or CLEV610B (Blueboard)

LPCXpresso board - LPC1115

NXP Reader Library 2.1 - http://www.nxp.com/documents/software/200312.zip

RC663 Polling Example - http://www.nxp.com/documents/software/233812.zip

2. Step by step manual

2.1 Switching to SPI

As a first step you have to change the interface from I2C to SPI at the blueboard itself and in the SW (RC663 Polling example)

1. The HW changes which must be done are documented in the “Quick Start Up Guide RC663 Blueboard” http://www.nxp.com/documents/application_note/AN11211.pdf
2. In the RC663 Polling example you have to change the comment lines:
    a. In \RC663-Polling\src\phSubBal\include\RegCtl_I2cHw.h comment line 26 `//#define I2C_USED`
    b. In \RC663-Polling\src\phSubBal\include\RegCtl_SpiHw.h comment line 29 `#define SPI_USED`
3. Start the changed RC663 Polling example

2.2 Replacing the NXP Reader Lib

1. Make a local copy of the following files from the NXP Reader lib:

- RC663-Polling\src\NxpRdLib_PublicRelease\comps\phbalReg\src\Stub
- RC663-Polling\src\NxpRdLib_PublicRelease\comps\phhalHW\src\Rc663\phhalHw_Rc663.c
- RC663-Polling\src\NxpRdLib_PublicRelease\comps\phhalHW\src\Rc663\phhalHw_Rc663_Int.c

2. Close the LPCXpresso and and delete the complete NXP Reader Lib from the Project Directory. (“NxpRdLib_PublicRelease” in “\RC663-Polling\src)
3. Copy the NXP Reader Lib 2.1 in the project directory
4. Delete the Example folder (“ex”) from \RC663-Polling\src\NxpRdLib_PublicRelease

5. Replace the “Stub” folder in \RC663-Polling\src\NxpRdLib_PublicRelease\comps\phbalReg\src\Stub with the old one (Step 1)
6. Replace the file \RC663-Polling\src\NxpRdLib_PublicRelease\comps\phhalHw\src\Rc663\phhalHw_Rc663.c with the old one (Step 1)
7. Replace the file \RC663-Polling\src\NxpRdLib_PublicRelease\comps\phhalHw\src\Rc663\phhalHw_Rc663_Int.c with the old one (Step 1)
8. Open LPCXpresso again and refresh your project
9. Following comment lines must be changed
    1. In “\RC663-Polling\src\NxpRdLib_PublicRelease\types\ph_TypeDefs.h” comment the Line 46
    ```
    //typedef unsigned char uint8_t;
    ```
    and the Line 76
    ```
    //typedef char int8_t;
    ```
    and include <stdint.h> in the file
    ```
    #include <stdint.h>
    ```
    2. In “\RC663-Polling\src\NxpRdLib_PublicRelease\types\ph_NxpBuild.h” comment line 46 – 50
    ```
    //#define NXPBUILD__PHBAL_REG_SERIALWIN
    //#define NXPBUILD__PHBAL_REG_PCSCWIN
    //#define NXPBUILD__PHBAL_REG_
    //#define NXPBUILD__PHBAL_REG_PIPELIN
    //#define NXPBUILD__PHBAL_REG_SOCKETWIN
    ```
    comment line 63-67 and 69
    ```
    //#define NXPBUILD__PHHAL_HW_RC523
    //#define NXPBUILD__PHHAL_HW_RD70X
    //#define NXPBUILD__PHHAL_HW_RC632
    //#define NXPBUILD__PHHAL_HW_RDCARDSIM
    //#define NXPBUILD__PHHAL_HW_CALLBACK
    //#define NXPBUILD__PHHAL_HW_RD710
    ```
    comment line 84, 85, 104, 114, 125, 232, 242 and 244
    ```
    //#define NXPBUILD__PHPAL_I14443P3A_RD70X
    //#define NXPBUILD__PHPAL_I14443P3A_RD710
    //#define NXPBUILD__PHPAL_I14443P4A_RD710
    //#define NXPBUILD__PHPAL_I14443P4_RD710
    //#define NXPBUILD__PHPAL_MIFARE_RD710
    //#define NXPBUILD__PH_KEYSTORE_RC632
    //#define NXPBUILD__PH_KEYSTORE_RD710
    ```
    comment line 255
    ```
    //#define NXPBUILD__PH_LOG
    ```
    3. In \RC663-Polling\src\phSubBal\src\RegCtl_SpiHw.c on line 425 replace 0x0A with 0x01
    ```
    LPC_SYSCON->SSP0CLKDIV = 0x01;
    ```

Now if you add `#define DEBUG_MESSAGE 1` in your main.c, you should again be able to run the polling example and detect 15693 tags.

2.3 Editing the ISO18000-3.3 protocol

In \RC663-Polling\src\NxpRdLib_PublicRelease\comps\phpalI18000p3m3\src\Sw\phpalI18000p3m3_Sw.c:

1. comment line 94 – 97

```
/* Set Frame Sync */
//PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_SetConfig(
//   pDataParams->pHalDataParams,
//     PHHAL_HW_CONFIG_FRAME_SYNC,
//     0));
```

2. At line 98 (before the exchange) add following lines:

```
uint8_t bReg;
phhalHw_ReadRegister(pDataParams->pHalDataParams,
    PHHAL_HW_RC663_REG_TXDATANUM, &bReg);
phhalHw_WriteRegister(pDataParams->pHalDataParams,
    PHHAL_HW_RC663_REG_TXDATANUM, (bReg & ~0x7) | bTxLastBits);
```

3. At line 361 extend the T1 + T3 timeout by 10 us:

```
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_SetConfig(
        pDataParams->pHalDataParams,
        PHHAL_HW_CONFIG_TIMEOUT_VALUE_US,
        PHPAL_I18000P3M3_SW_T1_MAX_US + PHPAL_I18000P3M3_SW_T3_MIN_US + 10));
```

4. At line 516 extend the T1 + T3 timeout by 10 us:

```
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_SetConfig(
        pDataParams->pHalDataParams,
        PHHAL_HW_CONFIG_TIMEOUT_VALUE_US,
        PHPAL_I18000P3M3_SW_T1_MAX_US + PHPAL_I18000P3M3_SW_T3_MIN_US + 10));
```

5. At line 524(before the exchange) add following line:

```
  PH_CHECK_SUCCESS_FCT(statusTmp,
      phhalHw_SetConfig(pDataParams->pHalDataParams, PHHAL_HW_CONFIG_TXCRC,
PH_OFF));
  PH_CHECK_SUCCESS_FCT(statusTmp,
      phhalHw_SetConfig(pDataParams->pHalDataParams, PHHAL_HW_CONFIG_RXCRC,
PH_OFF));
```

6. At line 618 extend the T1 + T3 timeout by 10 us:

```
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_SetConfig(
        pDataParams->pHalDataParams,
        PHHAL_HW_CONFIG_TIMEOUT_VALUE_US,
        PHPAL_I18000P3M3_SW_T1_MAX_US + PHPAL_I18000P3M3_SW_T3_MIN_US + 10));
```

7. At line 969 (before the exchange) add following lines:

```
    uint8_t bReg;
    phhalHw_ReadRegister(pDataParams->pHalDataParams,
        PHHAL_HW_RC663_REG_TXDATANUM, &bReg);
    phhalHw_WriteRegister(pDataParams->pHalDataParams,
        PHHAL_HW_RC663_REG_TXDATANUM, (bReg & ~0x7) | bBitLength);
```

8. At line 989 (before the exchange) add following lines:

```
    uint8_t bReg;
    phhalHw_ReadRegister(pDataParams->pHalDataParams,
```

```
        PHHAL_HW_RC663_REG_TXDATANUM, &bReg);
    phhalHw_WriteRegister(pDataParams->pHalDataParams,
        PHHAL_HW_RC663_REG_TXDATANUM, (bReg & ~0x7) | bBitLength);
```

2.4 Implementation of the anti-collision

1. Add `#include <phpalI18000p3m3.h>` to your main.c
2. Add following function to your main.c, before your main function, or make a forward declaration:

```
uint8_t I180003m3Anticollision(void *pHal)
{
  uint8_t bCRC[200][14];
  phStatus_t status = PH_ERR_SUCCESS;
  phpalI18000p3m3_Sw_DataParams_t palI180003m3;
  uint8_t pStore[20];
  uint8_t * pBuffer;
  uint16_t len;
  uint8_t coll = 0;
  uint8_t successes = 0;
  uint8_t timeouts = 0;
  uint8_t bQ = 1;
  uint8_t bi;

  //Initial I180003m3
  status = phpalI18000p3m3_Sw_Init(&palI180003m3,
      sizeof(phpalI18000p3m3_Sw_DataParams_t), pHal);
  //Applay protocollsetting
  status = phhalHw_ApplyProtocolSettings(pHal, PHHAL_HW_CARDTYPE_I18000P3M3);
  // make as many rounds as there are responding tags
  for (timeouts = 0; timeouts != (1 << bQ);)
  {
    timeouts = 0;
    // if more than 50% of the slots was timeouts inc slots
    if ((coll / (1 << bQ)) > 0.5 && !(bQ == 15))
    {
      bQ++;
    }
    // reset collision counter
    coll = 0;
    //begin round with the desired slots
```

```
    status = phpalI18000p3m3_BeginRound(&palI180003m3, 0,
        PHPAL_I18000P3M3_M_MANCHESTER_4, 0, 0, 0, 0, bQ, pStore);
    //check if a tag responds
    if (status == PH_ERR_SUCCESS)
    {
      //ack the tag
status = phpalI18000p3m3_Ack(&palI180003m3, PHPAL_I18000P3M3_ACK_USE_CRC,
          pStore, &pBuffer, &len);
      //if the tag doesn't responds correctly send a nak to get him in the
next round
      if (status != PH_ERR_SUCCESS)
      {
        status = phpalI18000p3m3_Nak(&palI180003m3);
        coll++;
      }
      // else the tag is invented
      else
      {
        memcpy(bCRC[successes], pBuffer, 14);
        successes++;
      }
    }
    //timeout occurs
    else if ((status & PH_ERR_MASK) == PH_ERR_IO_TIMEOUT)
    {
      timeouts++;
    }
    // collision occurs
    else if ((status & PH_ERR_MASK) == PH_ERR_COLLISION_ERROR)
    {
      coll++;
    }
    // do as many times as there are slots
    for (bi = 1; bi < (1 << bQ); bi++)
    {
      // next slot command
      status = phpalI18000p3m3_NextSlot(&palI180003m3, pStore);
      //check if a tag responds
      if (status == PH_ERR_SUCCESS)
      {
```

```
        //ack the tag
        status = phpalI18000p3m3_Ack(&palI180003m3,
            PHPAL_I18000P3M3_ACK_USE_CRC, pStore, &pBuffer, &len);
        //if the tag doesn't responds correctly send a nak to get him in the
next round
        if (status != PH_ERR_SUCCESS)
        {
          status = phpalI18000p3m3_Nak(&palI180003m3);
        }
        // else the tag is invented
        else
        {
          memcpy(bCRC[successes], pBuffer, 14);
          successes++;
        }
      }
      //timeout occurs
      else if ((status & PH_ERR_MASK) == PH_ERR_IO_TIMEOUT)
      {
        timeouts++;
      }
      // collision occurs
      else if ((status & PH_ERR_MASK) == PH_ERR_COLLISION_ERROR)
      {
        coll++;
      }
    }
    // send last next Slot for the last Tag
    status = phpalI18000p3m3_NextSlot(&palI180003m3, pStore);
  }
  printf("%d Tags found\n", successes);
  for (bi = 0; bi < successes; bi++)
  {
    uint8_t bj = 0;
    for (bj = 0; bj < 14; bj++)
    {
      printf("%02x", bCRC[bi][bj]);
    }
    printf("\n");
  }
```

```
    return status;
}
```

10. Add the following code at line 336:

```
I180003m3Anticollision(pHal);
```

3. Legal information

3.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

3.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

3.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

ICODE — is a trademark of NXP B.V.

4. Contents

For more information, visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 23 October 2013
274910
Document identifier: AN11402