CRC Implementation

ABSTRACT

This document is an overview of the CRC implementation for the CC430, CC1100, CC1100E, CC1101, CC1110, CC1111, CC1150, CC2500, CC2510, CC2511, and CC2550. See Section 4 for links to the data sheets and user's guides.

Contents

List of Figures

Trademarks

All trademarks are the property of their respective owners.

1 Abbreviations

CRC	Cyclic redundancy check
MSB	Most significant bit

2 CRC Implementation

CC2500, CC2510, CC2511, and CC2550 all support two CRC implementations:

- Normal mode CRC
- CC2400 CRC (only for backwards compability)

Bit 3 (CC2400_EN) in the PKTCTRL0 register is used to select which mode to use.

CC430, CC1100, CC1100E, CC1101, CC1110, CC1111, and CC1150 only support the normal mode CRC, and hence bit 3 in the PKTCTRL0 register is not used on these devices.

In the normal mode (PKTCTRL0.CC2400_EN = 0), the CRC polynomial is CRC16 ($x^{16} + x^{15} + x^2 + 1$) with the CRC register reset to all ones. Figure 1 shows the shift register implementation. Note that the data input is at x^{16}.

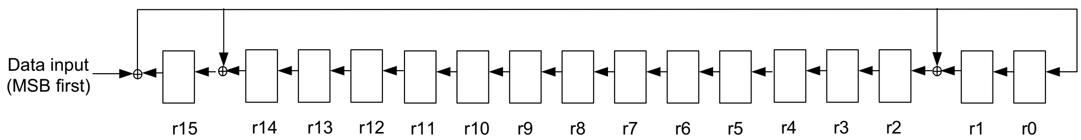

Figure 1. Normal Mode CRC

For backward compatibility, the CC2400 CRC implementation is also supported (PKTCTRL0.CC2400_EN = 1). Figure 2 shows the shift register implementation. The data input is at x^0. When PKTCTRL0.CC2400_EN = 1, PKTCTRL0.WHITE_DATA and PKTCTRL1.CRC_AUTOFLUSH must be 0.

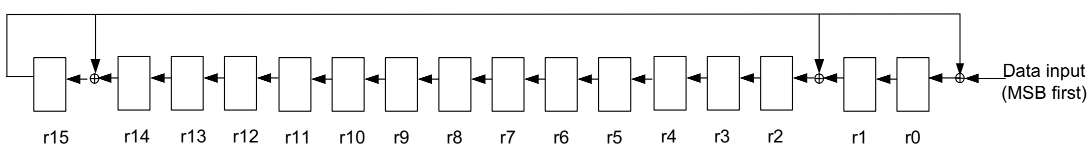

Figure 2. CC2400 CRC

CRC is calculated over all bytes transmitted after the sync word. The CRC16 checksum is automatically transmitted after the last data byte. MSB is transmitted first.

See the device-specific data sheets for detailed use of the CRC functionality.

3 Code Example

```
#define  CRC16_POLY 0x8005


UINT16 culCalcCRC(BYTE crcData, UINT16 crcReg) {
    UINT8 i;

    for (i = 0; i < 8; i++) {
        if (((crcReg & 0x8000) >> 8) ^ (crcData & 0x80))
            crcReg = (crcReg << 1) ^ CRC16_POLY;
        else
            crcReg = (crcReg << 1);

        crcData <<= 1;
    }
    return crcReg;
}// culCalcCRC

//------------------------------------------------------------------
// Example of Usage

#define  CRC_INIT 0xFFFF
UINT8 txBuffer = {0, 1, 2, 3, 4, 5};
UINT16 checksum;
UINT8 i;

checksum = CRC_INIT;   // Init value for CRC calculation

for (i = 0; i < sizeof(txBuffer); i++)
    checksum = culCalcCRC(txBuffer[i], checksum);
```

4 References

1. CC430 User's Guide
2. CC1100 Single-Chip Low Cost Low Power RF-Transceiver
3. CC1100E Low-Power Sub-GHz RF Transceiver (470-510 MHz & 950-960 MHz)
4. CC1101 Single-Chip Low Cost Low Power RF-Transceiver
5. CC1110Fx/CC1111Fx Low-Power Sub-1 GHz RF System-on-Chip (SoC) with MCU, Memory, Transceiver, and USB Controller
6. CC1150 Single Chip Low Cost Low Power RF-Transmitter
7. CC2500 Single-Chip Low Cost Low Power RF-Transceiver
8. CC2510Fx/CC2511Fx Low-Power SoC (System-on-Chip) with MCU, Memory, 2.4 GHz RF Transceiver, and USB Controller
9. CC2550 Low-Cost Low-Power 2.4 GHz RF Transmitter

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from October 27, 2009 to September 27, 2018	Page
• Formatting and editorial changes throughout document	1