XT32H05x

XT32 microcontroller CRC

Application notes

Rev 0.0.0

Original Release Date: 28-Sep-2023

Revised :

Revision History

Release	Date	Author	Summary of Change
V0.0.0	28/09/2023	Shirling Liu	Initial

Contents

1 INTRODUCE 1

1.1 REQUIRED PERIPHERALS 1

1.2 COMPATIBLE DEVICES 1

2 DESIGN DESCRIPTION 2

2.1 FEATURE OVERVIEW 2

2.2 DESIGN STEPS 2

2.3 DESIGN CONSIDERATIONS 3

2.4 SOFTWARE FLOWCHART 3

2.5 REFERENCE CODE 3

2.6 ADDITIONAL RESOURCES 4

List of Figures

Figure 1. Application flow.....3

List of Tables

Table 1. Modules in example.....1

Table 2. Device list.....1

1 Introduce

This application note serves as a comprehensive guide for software developers, offering essential information on CRC (Cyclic Redundancy Check) algorithm. It covers fundamental concepts and provides guidelines to ensure proper utilization of CRC in software development projects. Whether you're a beginner or an experienced developer, this document will equip you with the necessary knowledge and best practices to effectively configure and utilize CRC in your applications.

1.1 Required peripherals

This application involves modules as table 1.

Table 1. Modules in example

Sub-module	Peripheral use	Note
CRC	generates a checksum	

1.2 Compatible devices

This example is compatible with the devices in Table 2.

Table 2. Device list

Product	EVB
XT32H050	XB002823

2 Design description

2.1 Feature overview

CRC module supports four polynomials, such as CRC32 (MSB 0x04C11DB7), CRC16 (MSB 0x1021), CRC16 (MSB 0x8005), CRC8 (MSB 0x07). Input and output data can be reflected. Output data may choose XOR mask.

2.2 Design steps

1. Configure CRC algorithm parameters.

   In this example, set CRC to CRC32_MPEG2 algorithm using the CRC_POLY_32 polynomial equation, initial value is 0xffffffff, the data inversion mode is none, the output XOR mode is disable, and output XOR mask is 0x00000000.

2. Calculate and read the CRC result.

2.3 Design considerations

2.4 Software flowchart

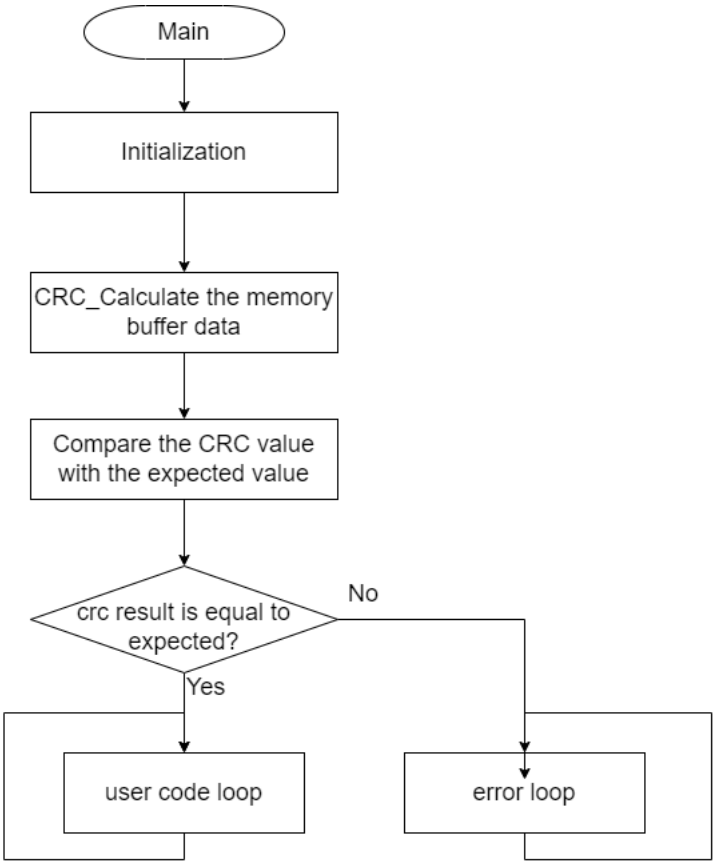

Figure 1. Application flow

2.5 Reference code

Configure Peripheral CRC using HAL_CRC_Init.

```
void XT_Crc_Init(void)
{
  /*CRC32_MPEG2 */
  hcrc.Instance = CRC;
```

```
  hcrc.Init.CRCPoly = CRC_POLY_32;
  hcrc.Init.InitValue = 0xffffffff;
  hcrc.Init.DataInversionMode = CRC_DATA_INVERSION_NONE;
  hcrc.Init.OutputXorMode = CRC_OUTPUTDATA_XOR_DISABLE;
  hcrc.Init.OutputXorMask = 0x00000000;
  hcrc.InputDataFormat = CRC_INPUTDATA_FORMAT_WORDS;
  if (HAL_CRC_Init(&hcrc) != HAL_OK)
  {
    /* Error_Handler */
  }
}
```

XT_Crc_Task handles the CRC calculattion process.

```
void XT_Crc_Task(void)
{
    /* USER CODE */
    uwCRCValue = HAL_CRC_Calculate(&hcrc, (uint32_t *)a32DataBuffer,
CRC_TEST_BUFFER_SIZE /*BUFFER_SIZE*/);
    /* Compare the CRC value with the Expected  */
    if (uwCRCValue != CRC32MPEG2_EXPECTEDVALUE)
    {
        /* Wrong CRC value: enter Error_Handler */
        Error_Handle();
    }
}
```

2.6 Additional resources

- XT32H0xxB--reference manual