KingKong.tech
金 钢 科 技

# ДВОЙНЫЕ МАГНИТНЫЕ ЭНКОДЕРЫ СЕРИИ DPT V0.3

KingKong.tech
金 钢 科 技

# Двойные магнитные энкодеры серии DPT

**Двойной магнитный энкодер серии DPT предназначен для определения углового положения манипуляторов роботов. Принцип работы датчика основан на технологии определения характеристик магнитного поля KingKong. Даже при наличии пространственных ограничений датчик точно определяет взаимное положение двух осей, демонстрируя при этом разрешение и точность, аналогичные оптическим системам. Он также отличается высокой помехоустойчивостью.**

Датчик сочетает в себе две уникальные технологии: высокоточную технологию измерения величины магнитного поля и технологию экранирования помех. Сканирование пространственного распределения магнитного поля намагниченного кольца, находящегося на подвижной части, осуществляется при помощи высокоточных датчиков Холла внутри устройства. При этом точность калибровки достигается за счет использования технологии KingKong. Каждое из устройств поставляется с уникальным набором данных калибровки магнитного поля, что обеспечивает оптимальную точность измерений в ходе дальнейшей эксплуатации.

Предложенная технология не накладывает жестких ограничений на монтаж устройства на движущейся или неподвижной частях, упрощая тем самым процедуру монтажа для пользователя, но сохраняя при этом точность бесконтактных измерений.

Такая комбинация с отдельной установкой датчика Холла и магнита представляет собой эффективное решение и обеспечивает стойкость к воздействию факторов окружающей среды, таких как вибрация, а также наличие пыли и масляных загрязнений. Датчик позволяет выполнять измерения при сверхвысоких скоростях, которые абсолютно не влияют на его точность и срок службы.

Интеграция в систему такой сверхтонкой и миниатюрной полой конструкции не составит труда независимо от варианта применения.

- Два выходных сигнала, 24 бита, при абсолютных измерениях
- Двойная точность позиционирования ±0.01° при абсолютных измерениях
- Миниатюрное устройство (ширина кольца по радиусу 7 мм), модель построена для внутреннего диаметра кольца с шагом 5 мм
- Технология не предусматривает жестких монтажных допусков при установке устройства на движущейся и неподвижной частях
- Технология экранирования магнитных помех
- Конструктивное исполнение с полым валом позволяет не накладывать пространственных ограничений при монтаже
- Максимальная скорость 8000 об/мин
- Уникальный набор данных, получаемый в процессе калибровки
- Выходной сигнал на основе нескольких интерфейсов
- Устойчивость к воздействиям окружающей среды

# МОДЕЛИ

## DPT-10-15-25-A-24-R-N-A-M

**Внутренний диаметр внутреннего ротора**
Пожалуйста, проверьте размер для получения подробной информации

**Внутренний диаметр наружного ротора**
Пожалуйста, проверьте размер для получения подробной информации

**Внешний диаметр статора**
Пожалуйста, проверьте размер для получения подробной информации

**Тип выхода**
A – абсолютный

**Выходные параметры внутреннего энкодера**
24–24 бита

**Выходные параметры внешнего энкодера**
24–24 бита

**Выходной интерфейс**
R – RS485
B – BISS-C

**Входное напряжение**
N – 5 В

**Рабочая температура**
A – от -40 до 85 °C
B – от -40 до 105 °C
C – от -40 до 125 °C

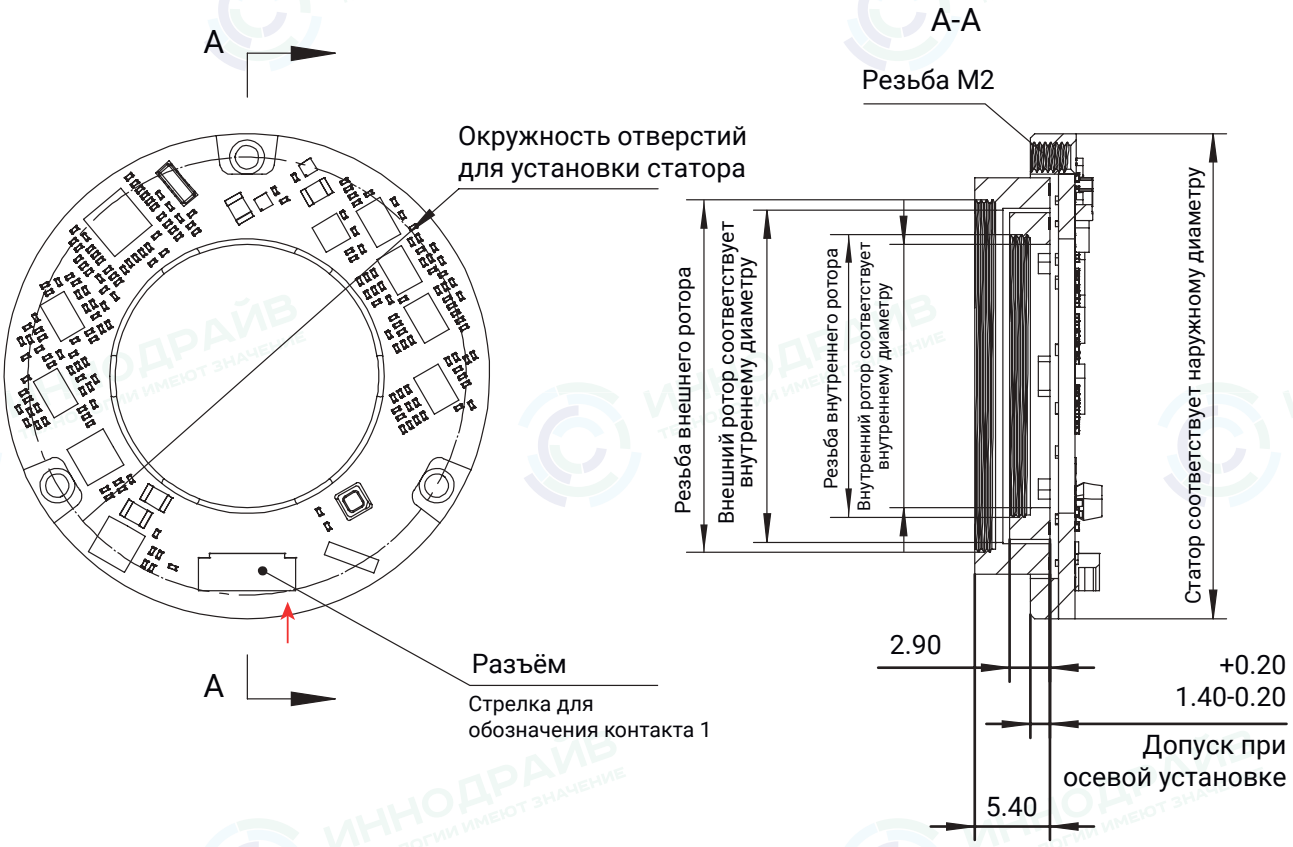
**Дополнения**
M – без доп.

# РАЗМЕР

| СЕРИЯ | РАЗМЕР | РЕЗЬБА ВНУТРЕННЕГО РОТОРА | ВНУТРЕННИЙ РОТОР СООТВЕТСТВУЕТ ВНУТРЕННЕМУ ДИАМЕТРУ (H7) | РЕЗЬБА ВНЕШНЕГО РОТОРА | ВНЕШНИЙ РОТОР СООТВЕТСТВУЕТ ВНУТРЕННЕМУ ДИАМЕТРУ (H7) | СТАТОР СООТВЕТСТВУЕТ ВНЕШНЕМУ ДИАМЕТРУ (H7) |
|---|---|---|---|---|---|---|
| 10-15-25 | DPT-10-15-25 | M10x0.4 мм | 9 | M15x0.4 мм | 14 | 25 |
| 15-20-30 | DPT-15-20-30 | M15x0.4 мм | 14 | M20x0.4 мм | 19 | 30 |
| 20-25-35 | DPT-20-25-35 | M20x0.4 мм | 19 | M25x0.4 мм | 24 | 35 |
| 25-30-40 | DPT-25-30-40 | M25x0.4 мм | 24 | M30x0.4 мм | 29 | 40 |
| 30-35-45 | DPT-30-35-45 | M30x0.4 мм | 29 | M35x0.4 мм | 34 | 45 |

1. Скачать 3D-модели: https://kingkong.tech/encoder/

# ЧЕРТЕЖИ




Для установки статора можно использовать винты M1.6, проходящие через отверстия с резьбой M2, или использовать резьбу M2 с винтами M2 для фиксации. Для установки ротора необходимо использовать специальные инструменты для затягивания ротора, подробности см. в «Рекомендации по проектированию и установке DPT».

# ЭЛЕКТРИЧЕСКОЕ СОЕДИНЕНИЕ

## Разъёмы

| | Разъемы для соединения проводов с платой |
|---|---|
| **Модели** | SM06B-SURS-TF |
| **Тип** | Провод к плате |
| **Провод** | Жгут проводов AWG32 |

## Вывод

| Вывод | Цвет | R | B |
|---|---|---|---|
| | | **RS485** | **BISS-C** |
| **1** | красный | +5 В | |
| **2** | чёрный | 0 В (GND) | |
| **3** | синий | A | MA + |
| **4** | зеленый | B | MA - |
| **5** | желтый | — | SLO + |
| **6** | оранжевый | — | SLO - |

# ХАРАКТЕРИСТИКИ ПАРАМЕТРОВ

## Параметры системы

| | |
|---|---|
| Способ установки | Полость осевого направления |
| Разрешение | 24 бита |
| Точность | ±0,01° |

## Электрические параметры

| | |
|---|---|
| Источник питания | 4,5–5,5 В |
| Время пуска | 15 мс |
| Способ подключения | Разъемы для соединения проводов с платой |
| Сила тока | ≈130 мА |
| Электростатическая защита | Модель человеческого тела (HBM), макс. ±2 кВ<br>Модель заряженного устройства (CDM), макс. ±1 кВ |

## Механические параметры

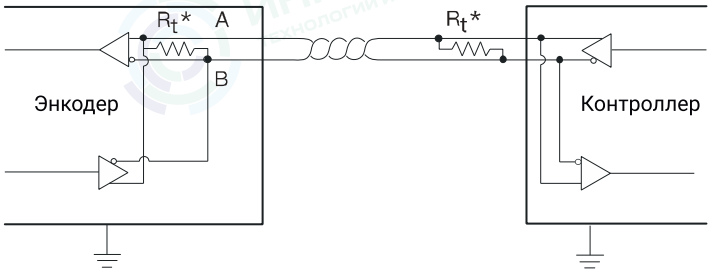
| | |
|---|---|
| Кронштейн для магнитного кольца | нержавеющая сталь |

## Параметры окружающей среды

| | |
|---|---|
| Рабочая температура | от -40 до 85 °C / от -40 до 105 °C / от -40 до 125 °C |

## Интерфейс протокола RS485

Схема электрического подключения RS485:




Интерфейс представляет собой двухпроводную систему, которая в основном состоит из дифференциальных фаз A и B. Клеммы двух линий должны быть подключены к согласующим клеммным резисторам. Согласующие клеммные резисторы на стороне энкодера встроены в энкодер. Пользователю необходимо подключить клеммные резисторы или другие схемы согласования импеданса на стороне контроллера A и B.

Базовым протоколом RS485 является UART. Поскольку в этом протоколе отсутствует тактовая линия, для завершения передачи данных эндкодер и контроллер должны работать на одной и той же частоте и в одном и том же формате данных.

Конфигурация протокола:

| Длина символа | Контроль по четности | Стоповый бит | Управление потоком | Порядок следования байтов |
|---|---|---|---|---|
| 8 бит | нет | 1 | нет | LSB первый |

Поддерживаемые скорости передачи данных
(по умолчанию и рекомендуемая B, если не отмечено в дополнении):

| Код | B | C | C |
|---|---|---|---|
| Скорость передачи данных (Мбит/с) | 2,5 | 5 | 7,75 |

## Инструкции и данные:

| Команда | О команде | | N | Возвращаемые данные (N байт) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
| 0x29 | Настройка | Внутреннее нулевое значение (1) | 2 | C | CRC | | | | | | |
| 0x30 | Настройка | Внешнее нулевое значение (2) | 2 | C | CRC | | | | | | |
| 0x31 | Получение | Внутренний угол | 4 | A0 | A1 | A2 | CRC | | | | |
| 0x32 | Получение | Внутренний угол | 4 | B0 | B1 | B2 | CRC | | | | |
| 0x33 | Получение | Внутренний угол Внешний угол | 7 | A0 | A1 | A2 | B0 | B1 | B2 | CRC | |
| 0x41 | Получение | Внутренний угол Информация о состоянии | 5 | A0 | A1 | A2 | S | CRC | | | |
| 0x42 | Получение | Внешний угол Информация о состоянии | 5 | A0 | A1 | A2 | S | CRC | | | |
| 0x42 | Получение | Внешний угол Информация о состоянии | 8 | A0 | A1 | A2 | B0 | B1 | B2 | S | CRC |
| 0x42 | Получение | Информация о состоянии | 3 | T0 | T1 | CRC | | | | | |

В приведенной выше таблице буквы соответсвуют следующим данным::

| A | B | C | S | T | CRC |
|---|---|---|---|---|---|
| Внутренний угол энкодера | Внешний угол энкодера | Сбросить на ноль величину отсчёта | Состояние | Температура | Проверка CRC |

(1) Чтобы установить нулевое положение, вам необходимо отправить команду 0x31 и команду 0x29 с последовательными интервалами в общей сложности 10 раз, чтобы успешно установить их; когда значение обратного отсчета равно 10, срабатывает нулевое положение.
(2) Чтобы установить нулевое положение, вам необходимо отправить команду 0x32 и команду 0x30 с последовательными интервалами в общей сложности 10 раз, чтобы успешно установить их; когда значение обратного отсчета равно 10, срабатывает нулевое положение.
(3) Информация о температуре представляет собой температуру перехода чипа (это значение равно °C*10).
(4) Число небольшое, в порядке байтов
(5) Байт CRC (полином CRC x8+x7+x4+x2+x1+1, метод расчета приведен в таблице приложения CRC-8 (x8+x7+x4+x2+x1+1)

**Пример:**

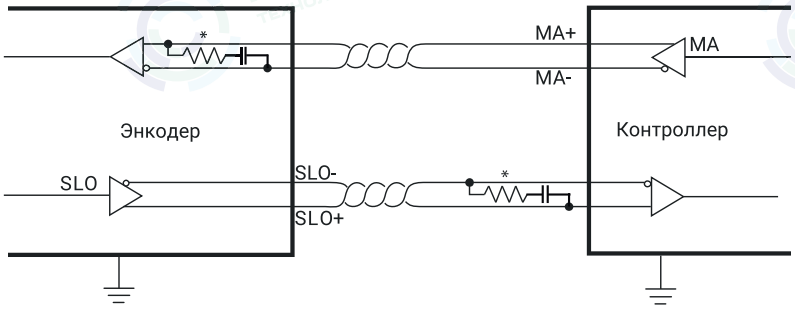
Например, если отправлено 0x33, будет получен uint8_t Buffer:7];
При использовании:

```
uint32_t angleInner = Buffer[2] << 16 | buffer[1] << 8 | buffer[0];
uint32_t anngleOutter = Buffer[5] << 16 | buffer[4] << 8 | buffer[3];
float angleInnerFloat = angleInner / (float)(1 << 24) * 360;
float angleOutterFloat = angleOutter / (float)(1 << 24) *360;
```

BISS-C

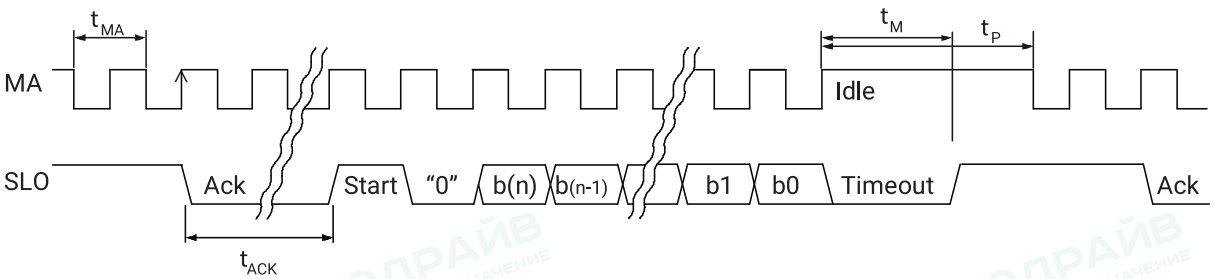
## Интерфейс протокола BISS-C

Схема электрического подключения BiSS-C:




Интерфейс использует четырехпроводную систему с двумя каналами: MA и SLO. Терминальный резистор Терминальный резистор MA встроен в энкодер. Пользователю необходимо подключить терминальный резистор или другую схему согласования импеданса к выводам SLO на стороне контроллера.

### Временная диаграмма:




Протокол использует MA для синхронизации тактовой последовательности, линия MA находится на высоком уровне в состоянии покоя, когда приходит первый спадающий фронт, система фиксирует текущие данные, обмен данными начинается по первому спадающему фронту, эндкодер конфигурирует SLO для низкого уровня по второму спадающему фронту MA, и MSB будет записывать данные в линию SLO по нарастающему фронту каждого MA, начиная с 0, а контроллер будет читать данные в линии SLO по спадающему фронту MA, и так далее, пока LSB не будет прочитан контроллером. После «0» MSB начинает записывать данные в линию SLO по нарастающему фронту каждого MA, а со стороны контроллера данные на линии SLO будут считываться по спадающему фронту MA, и так повторяется до тех пор, пока контроллер не прочитает LSB.

### Временная диаграмма:

| Параметр | Символ | Минимальное значение | Общепринятое значение | Максимальное значение |
|---|---|---|---|---|
| тактовая частота | tMA | 400 нс | | 14 мкс |
| тактовая частота | F | 120 кГц | | 2.5 МГц(1) |
| длина ACK | tACK | | 5 бит | |
| превышение времени передачи | tM | | 10 мкс | |
| время паузы | tP | 20 мкс | | |

(1) Если у пользователя есть технология фазовой компенсации для компенсации задержки между дифференциальными преобразованиями, частота может достигать 10 МГц

После завершения передачи, когда закончится время передачи $t_M$, линия SLO обработает высокий уровень, и сигнал MA должен оставаться высоким до тех пор, пока не будет разрешено следующее считывание, то есть по истечении времени $t_P$. $t_{CL}$ должно быть меньше $t_M$, и когда выполняется какая-либо операция чтения, время может превысить $t_M$ для завершения чтения.

Формат данных:

| **Бит** | b55 : b32 | b31 : b8 | b7 | b6 | b5 : b0 |
|---|---|---|---|---|---|
| **Длина** | 24 бита | 24 бита | 1 бит | 1 бит | 6 битов |
| **Данные** | внутренний угол | внешний угол | бит ошибки (1) | бит предупреждения (2) | CRC(3) |

(1)Бит ошибки активен на низком уровне.

(2) Бит предупреждения активен на низком уровне. Когда он равен 1, это означает, что никаких ошибок или предупреждений не возникает; когда он равен 0, это означает, что возникает хотя бы одна ошибка или предупреждение.

(3) Полином CRC равен x6+x1+1 (т. е. 0x43). В соответствии с требованиями протокола BISS-C вычисленный CRC будет инвертирован перед отправкой. Приложение «Расчет CRC-6» содержит код расчета, который может быть непосредственно перенесен для удобства использования.

Описание битов состояния см. в разделе «Биты состояния» далее в этом документе.

## Бит состояния

В протоколах SSI/BISS-C/RS485/RS422 используются одинаковые биты состояния. При появлении предупреждения или ошибки на выходе отображается бит предупреждения или бит положения ошибки, а затем бит состояния, чтобы четко понять причину текущей настройки.

Где находятся ошибки/предупреждения в каждом интерфейсе:

| | Бит ошибки | Бит предупреждения |
|---|---|---|
| **BISS-C** | b13 | b12 |
| **RS485** | b7 | b6 |

Бит состояния:

| **Расположение** | b3 | b2 | b1 | b0 |
|---|---|---|---|---|
| **Описание** | Ротор с внешним кольцом находится слишком далеко | Ротор с внешним кольцом находится слишком близко | Ротор с внутренним кольцом находится слишком далеко | Ротор с внутренним кольцом находится слишком близко |

Когда бит предупреждения равен 1, данные все еще действительны. В это время индикатор состояния горит желтым, но некоторые параметры в бите состояния близки к предельному значению, которое можно просмотреть с помощью бита состояния; когда бит ошибки равен 1, данные недействительны, тогда индикатор состояния горит красным, вы можете просмотреть конкретную ситуацию с помощью бита состояния; при нормальной работе индикатор состояния горит зеленым.

# Приложение

## Таблица CRC-8 (x8+x7+x4+x²+x¹+1)

```
//poly = x 8+x7+x4+x2+x1+1
uint8_t crcTable [256] = {
        0x00, 0x97, 0xB9, 0x2E, 0xE5, 0x72, 0x5C, 0xCB, 0x5D, 0xCA, 0xE4, 0x73, 0xB8, 0x2F, 0x01, 0x96,0xBA, 0x2D, 0x03, 0x94,
0x5F, 0xC8, 0xE6, 0x71, 0xE7, 0x70, 0x5E, 0xC9, 0x02, 0x95, 0xBB, 0x2C,0xE3, 0x74, 0x5A, 0xCD, 0x06, 0x91, 0xBF, 0x28, 0xBE,
0x29, 0x07, 0x90, 0x5B, 0xCC, 0xE2, 0x75,0x59, 0xCE, 0xE0, 0x77, 0xBC, 0x2B, 0x05, 0x92, 0x04, 0x93, 0xBD, 0x2A, 0xE1, 0x76,
0x58, 0xCF,0x51, 0xC6, 0xE8, 0x7F, 0xB4, 0x23, 0x0D, 0x9A, 0x0C, 0x9B, 0xB5, 0x22, 0xE9, 0x7E, 0x50, 0xC7,0xEB, 0x7C, 0x52,
0xC5, 0x0E, 0x99, 0xB7, 0x20, 0xB6, 0x21, 0x0F, 0x98, 0x53, 0xC4, 0xEA, 0x7D,0xB2, 0x25, 0x0B, 0x9C, 0x57, 0xC0, 0xEE, 0x79,
0xEF, 0x78, 0x56, 0xC1, 0x0A, 0x9D, 0xB3, 0x24,0x08, 0x9F, 0xB1, 0x26, 0xED, 0x7A, 0x54, 0xC3, 0x55, 0xC2, 0xEC, 0x7B, 0xB0,
0x27, 0x09, 0x9E,0xA2, 0x35, 0x1B, 0x8C, 0x47, 0xD0, 0xFE, 0x69, 0xFF, 0x68, 0x46, 0xD1, 0x1A, 0x8D, 0xA3, 0x34,0x18, 0x8F,
0xA1, 0x36, 0xFD, 0x6A, 0x44, 0xD3, 0x45, 0xD2, 0xFC, 0x6B, 0xA0, 0x37, 0x19, 0x8E,0x41, 0xD6, 0xF8, 0x6F, 0xA4, 0x33, 0x1D,
0x8A,  0x1C, 0x8B, 0xA5, 0x32, 0xF9, 0x6E, 0x40, 0xD7,0xFB, 0x6C, 0x42, 0xD5, 0x1E, 0x89, 0xA7, 0x30, 0xA6, 0x31, 0x1F, 0x88,
0x43, 0xD4, 0xFA, 0x6D,0xF3, 0x64, 0x4A, 0xDD, 0x16, 0x81, 0xAF, 0x38, 0xAE, 0x39, 0x17, 0x80, 0x4B, 0xDC, 0xF2, 0x65,0x49,
0xDE, 0xF0, 0x67, 0xAC, 0x3B, 0x15, 0x82, 0x14, 0x83, 0xAD, 0x3A, 0xF1, 0x66, 0x48, 0xDF,0x10, 0x87, 0xA9, 0x3E, 0xF5, 0x62,
0x4C, 0xDB, 0x4D, 0xDA, 0xF4, 0x63, 0xA8, 0x3F, 0x11, 0x86,0xAA, 0x3D, 0x13, 0x84, 0x4F, 0xD8, 0xF6, 0x61, 0xF7, 0x60, 0x4E,
0xD9, 0x12, 0x85, 0xAB, 0x3C
};

uint8_t calcCRC(uint8_t * buffer, uint8_t length){
        uint8_t temp = *buffer++;
        while(–  length){
                temp = *buffer++ ^ crcTable[temp];
        }

        return crcTable[temp];
}
```

# Расчет CRC-6

```
#define DATA_TOTAL_BIT_LENGTH 47


//poly = x^6+x^1+1
uint8_t  tableCRC6[64] = {
0x00, 0x03, 0x06, 0x05, 0x0C, 0x0F, 0x0A, 0x09, 0x18, 0x1B, 0x1E, 0x1D, 0x14, 0x17, 0x12, 0x11, 0x30, 0x33, 0x36, 0x35, 0x3C, 0x3F, 0x3A, 0x39, 0x28, 0x2B, 0x2E, 0x2D, 0x24, 0x27, 0x22, 0x21, 0x23,
0x20, 0x25, 0x26, 0x2F, 0x2C, 0x29, 0x2A, 0x3B, 0x38, 0x3D, 0x3E, 0x37, 0x34, 0x31, 0x32, 0x13, 0x10, 0x15, 0x16, 0x1F, 0x1C, 0x19, 0x1A, 0x0B, 0x08, 0x0D, 0x0E, 0x07, 0x04, 0x01, 0x02
};

uint8_t calcBissCCRC(uint8_t buffer[]){
	#define CRC_BIT_LENGTH 6
	#define DATA_CRC_MASK ((1 << CRC_BIT_LEN    GTH) -  1)
	#define DATA_WITHOUT_CRC_BIT_LENGTH (DATA_TOTAL_BIT_LENGTH  -  CRC_BIT_LENGTH)
	#define TOP_BYTE_BITLENGTH (DATA_WITHOUT_CRC_BIT_LENGTH % CRC_BIT_LENGTH)
	#if TOP_BYTE_BITLENGTH == 0
		#undef TOP_BYTE_BITLENGTH
		#define TOP_BYTE_BITLENGTH    CRC_BIT_LENGTH
	#endif

	uint32_t firstWord = __REV(*(uint32_t *) buffer);
	#if DATA_WITHOUT_CRC_BIT_LENGTH > 32
	uint32_t secondWord = __REV(*(uint32_t *) (buffer + 4));
	#endif

	uint8_t crc = tableCRC6[firstWord >> (32  -   TOP_BYTE_BITLENGTH)];

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 1)
	#if DATA_WITHOUT_CRC_BIT_LENGTH -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (firstWord >> (32 -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 2)
	#if DATA_WITHOUT_CRC_BIT_LENGTH -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (firstWord >> (32 - CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 3)
	#if DATA_WITHOUT_CRC_BIT_LENGTH    -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (firstWord >> (32 - CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 4)
	#if DATA_WITHOUT_CRC_BIT_LENGTH    -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (firstWord     >> (32  -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 5)
	#if DATA_WITHOUT_CRC_BIT_LENGTH    -  CURRENT_CRC_BIT_LENGTH >= 0
		#if 32  -  CURRENT_CRC_BI  T_LENGTH >= 0
			crc = tableCRC6[crc ^ (firstWord >> (32 -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
		#elif 32  -  CURRENT_CRC_BIT_LENGTH >  - CRC_BIT_LENGTH
			crc = tableCRC6[crc ^ (((firstWord << - (32 -  CURRENT_CRC_BIT_LENGTH)) & DATA_CRC_MASK) | (secondWord >> (64   -  CURRENT_CRC_BIT_LENGTH)))];
		#else
			crc = tableCRC6[crc ^ (secondWord >> (64 -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
		#endif
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 6)
	#if DATA_WITHOUT_CRC_BIT_LENGTH    -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (secondWord >> (64 -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 7)
	#if DATA_WITHOUT_CRC_BIT_LENGTH -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (secondWord >> (64-  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_BIT_LENGTH    (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 8)
	#if DATA_WITHOUT_CRC_BIT_LENGTH -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (secondWord >> (64 -  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#undef CURRENT_CRC_BIT_LENGTH
	#define CURRENT_CRC_B   IT_LENGTH (TOP_BYTE_BITLENGTH + CRC_BIT_LENGTH * 9)
	#if DATA_WITHOUT_CRC_BIT_LENGTH -  CURRENT_CRC_BIT_LENGTH >= 0
		crc = tableCRC6[crc ^ (secondWord >> (64-  CURRENT_CRC_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	#if 32  -  DATA_TOTAL_BIT_LENGTH >= 0
		crc  = tableCRC6[crc ^ DATA_CRC_MASK ^ (firstWord >> (32 -  DATA_TOTAL_BIT_LENGTH) & DATA_CRC_MASK)];
	#elif 32  -  CURRENT_CRC_BIT_LENGTH >    - CRC_BIT_LENGTH
		crc = tableCRC6[crc ^ DATA_CRC_MASK ^ (((firstWord <<- (32 -  DATA_TOTAL_BIT_LENGTH)) & DATA_CRC_MASK) | (secondWord >> (64   -  DATA_TOTAL_BIT_LENGTH)))];
	#else
		crc = tableCRC6[crc ^ DATA_CRC_MASK ^ (secondWord >> (64 -  DATA_TOTAL_BIT_LENGTH) & DATA_CRC_MASK)];
	#endif

	return crc;
}
```

## Инструкция по применению:

Эту программу можно применить к микроконтроллерам серии Arm. Она может генерировать самую быструю проверку CRC-6 с помощью компилятора.

Просто измените DATA_TOTAL_BIT_LENGTH на значение соответствующей модели

Например: 17M – 47, 16 – 30.

## Примечания по использованию:

Во время вызова для буфера используется 32-битная инструкция чтения, которая требует, чтобы буфер был выровнен по 4 байтам (некоторые версии ядра не поддерживают это, если он не выровнен; даже если оно поддерживает чтение без выравнивания, ядро тратит лишнее время на сращивание).

Для приема BISS-C первый полученный байт будет байтом-заполнителем с подтверждением ACK, а данные о местоположении начинаются со второго байта, поэтому мы должны вычислить CRC по адресу, с которого начинаются данные о местоположении, и адрес должен быть в режиме выравнивания по 4 байтам, для достижения первоначальной цели – быстрого считывания данных и вычисления CRC.

Например, случай вызова:

```
struct{

uint8_t notUsedForAlignment[3];          // Просто выравнивание адреса
uint8_t placeholder;                     //Первый фиксированный байт-заполнитель BISS-C 0x82
uint8_t buffer[8] _attribute_ ((aligned(4)));   //4-байтовый буфер с выравниванием по словам, удобный для последующего быстрого CRC

} receiveBuffer;

//Настройка SPI и DMA
//Использование &receiveBuffer.placeholder в качестве адреса получения
//……


//Расчет CRC
//Использование выровненного по 4 словам getBuffer.buffer для вычисления
uint8_t crc = calcBissCCRC(receiveBuffer.buffer);


//результат crc равен 0, что означает, что проверка пройдена
if ( crc != 0 ){
//проверка crc не удалась
}
```

# История версий

| Время | Версия | Детали изменений |
|---|---|---|
| 2023/09/01 | V0.1 | Начальная версия |
| 2023/03/18 | V0.2 | Обновление битов состояния<br>Добавление протокола BISS-C |
| 2023/05/01 | V0.3 | Добавление стандартных чертежей<br>Добавление цвета кабелей |

KingKong.tech
金 钢 科 技

ИННОДРАЙВ
ТЕХНОЛОГИИ ИМЕЮТ ЗНАЧЕНИЕ

ООО «ИнноДрайв» - официальный дистрибьютор
KingKong Technology

**+7 (812) 317-77-93**
www.innodrive.ru
sales@innodrive.ru