April 2021

Henne's Sites

AN012: DMX-Empfang mit AVR

Inhalt

Einleitung

In dieser Application Note wird der Empfang eines DMX-Signals mit Hilfe von AVR-Mikrocontrollern beschrieben. Die state machine wurde in C für den DMX-Transceiver von Henne's Sites geschrieben, sollte sich aber auch auf andere Schaltungen portieren lassen.

Nutzungsbedingungen

DMX 512

DMX 512 ist ein unidirektionales differentielles serielles Protokoll mit einer Übertragungsrate von 250kBaud. Ein Byte besteht aus einem Startbit [4], acht Datenbits [14, 15] und zwei Stoppbits [7, 8]. Es gibt kein Paritätsbit (8n2). Zwischen den Bytes können inter byte gaps [9] eingeschoben werden, um auch langsameren Empfängern einen stabilen Empfang zu ermöglichen.

Es sind ein Master und bis zu 32 Slaves in einem Bus vorgesehen. Die Zahl der Slaves kann durch Einsatz von Splittern oder Boostern erhöht werden. Ein Universum kann bis zu 512 Kanäle haben. Die Übertragung wird durch einen Break [1] von mind. 88µs Länge gefolgt von einem Mark-after-Break [2] von mind. 8µs Länge eingeleitet. Anschließend wird das Startbyte [14] übertragen, das bei DMX den Wert Null hat.

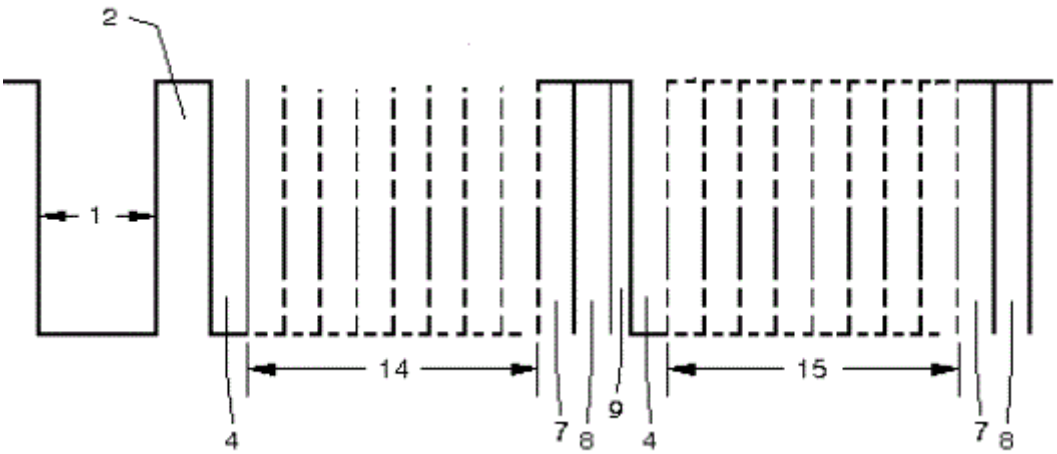

Abbildung 1: Aufbau des DMX-Protokolls

Beschreibung des Codes

Nach dem Einlesen des Datenbytes und des Statusregisters wird ein "Framing-Error" festgestellt, der auch ein Break sein könnte. Es wird zunächst davon ausgegangen, dass es sich um einen BREAK handelt. Dies wird im DMX Status vermerkt und zum Hauptprogramm zurückgekehrt.

Falls der vorige Framing-Error wirklich ein Break war, dürfte das nächste Byte ein Startbyte mit einem Wert von 0 sein. Ist dies so, wird der Status inkrementiert. Falls das Startbyte ungleich 0 ist, gab es einen echten Übertragungsfehler, der Empfang wird abgebrochen und bis zum nächsten BREAK gewartet.

Im Folgenden wird so lange mit jedem eintreffenden Byte der Byte-Counter dekrementiert, bis die Startadresse erreicht ist. In diesem Fall wird wieder der Status inkrementiert.

Bei den folgenden Bytes handelt es sich um die gewünschten Daten. Sind alle Daten empfangen, wird der DMX Status auf zurückgesetzt und auf den nächsten BREAK gewartet.

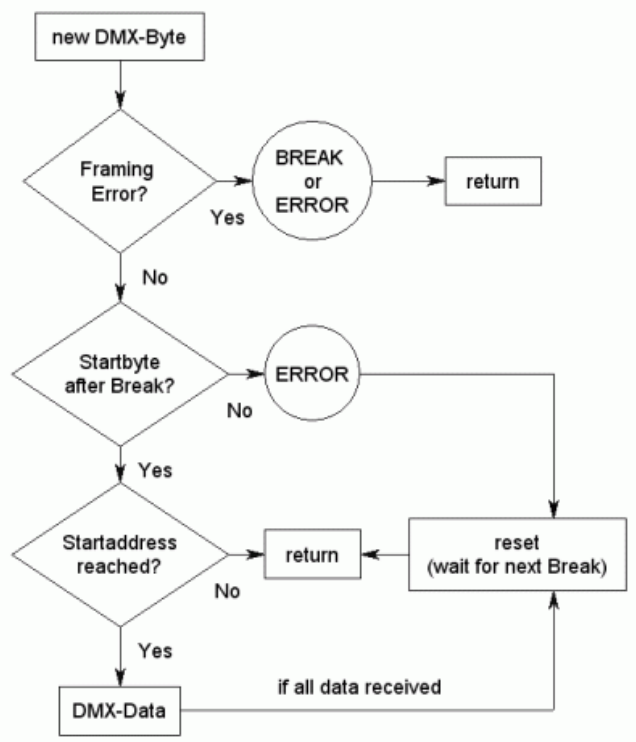

Abbildung 2: Flussdiagramm der state machine

Der Code wurde geschrieben im ATMEL Studio 7.
In der Headerdatei müssen folgende Variablen und Konstanten definiert sein:

```
#define  F_CPU    (8000000UL)                  //oscillator freq.(typical 8MHz or 16MHz)
#define  USE_DIP                               //get start address from DIPs

volatile  uint8_t    DmxRxField[8];            //array of DMX vals (raw)
volatile  uint16_t   DmxAddress;               //start address

extern void          init_DMX_RX(void);
extern void          get_dips(void);
```

F_CPU ist die Quarzfrequenz in Hz.

Wenn USE_DIP definiert ist, wird die Startadresse von den DIP-Schaltern des Transceivers bezogen. Andernfalls kann die Startadresse auch direkt in DmxAddress abgelegt werden.

Die DMX-Daten werden in DmxRxField[] gespeichert.

Mit „init_DMX_RX()“ wird der USART für den DMX-Empfang initialisiert und der Empfangspuffer auf 0 gesetzt.

Mit „get_dips()“ wird aus dem Status der DIP-Schalter des Transceivers die Startadresse bestimmt.