Send command sequence erklärt

Diese Funktion `SendCommandSeq` sendet eine Sequenz von 16-Bit-Kommandos über die SPI-Schnittstelle. Hier ist, was in den einzelnen Schritten passiert:

1. `uint32_t index;` definiert eine Indexvariable für die Schleife.
2. `uint8_t SendeByte;` definiert eine temporäre 8-Bit-Variable zum Senden der Bytes.
3. `for (index = 0; index < Anzahl; index++)` ist eine Schleife, die so oft durchlaufen wird, wie die `Anzahl` der Kommandos angibt.
4. `PORTD |= (1 << D_C);` setzt den Data/Command-Pin auf "hoch", was den Kommando-Modus signalisiert.
5. `SendeByte = (data[index] >> 8) & 0xFF;` extrahiert das höherwertige 8-Bit-Byte des aktuellen 16-Bit-Kommandos.
6. `SPISend8Bit(SendeByte);` sendet das höherwertige Byte über die SPI-Schnittstelle.
7. `SendeByte = data[index] & 0xFF;` extrahiert das niedrigwertige 8-Bit-Byte des aktuellen 16-Bit-Kommandos.
8. `SPISend8Bit(SendeByte);` sendet das niedrigwertige Byte über die SPI-Schnittstelle.
9. `PORTD &= ~(1 << D_C);` setzt den Data/Command-Pin auf "niedrig", was den Daten-Modus signalisiert.

Die Funktion iteriert also über ein Array von 16-Bit-Kommandos und sendet jedes Kommando als zwei 8-Bit-Bytes über die SPI-Schnittstelle. Vor dem Senden eines Kommandos wird der Data/Command-Pin auf "hoch" gesetzt, um den Kommando-Modus zu signalisieren. Nach dem Senden wird der Pin wieder auf "niedrig" gesetzt für den Daten-Modus.

Diese Art der Kommando-Sequenz-Übertragung ist typisch für die Kommunikation mit bestimmten Peripheriegeräten wie Displays, die eine Sequenz von Initialisierungs- oder Konfigurationsbefehlen benötigen, bevor Daten gesendet werden können.