Bau und Programmierung eines Roboters für ROBOCUP JUNIOR RESCUE Teil 2.2: Interrupt

Der zweite Teil des Tutorials.
Wir beschäftigen uns in diesem Video zunächst einmal mit dem Interrupt, zu deutsch Unterbrechung
Das wichtigste am Programmieren ist eine funktionierende Interruptroutine, hiermit können wir einen quasi parallelen Programmablauf generieren
Es gibt viele Möglichkeiten einen Interrupt auszulösen, extern und intern.
Zunächst behandeln wir einen internen Interrupt. Den Interrupt durch einen sog. Timer.
Der Timer 0 ist ein 8 Bit großes Register,
kann also Zahlen von Null bis 255 aufnehmen.
Wir schauen uns die Binärzahl auf dem Windowsrechner an, die Kombination aus Nullen und Einsen ergibt 172
Unser PIC arbeitet mit einer Taktfrequenz von 20 MHz, also 20 Millionen Schwingungen pro Sekunde.
Für die Ausführung eines Befehls braucht er 4 Takte,
so daß wir eine Arbeitstaktfrequenz von 5 MHZ haben.
Wir können den Timer so einstellen, dass er jeden 2. Arbeitszyklus inkrementiert bis zu jedem 256sten Takt.
Unser Timer 0 ist auf einen Prescaler von 256 eingestellt
Ein Arbeitszyklus dauert also 200ns, nach 256 Arbeitstakten wird der Timer 0 um eins erhöht, das dauert 51,2 Mikrosekunden,
Wenn das Timer 0 Register bei 255 angekommen ist
und inkrementiert wird
ist das Ergebnis 256
aber die Zahl sieht binär so aus, Unser Timer springt also von 255 auf null, wir nennen das einen Überlauf
Dieser Überlauf passiert alle 13 Millisekunden.
der Überlauf des Timers setzt automatisch ein sog. Flag, im Falle des Timer 0 das flag T M R 0 I F und löst einen Interrupt aus
das Programm wird an der aktuellen Stelle angehalten, der Pic springt zur Adresse der Interruptroutine, das Programm des Interrupts wird abgearbeitet, zum Schluss springt der Chip zurück an die Stelle hinter der Sprungadresse
Wir öffnen das nächste Projekt
Wir strukturieren jetzt unser Fenster neu, greifen, nach links ziehen, bis der rote Rahmen erscheint, los lassen
Klick, Klick, Klick, Die brauchen wir nicht mehr
Wir sehen uns den Interrupt jetzt mal im Programm an.
Wenn der TMR0 übergelaufen ist, wird das Programm unterbrochen und der PIC springt hier her
Da es einige Möglichkeiten für einen Interrupt gibt, müssen wir abfragen wer ihn ausgelöst hat
Wir fragen also, ob der Timer 0 daran Schuld hat
Wenn dies der Fall ist Inkrementieren wir eine Laufvariable T Null C K, wenn die 22 erreicht hat also 22 mal 13ms um sind, wird die Status-LED getoggelt, sprich: sie ändert ihren Zustand, ist sie aus geht sie an , Ist sie an geht sie aus
dann setzen wir unsere Laufvariable noch auf Null
Und, ganz wichtig das Interruptflag löschen, sonst bleibt das Programm im Interrupt hängen
Da kein weiteres Flag gesetzt wurde, springt der Chip an die Stelle zurück, an der das Programm unterbrochen wurde
Die ganze Aktion benutzen wir, um die LED automatisch blinken zu lassen, wir müssen das im Rest des Programms nicht mehr berücksichtigen
Das war der Timer 0. Im nächsten Video beschäftigen wir uns mit dem I 2 C Bus.