

# **AVRCo001**

## **Sortieralgorithmen für den AVRCo**

## Einleitung:

Eine der häufigsten Aufgaben für Computer und auch Mikrocontroller ist das Sortieren. In dieser Application Note wollen wir mehrere Sortieralgorithmen vergleichen, mit dem Ziel, ein möglichst speicherplatzschonendes und schnelles Verfahren zu finden.

## Theorie:

Ausgangspunkt unserer Betrachtung ist ein eindimensionales, ungeordnetes Array des Datentyps Word mit 1000 Werten:

*SortWord : Array[0..999] of Word;*

Für die Untersuchung verwenden wir die Routine **Mischen**, die das Array mit einer bestimmten Anzahl von zufälligen Werten (Word) füllt:

*Procedure Mischen;*

*Var*

*i : Word;*

*Begin*

*For i := 0 to 999 do*

*SortWord[i] := Random;*

*end\_For;*

*end;*

Bleibt noch das Problem der Zeitmessung: Der Simulator des AVRCo stellt in der Statusleiste die verbrauchte Zeit an. Durch den „Zero“ Button kann die Uhr auf Null zurückgesetzt werden. Gemessen wurde der Eintritt und der Austritt unserer Sortieralgorithmen.

In unserem Demoprogramm wollen wir folgende Sortieralgorithmen miteinander vergleichen, als Referenz dient uns ein Mega128 mit 16MHz.

- Ø Das **Austauschverfahren** (Exchange-Sort) ist eines der einfachsten und zugleich langsamsten. Beginnend mit dem ersten Element wird jedes Element der Liste mit allen anderen Werten verglichen. Ist der Vergleichswert kleiner als der aktuelle, werden beide Werte ausgetauscht.
- Ø Das **Auswahlverfahren** ähnelt in seiner Arbeitsweise dem Austauschverfahren. Beginnend mit dem ersten Feldelement wird das Minimum gesucht, im Erfolgsfall werden die Werte getauscht. In dieser Vorgehensweise ist der Vorteil gegenüber dem Austauschverfahren zu sehen. Bei jedem Schleifendurchlauf wird maximal ein Wert getauscht.
- Ø Das **Bubble-Sort-Verfahren** ändert die bisherige Vorgehensweise dahingehend, dass der jeweils größere zweier benachbarter Werte durch das gesamte Array „geschoben“ wird. Aus diesem Vorgang resultiert die Bezeichnung Bubble-Sort ..., die Maxima steigen wie Blasen zu ihren jeweiligen Positionen auf.
- Ø Das **Shell-Sort-Verfahren** ist die schnellste aber auch die aufwendigste Sortiermethode.

### **Auswertung:**

<u>Verfahren</u>	<u>Time</u>	<u>Cycles</u>	<u>Codesize</u>
Austauschverfahren	3,257 Sekunden	52109118	\$00436
Auswahlverfahren	2,520 Sekunden	40320550	\$0044C
Bubble-Sort	4,891 Sekunden	78259611	\$00420
Shell-Sort	0,1464 Sekunden	02342049	\$004C4

Wie Sie der Auswertung entnehmen können, führen viele Wege zum Ziel, die vier Sortiervverfahren sind von der Codegröße etwa identisch, doch der Rechenaufwand für den Mikrocontroller variiert ca. um den Faktor 30.

Durch die Einbindung von Massenspeichermedien in den uC und der steigenden Anforderungen durch Kunden wird die Suche nach dem optimalen Code immer wichtiger.

Leider wird oft die Meinung vertreten, der kleinste ist auch der schnellste Code. Wie sich zeigt ist das der falsche Weg.

Viel Spaß beim spielen!!