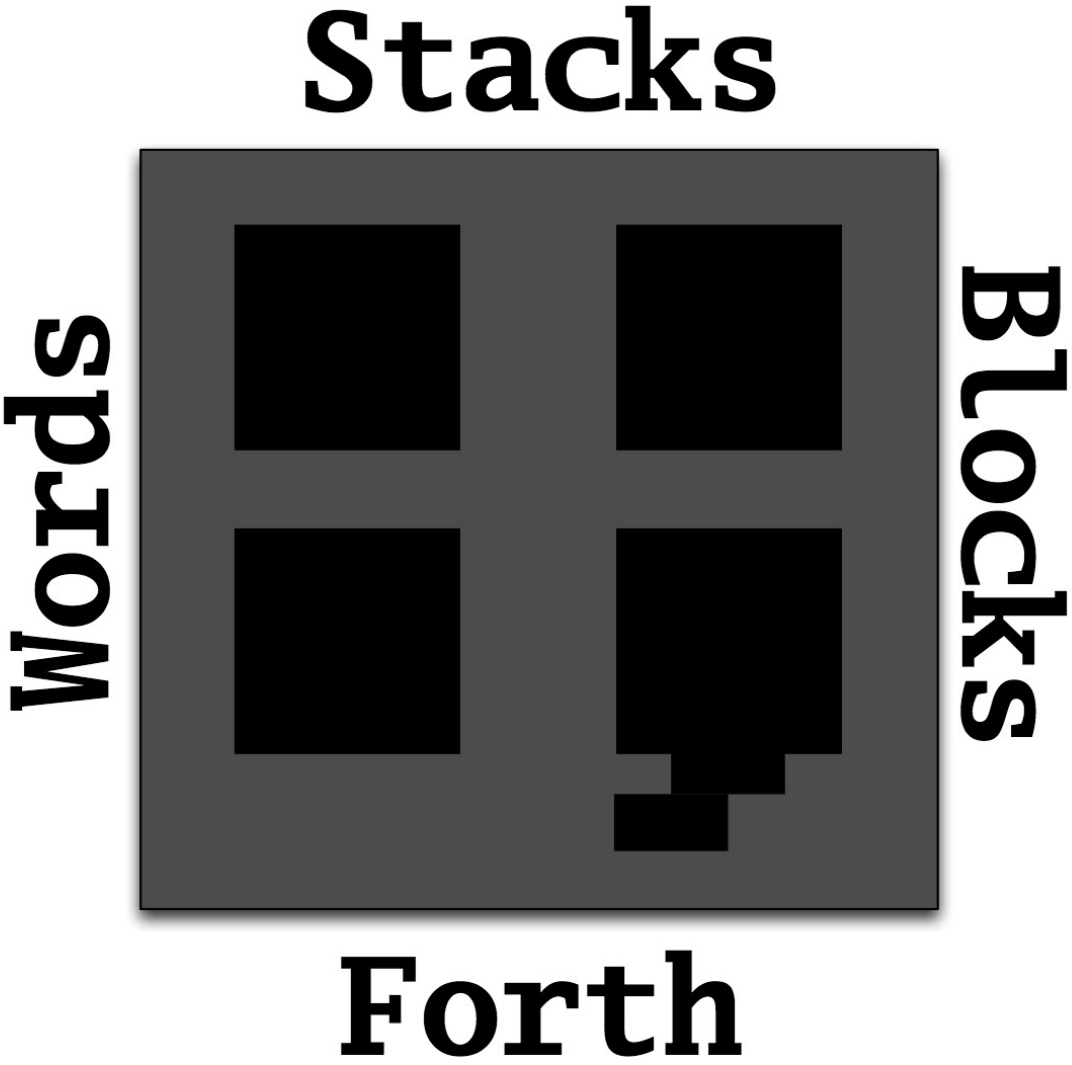
Arcus Forth 0.1 Der Full Sreen Editor - 1987 Arcussoftware (Michael Balig)

Erstellt mit Hilfe von Microsoft Word 6 und FreePDF2

Verwendete Hardware: Original ZX Spectrum 48K

Veröffentlicht im Thomas Lienhard Spectrum Forum - Februar 2021 für den Spectrum und Sam Profi Club



Hallo Arcus-Forth Fans und alle die es noch werden wollen!!

Heute geht es dann - wie ich euch in Teil 1 angedroht habe - also noch um den Full Screen Editor von Arcus-FORTH womit wir dann alle nötigen Dinge für die Bedienung dieses Programmiersystems kennengelernt haben und uns theoretisch an die Programmierung in Forth machen könnten. Der Full-Screen-Editor (FSE) - wie er so genannt wird - ist bereits im Dictionary des Arcus-FORTH enthalten und mit Ausnahme von drei sogenannten TOP-Worten in einem eigenen Vocabular untergebracht. Der Benutzer benötigt nur diese drei TOP-Worte die wie folgt lauten:

CLEAR (n ->)
Löscht/setzt den Screen Nr. n auf leer und schreibt ihn auf die interne RAM-Disk zurück. Ihr erinnert euch, das ist der Teil innerhalb des Speichers, der 16 K groß ist und alle 32 Screens enthält. Der Inhalt von Screen n wird überschrieben (sozusagen mit nichts) und ist dann leer respektive inhaltlich verloren. Das bedeutet, dass man besser gut überlegt wie man mit diesem Befehl umgeht.

COPY (n1 n2 ->)
Kopiert den Inhalt von Screen n1 auf den Screen n2 und schreibt die neue Fassung von Screen n2 zurück auf die RAM-Disk. Der alte Inhalt von Screen n2 geht dabei verloren. Das ist ähnlich wie beim CLEAR. Soweit so gut. Jetzt geht es aber wirklich (endlich?) an die Programmeingabe bzw. Programmcode.

EDIT (n ->)
Startet den Editor zur Bearbeitung des Screens n sowie evtl. weiterer Screens. Nach dem Aufruf des Editors sieht der Bildschirm beispielsweise etwa so aus (Beispiel von Michael Balig aus seiner Dokumentation entliehen):

```
Bildschirm-          Inhalt
zeile
00          arcus     Full Screen Editor      #
01          SCR * 19 ----------------------
02          (Beispiel f. Editor, Wort S)
03
04          : . S CR ." = STACK ="
os          SP$ S0 $ =
06          IF ."   ...leer."
07               10 SPACES
08          ELSE
09              SP$ 2 - S0 $ 2 -
10              DO CR
11                  1 SP$ - 2 / 7 .R
12                  1 $ 6 .R
13              -2 +LOOP
14              17 SPACES
15         ENDIF ;
16
17
18           --------------------------------
19
20           line stack:
21           ??????????????????????????????
```

Das #-Zeichen in Bildschirmzeile 00 ganz rechts stellt dabei den blinkenden Cursor dar, das den aktuellen Tastaturstatus angibt wie wir ihm vom Spectrum her schon kennen. An dieser Stelle darf angemerkt werden, dass es den „K" Status (Keyword) natürlich logischerweise nicht gegeben kann, da Forth ja keine TOKENS benutzt. Weitere Funktionen hat dieses Zeichen nicht. Der eigentliche Cursor ist eine helle Markierung am äußersten linken Rand von Zeile 2. In Arcus-FORTH handelt es sich um einen **nicht** blinkenden Cursor. Deshalb ist er auch bei schnellen Bewegungen immer zu sehen. Natürlich bedeutet das im Umkehrschluss, dass er am Anfang etwas schlecht aufzufinden ist, eben weil er nicht blinkt und wir das so von einem Cursor ja eigentlich gewohnt sind. Ab dieser Zeile 2 beginnt die Darstellung des eigentlichen Screeninhalts (hier handelt es sich beispielhafterweise um Screen Nr. 19). Mit

den uns bestens bekannten Cursortasten kann man den Cursor auf jeden beliebigen Platz auf diesem Screen bewegen und dort ein alphanumerisches Zeichen eingeben. Was zuvor an dieser Stelle gestanden hat wird gnadenlos überschrieben, d. h. der bisherige Text wird nicht nach rechts weitergeschoben. Jedoch erfolgt diese Eingabe zunächst aber nur auf dem Bildschirm und damit verbunden dem Disk-Buffer, nicht jedoch auf der RAM Disk wie vielleicht der eine oder andere von euch vermutet hätte. Dort befinden sich nur fertige Screens also Screens die nicht mehr bearbeitet werden. Die Eingabe erfolgt unter den gleichen Voraussetzungen wie im BASIC-Betrieb auch. Alle Sonderzeichen können gemäß ihrer Positionierung auf der Tastatur (durch Drücken einer oder mehrerer Funktionstasten) auch erreicht werden. Als Beispiel sei der „\" genannt, den man über die Kombination „EXTENDED MODE - SYMBOL SHIFT - D" erreicht. Einige der BASIC-Keywords erfüllen Sonderaufgaben. Wir wollten uns jedoch daran erinnern, dass Sie allerdings nicht als Tokens eingegeben werden können, da dies ausschließlich für den BASIC Interpreter vorgesehen ist. Zeichen können auf dem Screen einfach überschrieben werden. Außerdem gibt es noch eine Reihe von Funktionen im FSE, die wir jetzt kurz betrachten/kennenlernen wollen:

Gruppe C /Zeichen <>
Leerschritt einfügen. Der Rest der Zeile rückt dann um eine Stelle nach links, was bedeutet, dass das vorletzte Zeichen verloren geht, das letzte bleibt jedoch erhalten. Im Programmtext ist unbedingt darauf zu achten, dass dieses letzte Zeichen ohnehin nicht belegt sein soll, außer mit einer schließenden Kommentarklammer und diese muß erhalten bleiben.

Gruppe C / Zeichen DELETE
Das Zeichen auf der aktuellen Cursorposition wird entfernt. Der Rest der Zeile rückt eine Position von links nach. Ganz links wird dadurch ein Leerzeichen eingeschoben.

Gruppe C / Zeichen >=
Tabulierung. Der Cursor rückt in Richtung aufsteigender Position also nach rechts auf die nächste ohne Rest durch 8 teilbare Position. Die Lage der Tabulatoren ist nicht veränderbar und auf jede 8-te Position festgelegt. Das macht bei 32 Zeichen in einer Zeile natürlich nicht nur mathematisch Sinn.

Gruppe C /Zeichen =<
Das Gegenteil von eben, nämlich die Rückwärtstabulierung. Ansonsten gilt das Gleiche wie unter >= gesagt. Das bedeutet es geht nach links auf die nächste ohne Rest durch 8 teilbare Position.

Gruppe C / Zeichen MOVE
Der Cursor rückt in die linke obere Ecke des Screens. Dies ist die Position 0 auf dem Screen (siehe auch „EXTENDED MODE - SYMBOL SHIFT - 6").

Gruppe C / Zeichen ENTER
Der Cursor rückt an den Beginn der folgenden Zeile. Ganz normal, wie in anderen Editoren auch.

Gruppe Z / Zeichen TRUE VIDEO
Ab der aktuellen Cursorposition (diese berücksichtigt) werden 32 Zeichen in aufsteigender Richtung (also nach rechts) entfernt. Am Ende des Screens (nicht der Zeile!!!) werden dafür 32 Leerzeichen eingefügt. Befindet sich der Cursor an einem Zeilenanfang, dann wird wegen der Zeilenlänge von 32 Zeichen eben genau diese Zeile entfernt.

Gruppe Z / Zeichen INV VIDEO
Ab Cursorposition (inklusive) werden nun 32 Leerzeichen eingeschoben. Die letzte Zeile des Screens geht damit verloren. Befindet sich der Cursor an einem Zeilenanfang, so bedeutet

dies, dass von hier aus genau eine Zeile eingefügt wird. Diese ist im Moment noch leer, kann aber natürlich noch mit Eingaben gefüllt werden.

Gruppe Z / Zeichen OR
Die Zeile, in der sich der Cursor gerade befindet wird in einen Zwischenspeicher kopiert. Das ist analog der Zwischenablage bei DOS. Der Inhalt des Zwischenspeichers (Line Stack) wird informativ für den User in Bildschirmzelle 21 dargestellt. Stehen dort nur Fragezeichen, so ist momentan nichts im Zwischenspeicher enhalten (Ausgangszustand).

Gruppe Z / Zeichen AND
Die im Line Stack aktuell befindliche Zeile (Inhalt siehe Zeile 21) wird an die Stelle/Zeile kopiert, an der sich der Cursor gerade befindet. Der momentane Inhalt dieser Zeile wird gelöscht und ist verloren.

Gruppe S / Zeichen STOP
Das ist der Befehl, mit dem der Editor verlassen wird. Der angezeigte (und oftmals gegenüber der RAM-Disk veränderte Inhalt) wird in die RAM-Disk geschrieben. Damit wird er als im System gültig gekennzeichnet.

Gruppe S / Zeichen NOT
Der Editor wird verlassen ohne den angezeigten Screeninhalt in die RAM-Disk zu übernehmen. Aufgepasst: Die gemachten Änderungen sind damit verloren.

Gruppe S / Zeichen TO
Alle nach dem letzten Zurückschreiben auf die RAM-Disk gemachten Änderungen auf dem aktuell angezeigten Screen werden wieder rückgängig gemacht. Kann manchmal sehr sinnvoll sein, wenn man logische Fehler ausbessern will und zu einem geordneten (bekannten) Zustand des Screens zurückkehren will.

Gruppe S / Zeichen STEP
Der aktuelle Screen Inhalt wird in die RAM-Disk geschrieben und der vorherige wird zum Editieren freigegeben.

Gruppe S / Zeichen THEN
Der momentan aktuell angezeigte Screen wird in die RAM-Disk geschrieben und der nachfolgende wird zum Editieren freigegeben.

Gruppe S / Zeichen AT
Der angezeigte Screen wird auf den nächstfolgenden numerisch anschließenden kopiert. Dieser und die nächstfolgenden werden um jeweils einen Screen nach hinten versetzt. Der letzte Screen (richtig: das ist die Nummer 31) geht unweigerlich verloren. Solch weitreichenden Operationen sollte man deshalb immer gut überdenken, denn schnell ist mal ein Screen weg, den man vielleicht noch gebraucht hätte.

Gruppe S / Zeichen ERASE
Der angezeigte Screen wird auf dem Bildschirm und der RAM-Disk auf leer gelöscht. („EXTENDED MODE - SYMBOL SHIFT - 7“)
ACHTUNG!!! Diese Funktion wird sofort ausgeführt und kann nicht gestoppt werden.

Soviel also von meiner Seite zur Bedienung des Editors von Arcus-Forth. Wie ihr seht, ist die Bedienung an sich kein Hexenwerk. Man muss sich nur dran gewöhnen welche Tasten für die entsprechenden Aktionen benötigt werden. Ich hoffe, ich konnte den einen oder anderen auf diese durchaus interessante Programmiersprachenimplementierung aufmerksam machen. Theoretisch haben wir nun das Rüstzeug um mit Arcus Forth zu arbeiten. Jetzt brauchen wir

uns „nur" noch mit Forth an sich zu beschäftigen. Vielleicht hat ja der eine oder andere von euch schon Erfahrungen mit dieser Programmiersprache gemacht, die er auch uns anderen Usern mitteilen möchte? Es wäre auch interessant ob sich mal einer von uns an einen Einsteigerkurs für Forth ranmachen will. Eine mögliche Bedientechnik mittels Arcus Forth haben wir ja jetzt kennengelernt.

In diesem Sinne - happy computing

English summary:
Hello everyone,

it´s Forth time again and as I have told you last time in this article I will show you how to operate the Arcus Forth editor. As Forth is a very special language and not too easy to use, also the editor has some very strange functions that make it powerful but not too simple in use. The full screen editor is included in the dictionary of the Arcus Forth itself but you additional may need three so called TOP words. I will present and explain these words in the article followed by the detailed explanation of the editor own functions. So if this may have made you hungry for some more please be kind and write an article for the club magazine showing and explaining your experience using forth programming language.