

Wieso ein FOX anders ist

update: 19.11.2008

BauteilAuswahl

Ein FOX Microcomputer ist ein embedded computer. Eingebaut in einen Apparat soll er viele Jahre klaglos seinen Dienst versehen. Die Bauteile, aus denen ein FOX embedded computer besteht, sollen bewährte Industrieteile sein, die langfristig lieferbar sind. Durch den einfachen, robusten Aufbau kann das Design mit einer 4-lagigen Leiterplatte beherrscht werden, wodurch displayspezifische- oder kundenspezifische Anpassungen auch für Stückzahlen ab hundert Stück interessant werden.

Konzept

Im Gegensatz zum PC oder IPC soll der FOX ein Konzept haben, das sich nicht den vom Consumermarkt vorgegebenen Innovationszyklen angleichen muss. Das Diktat von immer leistungshungriger Software, die die Hardwareanforderungen immer höher schraubt, soll durchbrochen werden. Mit 16-bit breiten Datenbussen kann ein Optimum zwischen Leistung und Kosten erreicht werden.

Entwicklung

Im Vordergrund steht der Kunde. Er soll in der Lage sein, Änderungen selbst auszuprogrammieren, auch ohne ein Hochschulstudium in Informatik. Die zur Entwicklung erforderlichen Programmierwerkzeuge eigerStudio und eigerGraphicSuite können kostenlos von der Internetseite www.eigergraphics.com heruntergeladen werden.

Die grafische Oberfläche kann mit Photoshop, Paint, Word, Excel oder Powerpoint gestaltet werden.

Fotos können leicht in die Applikation eingebunden werden.





Schnittstellen zur Applikation

Die FOX embedded computer verfügen über serielle Schnittstellen zur Applikation. Ein Controllerboard steuert die Applikation. Es enthält alle erforderlichen I/O's und Speisungen zur Ansteuerung von Sensoren und Aktoren in der Maschine. Ganz einfache Applikationen können auch vom FOX direkt gesteuert werden. Wenn die Steuerung schon besteht, kann der FOX über eine serielle Schnittstelle mit der Steuerung kommunizieren.

Aufgaben des Displayrechners

Der Displayrechner stellt das HMI (human machine interface) her und ist das Bindeglied zwischen Echtzeitsteuerung und Bedienperson. Die Bedienperson erwartet eine flüssige Bedienung des Touchscreens ohne lästige Wartezeiten. Der Displayrechner hat folgende Aufgaben:

1. Darstellung eines Bildinhaltes: es kann sich um ein Foto handeln oder um einen aufgebauten (gerenderten) Bildschirm-Inhalt.
2. Refresh des Displays. Das Display muss 60 mal in der Sekunde neu geschrieben werden, damit ein flimmerfreies Bild entsteht.
3. Abfrage und Auswertung des Touchscreens für die Benutzerinteraktion.
4. Kommunikation mit der Applikation über die seriellen Schnittstellen.

Bandbreitenberechnung für VGA und WVGA-Displays

Das VGA-Display hat 640 Pixel in der Breite und 480 Pixel in der Höhe. Dadurch ergibt sich eine Pixelzahl von 307'200 Pixel. Wenn für die Darstellung eines Pixels 16bit (2 Bytes) benötigt werden, ergeben sich 614'400 Bytes pro Bild. Wenn das Display 60 mal pro Sekunde aufgefrischt werden soll, resultiert daraus eine Datenrate von 36'864'000 Bytes pro Sekunde oder 36.9 MB/s netto. In der Realität wird mit einem Pixelclock von 25 MHz gearbeitet, weil das Displaytiming noch unsichtbare Anteile im Timing enthält (horizontal- / vertical back- und frontporrch)

Das WVGA Display enthält horizontal 800 Pixel, entsprechend ist die Nettodatenrate 46'080'000 Bytes/s oder 46.1 MB/s. Der Pixelclock für das WVGA-Display beträgt 33 MHz.

Das Timing des Video-Refresh muss absolut genau erfolgen, sonst flackert das Display !





Unterschiede in der Hardware

In der Industrie existieren Mikrocomputer mit einer "unified memory architecture" (z. B. ARM-basierte Boards). Bei diesen Computern ist der Videospeicher, der den Bildschirm-Inhalt enthält, im gleichen Speicherbaustein abgelegt, wie das Programm und die Daten. Dadurch entsteht ein Flaschenhals beim Zugriff auf den Speicherbaustein. Entsprechend hoch muss die Taktrate sein, damit die Performance stimmt.

Der FOX hat demgegenüber ein VideoMemory, das 2 Ebenen hat und vom Prozessorspeicher entkoppelt ist. Der Mikrocontroller arbeitet ohne vom Displayrefresh beeinflusst zu werden. Der Grafikprozessor "EVE anna" beschleunigt Kopier- und Zeichenaufgaben. Durch die parallele Architektur mit 3 Datenbussen fließen die Daten beständig ohne Wartezyklen. Die schnellen SRAM als VideoRAM kennen keine Latenzzeit.

Unterschiede in der Software (-entwicklung)

Die Softwarearchitektur des FOX basiert auf einer virtuellen Maschine (eigerVM). Die virtuelle Maschine arbeitet den vom eigerStudio übersetzten Bytecode ab. Die Entwicklungsumgebung eigerStudio (IDE englisch für Integrated Development Environment) prüft die Syntax des Sourcecodes (Ihres Programms) und übersetzt es in ein EVI-File (Kompilierung). Die eigerVM interpretiert dann diesen ByteCode. Das Programm wird von der CompactFlash Card (CFC) in den Arbeitsspeicher geladen und dann ausgeführt. Verschiedene Programmierer können einzelne Seiten (Views) programmieren, die sich dann zu einem Gesamtprojekt zusammenfügen lassen (einfach alle auf die CFC kopieren). Die üblicherweise mühselige Integration der Software entfällt komplett.

Funktionsweise der Software am Beispiel einer einfachen Aufgabe

Im untenstehenden Beispiel soll ein gefülltes, rotes Rechteck an Position X=20 / Y=30 gezeichnet werden. Die Breite ist 200 und die Höhe 50 Pixel.

eigerStudio übersetzt die Programm-Anweisungen in einen ausführbaren Bytecode. Die virtuelle Maschine eVM holt sich den Registerladebefehl (vgl. Abbildung 1, roter Pfeil) und lädt die 5 Register der virtuellen Maschine, die für die Aufgabe vorgesehen sind. Der nächste Befehl (violetter Pfeil, violette Schrift) wird als "zeichne gefülltes Rechteck" interpretiert. Das Rechteck erscheint auf dem Bildschirm



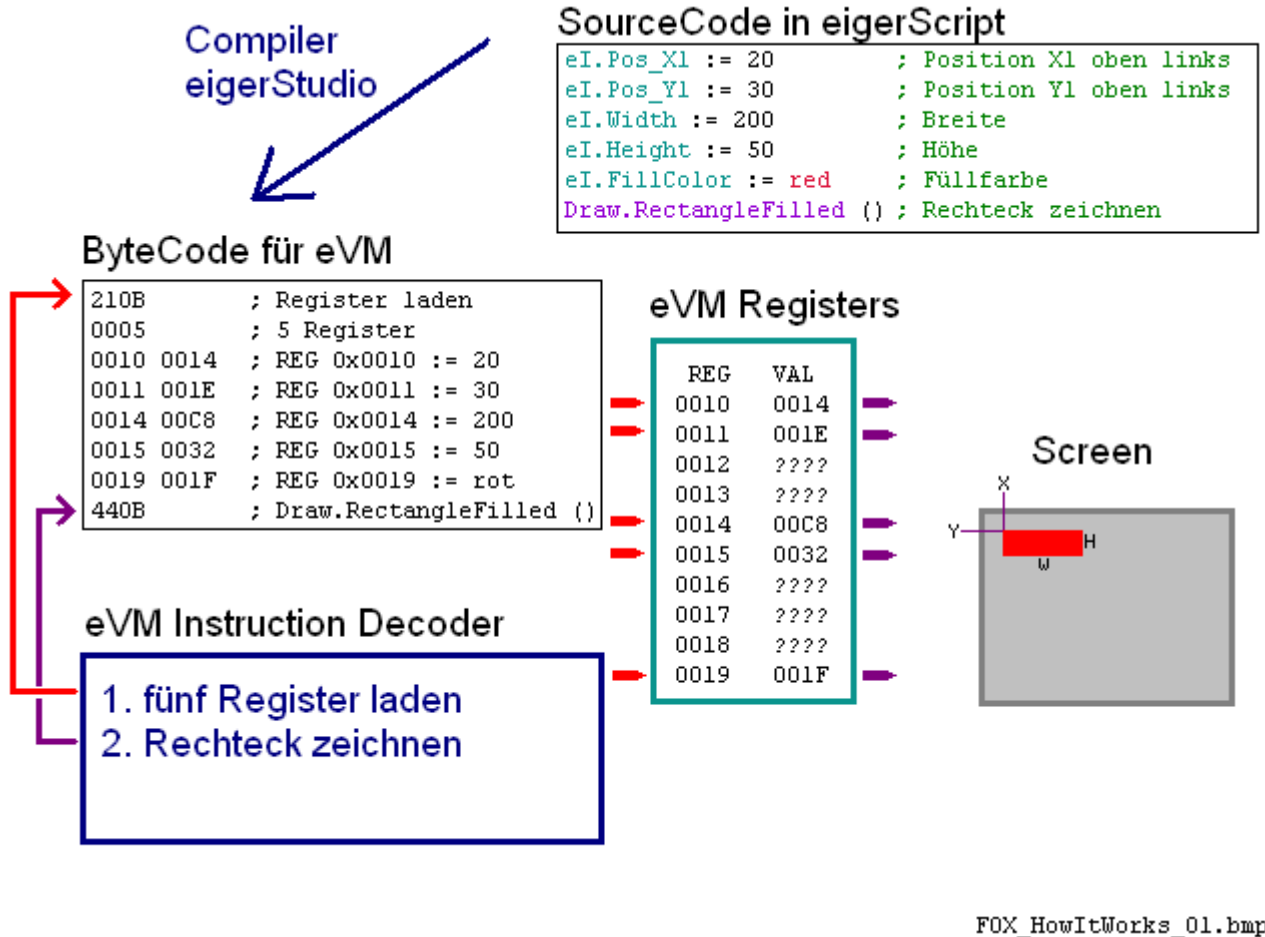


Abbildung 1: Beispiel der Funktionsweise des eigerSystems – wie der FOX embedded Computer auf dem Bildschirm ein rotes Rechteck zeichnet.

Bildschirmverwaltung

Der Bildschirm im FOX funktioniert wie ein Anschlagbrett; was zuletzt gezeichnet wurde, ist auf dem Bildschirm "zuoberst" sichtbar. Die Masseinheit, in der programmiert wird, ist Pixel. Die X-Koordinate zählt von links nach rechts, die Y-Koordinate von oben nach unten (vgl. Abbildung 1).

Es gibt zwei Bildschirmenebenen (vgl. Abbildung 2): Aus dem Refresh-VideoRAM RVR geschieht der Bildschirm-Refresh. Die Daten in diesem VideoRAM sind auf dem Bildschirm sichtbar.

Auf das accessible VideoRAM AVR kann die eigerVideo-Engine EVE ana lesend und schreibend zugreifen. Die Daten in diesem VideoRAM sind auf dem Bildschirm unsichtbar. Der





Grafikprozessor kann gleichzeitig in beide VideoRAM schreiben und so eine Kopie der Daten anfertigen. Bei einem Popup (z. B. Bildschirmtastatur oder PIN-Eingabe) kann das RVR unabhängig vom AVR beschrieben werden. Beim Abräumen (verschwinden lassen) des Popup wird der unveränderte Inhalt des AVR einfach wieder ins RVR kopiert und so das ursprüngliche Bild wiederhergestellt. Dazu stehen in eigerScript leistungsfähige Methoden der Klasse Display.XXX zur Verfügung.

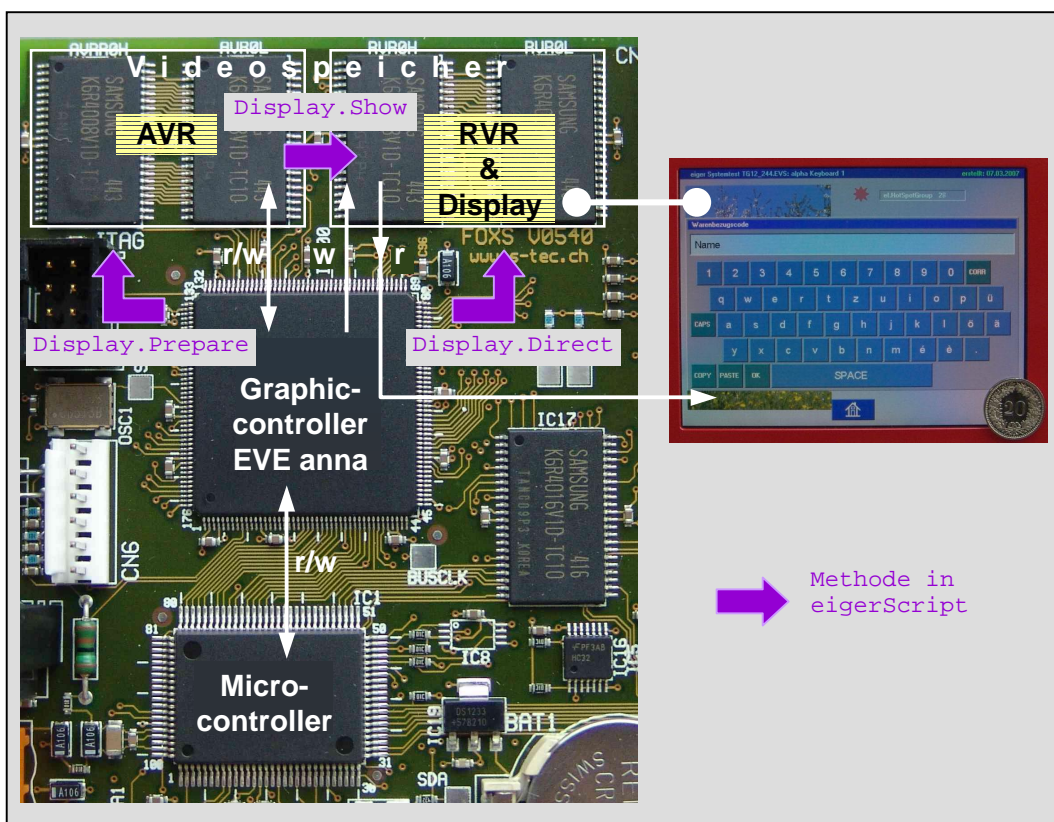


Abbildung 2: Speicherorte und Steuerung der Bildinformation auf der virtuellen Maschine des FOXS. Der Microcontroller (Prozessor) hat via EVE Lese- und Schreibzugriff (r/w) auf den Arbeitsspeicher. Die Information im RVR wird im Display dargestellt.



eigerScript: Event driven programming

Im Gegensatz zu den prozeduralen Sprachen ist mit eigerScript eine eventgesteuerte Programmierung implementiert. Auf Ereignisse im System (Timer, HotSpots etc.) werden kurze Unterprogramme (Handler) ausgeführt. Die Reaktionszeit ist äusserst kurz: ein Seitenwechsel ist innert 20ms vollzogen.

Im folgenden Codebeispiel sind drei Unterprogramme (Subroutinen) gezeigt, die beim Eintreffen der entsprechenden Events ausgeführt werden. Die auslösenden Events werden vom Touchscreen erzeugt, wenn ein HotSpot berührt wird.

```
STRING [64] HS1_Text.¢ = 'TopLine +1'
```

```
SUB HS1_LE
  Fill.LabelParameter ( KMCO_Button_UP_Style )
  Load.Pos_X1Y1 ( Btn_COLL, Btn_ROW1 )
  eI.Width := 120
  Label.Text ( HS1_Text.¢ )
ENDSUB
```

Button Leave Event

```
SUB HS1_DN
  Fill.LabelParameter ( KMCO_Button_DN_Style )
  Load.Pos_X1Y1 ( Btn_COLL, Btn_ROW1 )
  eI.Width := 120
  Label.Text ( HS1_Text.¢ )
ENDSUB
```

Button Down Event

```
SUB HS1_UP
  CallSubroutine ( HS1_LE )
  IF EF_TopLine.I < EF_MaxLines.I THEN

    EF_TopLine.I := EF_TopLine.I + 1
    Load.Pos_X1Y1 ( EF_Pos_X, EF_Pos_Y )
    CallSubroutine ( EditField_DrawLayout )

  ENDIF
ENDSUB
```

Button Up Event

```
; Hotspots installieren -----
CallSubroutine ( HS1_LE )
HotSpot.Install ( NIL, HS1_LE, HS1_DN, HS1_UP )
```

FOX_HowItWorks_02.bmp



Vor- und Nachteile der Lösung

Die FOX-Lösung hat folgende Vorteile:

- einfaches, robustes Design ohne höchste Taktfrequenzen; dadurch low EMI (elektromagnetische Interferenzen).
- einfaches Printplatten-Layout erlaubt kostengünstige Anpassung an verschiedene Displays.
- hohe Parallelverarbeitung; dadurch hoher Durchsatz bei moderaten Taktraten.
- kurze Reaktionszeiten, auch bei Projekten mit vielen Views.
- lange Verfügbarkeit der Industriekomponenten.
- Grafikmaschine als IP (intellectual property) in programmierbarem Chip; dadurch keine Abkündigung der Grafiklösung absehbar.
- geringer Stromverbrauch dadurch geringe Erwärmung.
- keine beweglichen Teile ⇒ kein Verschleiss.
- kostenlose Entwicklungsumgebung für den PC.
- dank der virtuellen Maschine eigerVM ist keine Software-Integration nötig.
- die virtuelle Maschine lässt sich auf andere Hardware übertragen.
- die Gesamtlösung ist kostengünstig durch optimales Zusammenspiel von Software und Hardware.
- die Software kann einfach kopiert werden (einfach CFC kopieren).
- die Software kann als ZIP-File per email versandt werden.
- die Software kann einfach getauscht werden (einfach CFC wechseln).
- äusserst kompakter ByteCode (kleine Programme sind schnell geladen).

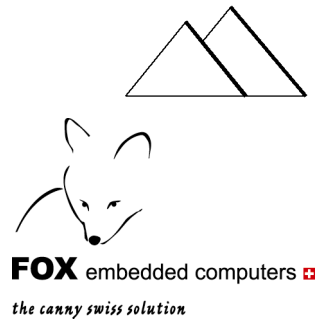
Die FOX-Lösung hat folgende Nachteile:

- mehrere Chips als Video-RAM (teuer).
- teurer Grafikchip.



S-TEC electronics AG

industrial electronics



FOX embedded computer

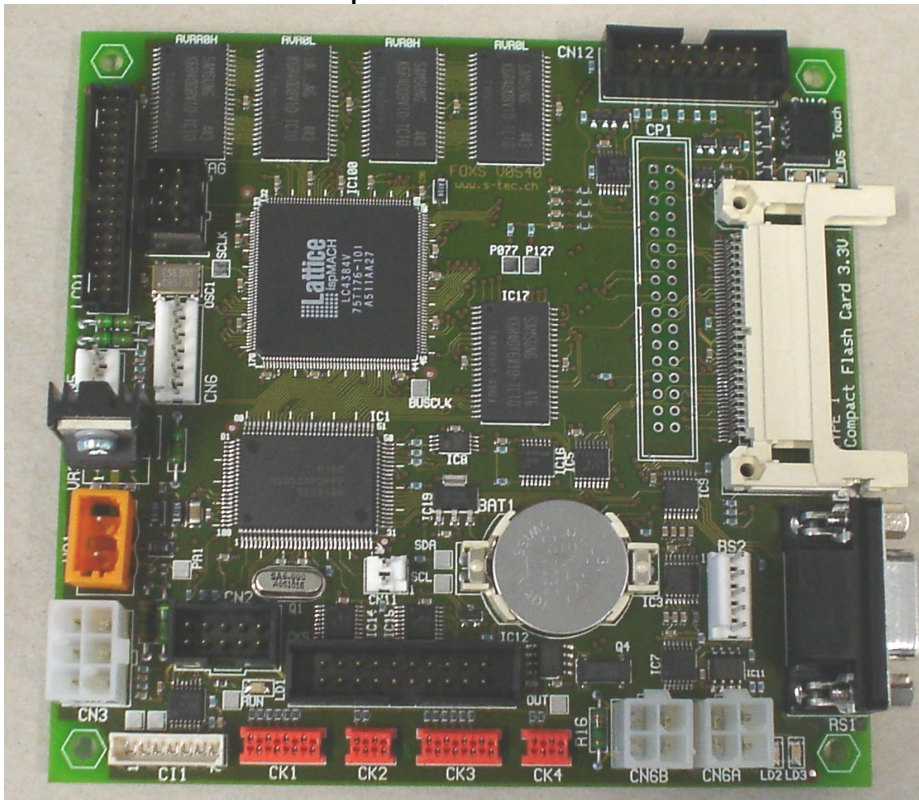


Abbildung 3: Platine des FOXS.

