Long time Oscilloskop with Atmega32 & LCD-Display 128x64 Pixel

Einleitung:

Das preiswerte Display LCD DM19264A (Pollin) (Siehe auch http://www.mikrocontroller.net/topic/217600) führte zu meinem bisher größten Projekt mit ATmega-Microcontrollern. Eine umfassende Beschreibung siehe im folgenden oder in den verschiedenen Anhängen. Allen, die sich an einen Nachbau wagen, wünsche ich viel Spass und neue Erkenntnisse. Obwohl der Sourcecode in C sicher keinen Schönheitspreis gewinnen kann, habe ich ihn angefügt (Arbeitsverzeichnis von ATMEL AVR Studio 4.18 komplett als *.zip) . Zur Weiterentwicklung sind alle aufgerufen. Ich möchte mich nicht an der Weiterentwicklung beteiligen, stehe aber für Fragen nach bestem Wissen zur Verfügung.

Der Anfang zu meinem Oszilloskop mit LCD-Display war der Artikel http://www.serasidis.gr/circuits/AVR_oscilloscope/avr_oscilloscope.htm
Dazu entstand eine Platine mit zusätzlichem per Relais im Bereich x1 bis x100 einstellbaren Vorverstärker. Beim Aufspielen der Software-Version 1.01 und Verstehen des in der Programmierspache C geschriebenen Programms zeigte sich dann bald, dass alles ohne Verwendung eines Timers nur mit Verzögerungsschleifen im Hauptprogramm ablief. Es gab nur eine Timing-Frequenz und die Erweiterungsmöglichkeiten waren auch stark eingeschränkt.

Der neue Ansatz, diesmal mit der im Internet verfügbaren universellen Bibliothek (für LCD's mit der Auflösung 128x64 oder 192x64 Pixel) nach http://en.radzio.dxp.pl/ks0108/ begann mit einer kompletten universellen Grafikansteuerung als Grundlage und wurde von mir um eine Vielzahl von Scope Funktionen ergänzt. Grundlage für das Timing ist ein IRQ alle 10us bei 16 MHz Quarz. Über Zähler davon abgeleitet wird die variable Timing-Einstellung von 10us/Skalenteil (Skt) bis 200min/Skt (ergibt ~ 33 Stunden). Ebenso abgeleitet werden die Display-Zeit (0,5sec) und das Delay für die Tastaturentprellung von 2,5msec nach http://jump.to/fleury. In der „nicht-IRQ-Zeit" im main-Programm laufen die restlichen Programme wie z.B. LCD-Anzeige und Tastenauswertungen.

Beschreibung:

Auf einer Europaplatine (160x100mm) finden Platz:

- LCD-Display 128x64 Pixel mit Speicher und Hintergrundbeleuchtung
- Tastenfeld mit 6 Tasten (oder extern)
- Digitalteil mit ATmega 32-16PU (16 MHz) (oder ATmega 16)
- Analogteil mit per Relais einstellbarer Verstärkung x1 bis x100-fach
- Gleichrichter und Spannungsregler für +5Volt und –5Volt
- ISP Schnittstelle zum Programmieren per PC
-

Messbereiche (bis auf Offset-Poti und Vorteiler alle Umschaltungen per Software):

- Eingangsspannung: 5, 10, 25, 50, 100, 250, 500mV(ss) (per Skt = 10 Pixel)
- Frequenzgang: 0 – 500KHz bei x1 und 0 – 200 KHz bei x100
- Eingang umschaltbar DC oder AC (über Kondensator 100nF)
- Frequenzkompensierter wählbarer Vorteiler /10
- Per Potentiometer einstellbarer Display-Offset
- Timing (per Skt = 10 Pixel): 100, 200, 500usec, 1, 2, 5, 10, 20, 50, 100, 200, 500 msec, 1, 2, 5, 10, 20, 50, 100, 200 sec, 10, 20, 50, 100, 200 min

- Trigger: einstellbar in 64 Stufen über den vollen Anzeigebereich
- + Trigger und - Trigger wählbar und abschaltbar (für Signalerkennung)
- „Single shot“-Modus per Taste (sofort ohne Trigger)
- Anzeige mit zwei Betriebsarten (bereichsabhängig):
  - 100us bis 20 msec: komplette Messung => komplette Anzeige
  - 50 msec bis 200 min: jeder gemessene Punkt wird sofort angezeigt
- Alle Einstell-Werte werden am rechten Bildrand kontinuierlich angezeigt
-

Hardware:

Schaltungsbeschreibung:

Netzteil: Ein Steckernetzteil mit 2 Wicklungen a 8,3V/0,26A versorgt über 4 Dioden und zwei Elkos den positiven und negativer Spannungsregler und liefert +5V und –5V für den 8-bit AVR-Microcontroller, das Display, das Referenzelement TL431, die Operationsverstärker 2x TL071 / 1x TL072 sowie die 4 Relais.
Vorverstärker: Über einen 3-poligen Stecker gelangt das Eingangssignal direkt an den 3-stufigen in der Verstärkung umschaltbaren Vorverstärker. Der zweite Eingang teilt das Signal zuerst frequenzkompensiert auf 1/10. Über einen Schutzwiderstand von 10 kOhm und zwei Schottky-Dioden (aussuchen auf geringen Reststrom!) wird der erste Operationsverstärker vor Überlast geschützt. Per Relais lassen sich die 3 Stufen einzeln umschalten: Verstärkung x1 oder je nach Stufe x10, x5 oder x2. Durch die maximale Verstärkung von 10 pro Stufe und die niederohmigen Spannungsteiler ergibt sich so ohne weitere Maßnahmen eine flacher Frequenzgang von DC bis weit über die Messgrenze. Über je einen 100 Ohm Widerstand (zum Stromsparen) und je 2 Schottky-Dioden (Schutz gegen Spannungsspitzen) werden die Relais vom Microcontroller angesteuert.
Referenzspannung: Das Display zeigt auf einem Bereich von (6x10 Pixel + 4 Pixel Rand) den gemessenen Spannungswert an. Bei 500 mV (Verstärkung 1x) liegt am Eingang des ADC (PA0 des Microcontrollers) eine Spannung von 6,4x 500mV = 3,2V an. Diesen Wert liefern die Operationsverstärker bei Versorgung mit +/-5Volt Versorgung problemlos.
Tastatur: Über 6 Einzeltaster lassen sich die Funktionen steuern. Über die internen „pull-up“-Widerstände des Microcontrollers liegen die Tasten auf +5V und werden bei Betätigung gegen Masse kurzgeschlossen. Auf dem Board ist Platz für einfache Taster, wer ein besseres Tastenfeeling möchte, kann über den 9-poligen Stecker externe Tasten anschließen.
Programmierung: Der 10-polige Stecker mit üblicher Beschaltung für AVR-Programmer ist so beschaltet, dass üblicherweise keine Probleme durch die Doppelbelegung von PB5 entstehen. Beim Programmieren ist PB3 hochohmig und über den 4,7kOhm Widerstand an RST wird das Display abgeschaltet und seine Anschlüsse sind dann hochohmig.

Leiterplatte:

Der Prototyp entstand mit der empfehlenswerten Freeware KiCad. Nach Erstellung der Schaltung und eindeutiger Nummerierung aller Komponenten werden die Layouts den einzelnen Komponenten zugeordnet. Das Ganze wird dann auf die Leiterplatte gebracht und verteilt. Die Verbindungen werden als Gummibänder eingeblendet, die Leiterbahnen werden dann manuell gezogen. Damit führt eine fehlerfreie Schaltung auch zu einer „fehlerfreien“ Leiterplatte. Die eigentliche Arbeit steckt im folgenden:
Alle Komponenten sind zugänglich, möglichst kurze Leiterbahnen und wenig Drahtbrücken bei einseitigem Layout, optimale Masseverbindungen, keine

Schwingungen oder kein Rauschen auf der Masse, kluge Verteilung von Analog- und Digital-Komponenten. Je kompakter das ganze sein soll, desto mehr Aufwand ist nötig.
Ausdruck der Leiterbahnen mit möglichst viel Toner und kopieren mit der Toner-Transfertechnik auf die gut gereinigte Leiterplatte. Ätzen und Bohren wie üblich.
Da Laserdrucker keine großen Flächen deckend drucken können, wird der Rand außen per wasserfestem Filzschreiber nachträglich zur großen Massefläche.

Software:

Mit der kostenlosen Software AVR Studio (hier noch V 4.16, aktuell V4.18) lassen sich sehr komfortabel auch größere Programme in der Programmiersprache C erstellen.
Ein größeres Programm besteht aus mehren *.c (Programm-File) und *.h (Header-File) sowie den erforderlichen Bibliotheken. Mit dem Befehl „build“ wird daraus die zum Speichern auf dem Microcontrollers erforderliche *.hex erzeugt.
Wichtig für ein erfolgreiches Vorgehen ist das Ändern in überschaubaren Schritten, beginnend bei einer funktionierenden Version und Speichern der Zwischenschritte!

Die Grundlage für das inzwischen recht umfangreich Programm ist eine frei verfügbare Bibliothek zur Ansteuerung von Matrix-LCD-Displays mit dem Controller KS108. Der Controller kann 64x64 Pixel ansteuern und hat ein eigenes RAM, das beschrieben und ausgelesen werden kann. Üblicherweise werden zwei Controller verwendet und ergeben so 128x64 Pixel, bei 3 Controllern gibt es 192x64 Pixel.
Ergänzt man die Bibliothek um ein Hauptprogramm (main.c) kann man folgende Befehle verwenden: Löschen des Displays, Ausgabe von Text im Raster 5x7, setzen oder (neu) toggeln von einzelnen Pixeln, zeichnen von Geraden, Rechtecken oder Kreisen, Ausgabe einer beliebigen rechteckigen Pixel-Grafik.

Konzept:

Das verwendete Konzept für ein möglichst universelles Scope besteht aus einem Timer-IRQ alle 10us für zeitkritische Aktionen und einem Hauptprogramm, das in der freien Zeit die aufgelaufenen Tätigkeiten abarbeitet. Für die IRQ-Routine stehen bei 16 MHz Quarz maximal 160 Takte (<160 Assembler-Befehle) zur Verfügung, in Wirklichkeit weniger, da auch das Hauptprogramm abgearbeitet werden muss.

IRQ-Routine:

1. Auslesen des ADC-Wandlers, der in einem der vorhergehenden IRQ gestartet worden war,aber nur dann, wenn die im Scope-Timing eingestelle Zahl an IRQ’s abgelaufen ist.
2. Starten des ADC nur dann, wenn ein neuer Timing-Zyklus startet
   Bei 2 MHz ADC-Clock Frequenz dauert ein Umsetzung ca 7usec
   und kann im nächsten IRQ ausgelesen werden. (Da nur 8 bzw. 6 Bit Auflösung angezeigt werden, verursacht die Ungenauigkeit durch die hohe Clock-Frequenz kein Problem.)
3. Auslesen der Tastenbetätigungen alle 2,5msec und entprellen, dann bereithalten für das Hauptprogramm, das die Tastenbetätigungen umsetzt

Hauptprogramm:

1. Initialisierung des Microcontrollers, des LCD-Displays und der Hardware
2. Aktualisierung des Displays: Messwerte und Textausgaben
3. Ausführung von Aktionen entsprechend den Tasteneingaben

Bei der Umsetzung ist folgendes zu beachten:

IRQ-Routine und Hauptprogramm sind zeitlich unabhängig. Parameter müssen über globale Variable übergeben werden. Die Arbeitsaufteilung muss über Flags gesteuert werden, damit die Abfolge stimmt. Wo möglich sollen lokale Variable verwendet werden, da diese von Compiler bevorzugt mit Registern umgesetzt werden und damit schneller abgearbeitet werden können.

Bedienung:

Nach dem Einschalten startet das Scope im Modus „Auto“ mit dem Eingang als AC, 500mV, 100usec, Trigger ein, Triggerlevel = 32 (50%) und + Trigger,

Auto:
Die Taste „Toggle“ wechselt zwischen AC und DC
Die Tasten „Up“ und „Down“ wählen den Spannungsbereich,
Die Tasten „left“ und „right“ wählen den Timing-Bereich
Mit der Taste „Modus“ wechselt man in den Modus „Cont“

Cont:
Keine Trigger-Schwelle aktiviert – alles wird kontinuierlich angezeigt
mit der Taste „Modus“ wechselt man in den Modus „Sing“

Sing:
Mit der Taste Toggle wird ein einzelner Durchlauf gestartet und angezeigt
Mit der Taste „Modus“ wechselt man in den Modus „Trig“

Trig:
Anzeige Triggerschwelle und Trigger-Polarität
Mit der Taste Toggle wird zwischen + Trigger und - Trigger gewechselt
Die Tasten „Up“ und „Down“ wählen den Trigger-Level
Mit der Taste „Modus“ wechselt man wieder in den Modus „Auto“

Mit dem Potentiometer stellt man den Anzeige-Offset ein.

Information für einen Testaufbau/Nachbau:

Der Anfang der Software-Entwicklung erfolgte auf dem Pollin Evaluationsboad Ver.2.01 mit einem ATmega 32-16PU und 6 Tasten, (ohne Eingangsverstärker) mit einem 192x64 Display. AREF ist auf dem Board auch an den Stecker geführt. Das dann mit dem kompletten Board verwendete Display im Schaltplan LCD TG12864B-13 ist von Pollin.
Der als PDF beigefügte Schaltplan und das dazugehörende Layout wurden mit den Erkenntnissen vom Prototyp (siehe Foto) verbessert.
Wenn sie sich nach meinem PDF-Layout eine eigene Platine fertigen wollen, vergessen sie nicht die zusätzliche Masse mit dem wasserfesten Filzschreiber am Leiterplattenrand! Die 4 Relais mit dem speziellem Footprint stammen aus meiner Bastelkiste. Die restlichen Bauteile (Ausnahme Steckernetzteil von Pollin) sind nach meiner Meinung handelsüblich.

4.3.2012

Karl-Anton