V1.1-2013 – Variante LM1881

Video Inserter
mit „Soft-Genlock“

Inhaltsverzeichnis

Hinweis zur Laborübung

Bitte lesen Sie vor der Übung folgende Dokumente sorgfältig durch:

- Diese Angabe – das Dokument, dass Sie gerade lesen, insbesondere die Abschnitte „**Aufbau analoger CCIR/PAL Videosignale**“ und „**Ein zeichenbasierter Videocontroller in Software**“
- Den Abschnitt über die synchrone serielle Schnittstelle (SSC) im XC164-Peripherie-Handbuch
- Den Abschnitt über „Fast external Interrupts“ im XC164-Peripherie-Handbuch
- Den Abschnitt über den Analog-Digital-Wandler (ADC) im XC164-Peripherie-Handbuch

Nützen Sie die genannten Unterlagen für die Laborvorbereitung! Die Wahrscheinlichkeit, dass Ihnen beim Eingangstest Fragen daraus gestellt werden ist hoch.

1. Einführung

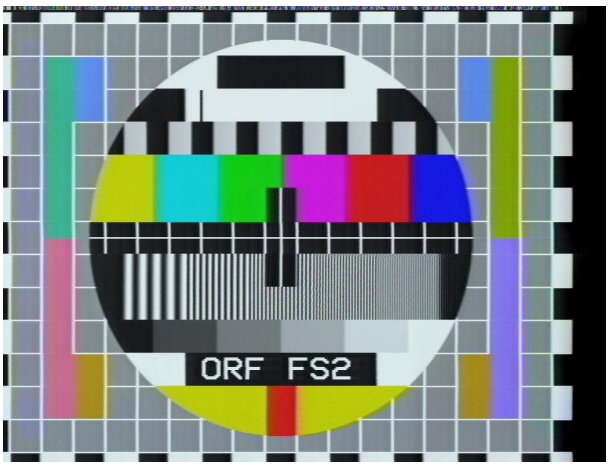
1.1 Vorwort

Unter Videosignalen versteht man zumeist analoge oder digitale Signale, die Bildinformationen seriell übertragen. In diesem Laborversuch werden nur analoge Videosignale nach der CCIR/PAL Norm genutzt, da sie einfach zu verstehen, beinahe universell kompatibel sind und der technische Aufwand um Größenordnungen kleiner ist als bei digitalen Videosignalen wie DVI/HDMI/SDI.

Schwarz-weißes analoges elektronisches Fernsehen und analoge Videosignale existieren seit Mitte der 20er oder Anfang der 30er Jahre des vergangenen Jahrhunderts. Zur elektronischen Übertragung musste die (2D-)-Realität zunächst optisch und dann elektronisch abgebildet werden. Das Bild wird dazu punkt- bzw. zeilenweise abgerastert und seriell (analog) übertragen, da eine gleichzeitige Übertragung eine sehr hohe (Funk-)-Bandbreite benötigen würde, das menschliche Auge träge genug ist um den Bildwechsel nicht zu bemerken und zudem das Rasterverfahren ideal der damals verfügbaren Technik, einem abtastenden Elektronenstrahl in der Kamera-Röhre und einem zeichnenden Elektronenstrahl in der Bildröhre angepasst ist.

Aus einem Kompromiss zwischen räumlicher und zeitlicher Auflösung und Bandbreite resultierte das Zeilensprung- oder Halbbild-Verfahren („Interlacing“), wobei zweiräumlich ineinander „verzahnte“ Halbbilder – das gerade und ungerade Halbbild - zeitlich nacheinander übertragen werden. Diese Tatsache werden wir außen vor lassen, da sie unseren Versuch nicht beeinträchtigt.

Es ist oft notwendig oder wünschenswert, Text oder Grafik wie beispielsweise Datum- und Uhrzeit sowie Ort in Überwachungsanwendungen, den Spielstand bei einem Fußballspiel, Marker-Linien sowie Messwerte in Industrieanwendungen, oder ein Senderlogo in Echtzeit (Live) in ein Videobild einzublenden. Beim Fernsehen geschah dies bis zu den 70er Jahren mithilfe von aufwändigen mechanischen Aufbauten und Tafeln, die mit synchron laufenden Kameras abgefilmt und mittels Helligkeits-Tastung (Lumakey) ins Live-Bild eingeblendet wurden - ganz ohne Digitaltechnik.



Die ersten voll-elektronischen digitalen Schriftgeneratoren, „Character Generatoren“ oder Video-Inserter genannt wurden Mitte der 70er Jahre verfügbar und hatten eine klötzchenartige niedrig aufgelöste Schrift, da Halbleiter-Speicher teuer, klein und langsam war – weswegen man sie, außer bei Sportereignissen nicht oft „on-air“ sah. Heimcomputer der 80er Jahre wie der Apple II, TRS-80 oder Commodore VIC-20 / 64 erzielten eine ähnliche Schriftqualität.

Der CCIR-Standard, der bis heute verwendet wird wurde Ende der 40er Jahre fixiert und Mitte der 60er Jahre um abwärtskompatible Farbübertragung zu PAL ergänzt. PAL war bis vor kurzem (bis zur Einführung von DVB-T) auch das Basisband-Signal des Fernsehens. Trotz dessen und der Einführung von digitalen HDMI-Signalen im Consumer-Bereich und SDI-Signalen im Studio-Bereich beherrscht nahezu jedes Gerät die Ausgabe und Wiedergabe von PAL-Composite-Signalen und auch im Bereich der Videoüberwachung werden weiterhin auf absehbare Zeit noch PAL-Composite-Signale verwendet, ebenso wie im Amateurfunk und bei Modellbauern, welche ihre Flugmodelle mit Funkkameras ausstatten.

Da die Farbinformation in PAL-Composite-Signalen auf das herkömmliche Helligkeitssignal quadratur-amplituden-moduliert ist, müssten wir, um auch Einblendungen in Farbe zu ermöglichen das PAL-Signal in seine RGB-Komponenten zerlegen (dekodieren) und nach der Bearbeitung wieder zu einem PAL -Signal zusammensetzen (enkodieren) was technisch sehr aufwändig wäre. Deswegen wird unser Projekt nur die Helligkeitsinformationen bearbeiten, die uns direkt zugänglich sind. Somit realisieren wir weiße Einblendungen auf einem farbigen Bild.

Was Sie von dieser Laborübung mitnehmen:

- Verständnis für Echtzeit-Systeme und Echtzeit-Signalverarbeitung (ohne DSP-Know-How)
- Eine der einfachsten Formen einer „Grafikkarte“ implementiert zu haben

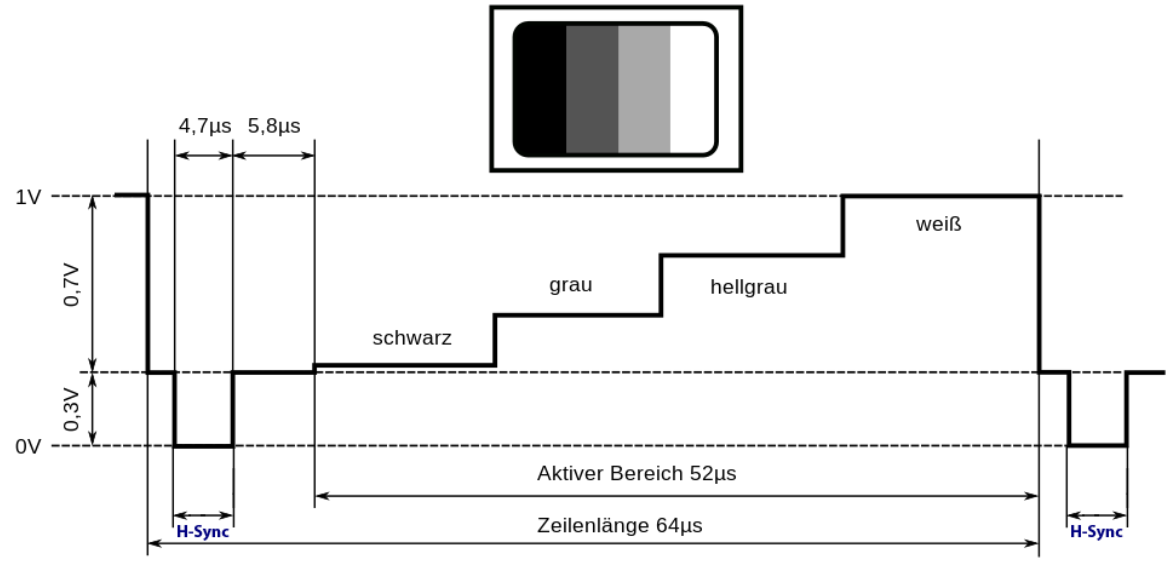
Die hier behandelten Konzepte finden Sie in abgewandelter Form auch bei digitalen Video- und Grafikanwendungen wieder. Das Verständnis für Zeichensätze und Grafikpuffer ist auch bei der Verwendung von LCD/TFT-Displays in Embedded-Anwendungen nützlich.

1.2 Aufbau analoger CCIR/PAL Videosignale

PAL/CCIR Signale sind analoge Spannungs-Signale mit einer Signalamplitude von meistens 1Vpp (Volt Peak-Peak), welche jedoch im Gegensatz zu Audiosignalen nicht gänzlich zeitkontinuierlich sind.

Das **Videobild ist in Zeilen unterteilt**, welche durch einen **horizontalen Synchronisations-Puls (H-Sync)**getrennt sind. Der zeilenweise Bildaufbau fängt links oben an und endet rechts unten.

Die Zeilen selbst sind kontinuierlich – man könnte sagen, es gibt keine exakten „Pixel“ in einer Zeile in diesem Sinne. Die Helligkeit an einem bestimmten Punkt in einer Zeile wird durch den momentanen Spannungswert zu dem Zeitpunkt bestimmt – je höher die Spannung, desto heller, je niedriger die Spannung, desto dünkler ist es. Jede Zeile dauert 64µs, von denen 52µs Bildinformationen tragen können.

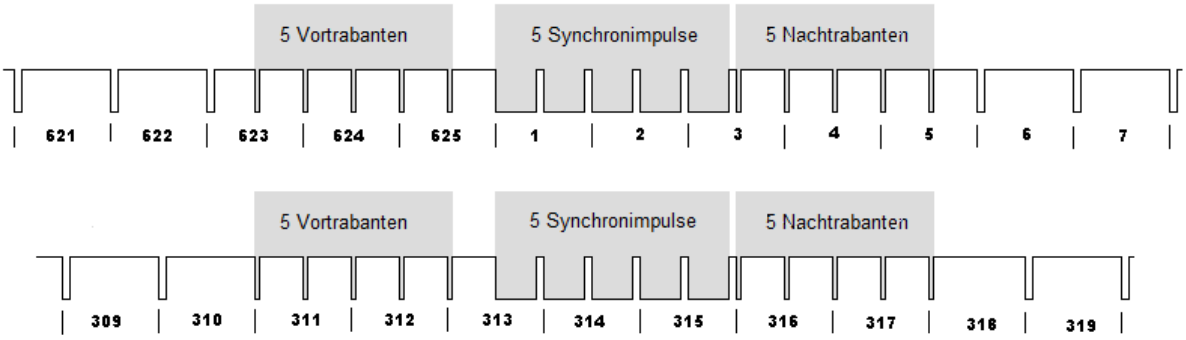


Die Bereiche vor und nach dem H-Sync, die nicht zum aktiven Bereich gehören nennt man vordere und hintere Schwarzschulter und deren Existenz hat historische Gründe, außerdem dient die hintere Schwarzschulter als eine Art Referenz für den Schwarzpegel – wir werden uns darum nicht weiter kümmern, da wir ja keine normgerechten Signale erzeugen werden, sondern uns nur darauf synchronisieren sollen.

CCIR/PAL arbeitet mit dem Zeilensprung-Verfahren („Interlacing“), wodurch 2 Halbbilder - ein gerades und ein ungerades - zeitlich nacheinander übertragen werden, die beim Empfänger räumlich ineinander „verzahnt“ sind. Es werden pro Sekunde 25 Vollbilder oder 50 Halbbilder übertragen. Die Tatsache, dass zwei Halbbilder ein Vollbild ergeben und das eines davon eine halbe Zeile länger ist können wir vernachlässigen – denn wir werden die **50 Halbbilder pro Sekunde** fortan nur einfach als 50 Bilder (mit der halben Auflösung eines Vollbildes)ansehen.

Jedes Halbbild besteht aus 312,5 Zeilen, von welchen 287,5 Zeilen (wir nehmen an 288 Zeilen) Bildinformationen tragen – die restlichen sind schwarz – auch das hat historische Gründe. Von diesen **288 Bildzeilen** sind – je nach Fernseher/Monitor – nicht alle sichtbar, denn oben und unten werden einige Zeilen abgeschnitten. Auch die ganze Zeile ist nicht immer sichtbar, weswegen man Text nicht unbedingt bis zu den Rändern laufen lassen sollte.

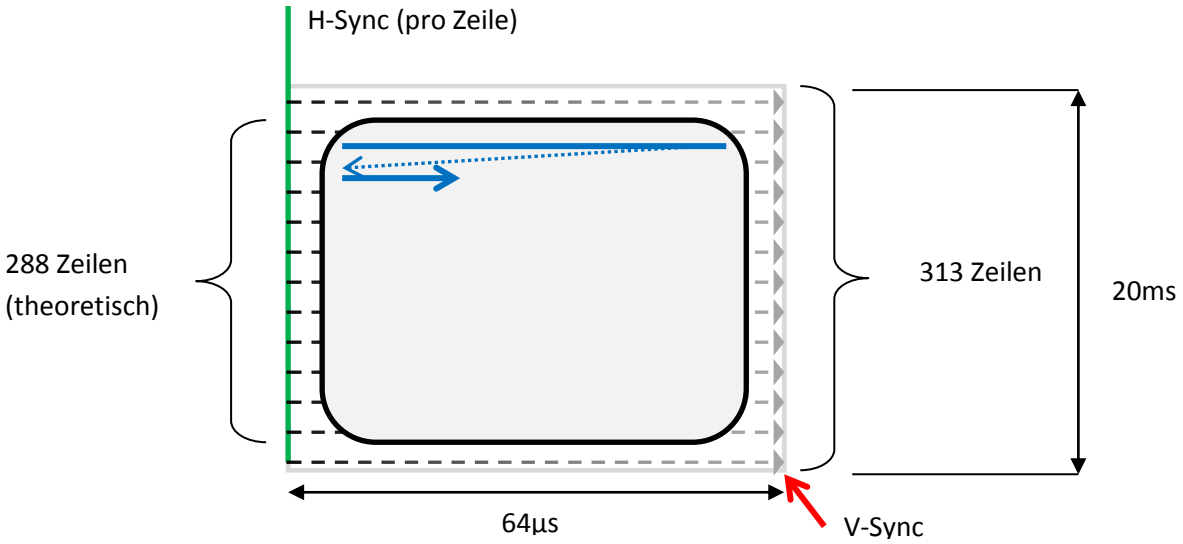
An ein Halbbild schließen sich 5 sog. Vortrabanten (kurze Sync-Pulse, 2.35µs), die wir vernachlässigen können, **5 VSync-Pulse (lange Sync-Pulse, 27,3µs – dauern also fast eine halb Zeile lang!)** und dann wieder 5 Nachtrabanten (kurze Sync-Pulse, 2.35µs, exakt wie die Vortrabanten), die wir ebenfalls vernachlässigen an. Mit diesen **vertikalen Synchronisations-Pulsen** beginnt ein neues Halbbild.



Fassen wir das wichtigste zusammen, woraus 1 Halbbild besteht:
-> 50 Halbbilder pro Sekunde
->ca. 313 Zeilen, von denen jede 64µs lang dauert
-> jede Zeile beginnt mit einem H-Sync Puls zu 4.7µs
-> jedes Halbbild beginnt mit 5 V-Sync Pulsen zu 27.3µs
(und vernachlässigbaren Trabanten)

-> 288 Zeilen mit Bildinhalten
-> 52µs aktiver Bereich in jeder Zeile

So kann man sich vereinfacht ein Halbbild vorstellen:



Die µs-Zeiten sind nicht immer exakt, manches Equipment hält sich nicht exakt an den Standard, was jedoch kaum ein Problem darstellt, solange dieser nicht grob verletzt wird.

Die QAM-Farbmodulation, welche aus einem schwarzweißen CCIR-Signal e ein farbiges PAL-Signal wird, bleibt wie zu Anfang angekündigt durch uns unberücksichtigt.

2. Allgemeines

Ziel dieser Aufgabe ist es, den Umgang mit externer Peripherie und die Programmierung eines zeitkritischen Echtzeitsystems zu erlernen. Es gibt einen Aufbau, der aus einer Video-Signalquelle, dem Video-Inserter-Aufsteck-Board und einem Video-Monitor besteht.

Es soll ein Mikrocontroller-Programm entwickelt werden, dass Text-Informationen in ein Videosignal einblendet

2.1 Laboraufbau

Der Laboraufbau setzt sich aus den folgenden Komponenten zusammen:

- Mikrocontroller-Board mit XC164 Mikrocontroller (XC164CM Easy Kit)
- Aufsteck- Platine mit Video-Inserter Schaltung, Temperatur-Sensoren und Chinch-Buchsen
- Video-Signalquelle (Überwachungskamera)
- Video-Monitor (TFT-Monitor, eventuell mit VGA-Konverter)

2.2 Inbetriebnahme

Lesen Sie die Slides “Entwicklung mit dem XC164” (**xc164_einfuehrung_handson.pdf**) und befolgen Sie die Anweisungen darin, um ein Projekt anzulegen und den Aufbau zum ersten Mal in Betrieb zu nehmen.

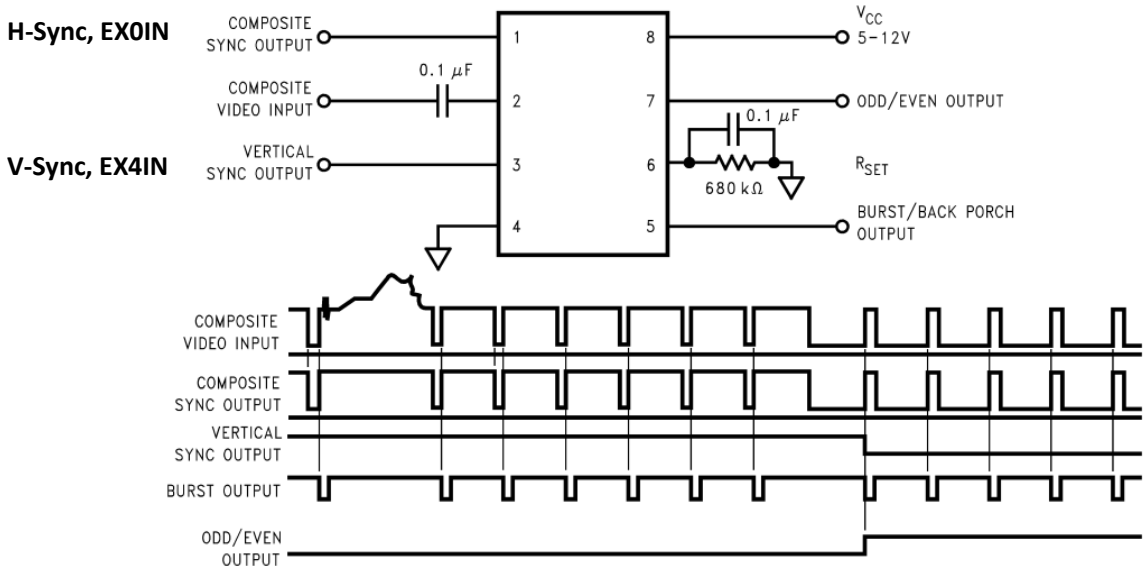
Überprüfen Sie, ob Sie das Kamera-Bild einwandfrei auf dem Video-Monitor sehen – es wird durch die Schaltung durchgeschliffen.

3. Hardware

Die – aus Sicht des XC164 - externe Hardware besteht aus der Video-Inserter Addon-Platine, welche einen Sync-Separator-Baustein **LM1881**, eine **Einblendeschaltung** und einen Temperatur-Sensor **TMP36** enthält.

3.1 Sync-Separator NSC/TI LM1881

Der Sync-Separator LM1881, der an den Pins mit den Funktionen EX0IN und EX4IN angeschlossen ist dient zur Abtrennung der horizontalen und vertikalen Timing-Informationen aus dem Eingangs-Videosignal. Diese Pins sind zur Nutzung mit „Fast External Interrupts“ vorgesehen.



Weitere Informationen können Sie dem Datenblatt des LM1881 entnehmen.

Wieviele V-Sync Impulse gibt der LM1881 pro Bild aus?

3.2 Video-Einblendeschaltung

Die Video-Einblendeschaltung ist an der Daten-Ausgabeleitung der **SSC1** angeschlossen und besteht aus einer Diode und einem Potentiometer als Spannungsteiler. Die Takt-Leitung und die Daten-Eingabeleitung werden nicht genutzt. Ein Logisch-1-Pegel erzeugt ein weißes Pixel, ein Logisch-0-Pegel erzeugt aufgrund der Diode kein bzw. ein transparentes Pixel.

Beachten Sie bitte, dass die SSC-Einheit des XC164 im Idle-Zustand einen Logisch-1 Pegel auf der Datenleitung erzeugt.

3.3 Temperatur-Sensor TMP36

Der **analoge** Temperatur-Sensor ist an den Pin mit der Funktion AN0 angeschlossen und gibt eine Spannung aus, welche der aktuellen Temperatur **direkt proportional** ist.

Die Kennlinie und genaue Funktion des Sensors entnehmen Sie bitte dem Datenblatt.

4. Software

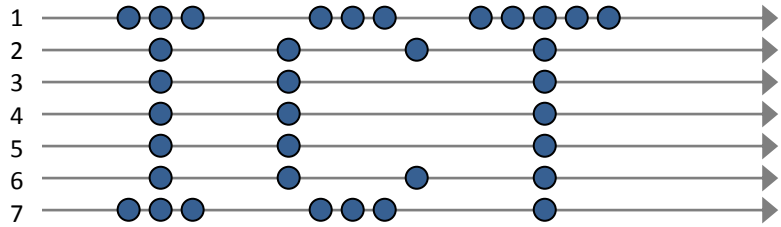
4.1 Ein zeichenbasierter Videocontroller in Software

Ein Videocontroller (auch bekannt als „CRTC“ oder „VDC“) ist eine zumeist integrierte, in der Anfangszeit auch diskrete Logik-Schaltung, deren Aufgabe die Erzeugung von Videosignalen aus dem Inhalt eines Bildschirmspeichers in einem Computer ist. Die Bezeichnung wird zumeist nur für einfache Realisierungen benutzt, welche zumeist nur Text-, Zeichen-, bzw. Pseudografik-Darstellungen ermöglichen und keine Hardwarebeschleunigung für 2D- oder 3D-Grafik bieten.

Einen solchen einfachen zeichenbasierten Videocontroller werden wir jetzt „in Software“ (Firmware) selbst realisieren. Unser Videocontroller wird außerdem im „Genlock“-Modus arbeiten, sich also auf ein fremdes Video-Eingangssignal synchronisieren und sein Video-Ausgangssignal dem Takt des Fremdsignals anpassen, denn nur so kann etwas in das Videosignal eingeblendet werden.

Der Grund, wieso wir uns auf zeichenbasierte Darstellung beschränken und nicht etwa Vollgrafik darstellen ist einfach: Unser XC164 hat nur 4KB RAM.

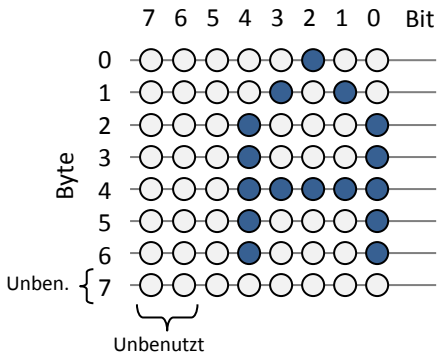
Da ein Videosignal ein serielles analoges Signal ist, benötigen wir eine Möglichkeit, möglichst schnell zeitlich wechselnde Spannungswerte auszugeben. Zu diesem Zweck nutzen wir die Datenleitung der **SSC-Schnittstelle** auf eine kreative Weise: Als ein im Chip eingebautes **Schieberegister**. Aufgrund der Tatsache, dass der aktive Bereich eines Videozeile nur 52µs lang ist, kommt ein Umschalten von Port-Pins nicht in Frage, da dazu viele CPU-Takte (besonders in C) notwendig sind und dadurch die horizontale Auflösung somit lediglich wenige Pixel betragen würde. Bei Nutzung der SSC braucht lediglich ein Bitmuster, welches einer **Zeile eines auszugebenden Zeichens** entspricht geladen werden und die SSC schiebt das Bitmuster selbstständig auf den Video-Ausgang, während dessen bereits das **nächste Bitmuster vorbereitet werden kann muss** – da sonst große Lücken zwischen den Zeichen entstehen würden.

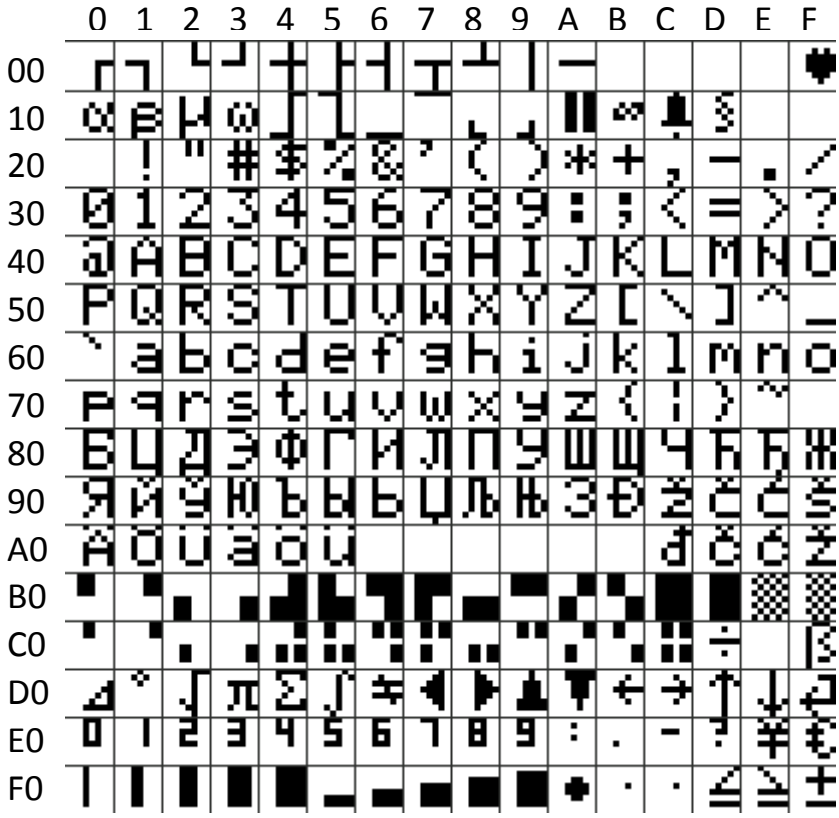


Wieviele Pixel bzw. Zeichen pro Zeile darstellbar sind hängt zunächst vom sog. Pixeltakt ab, der in unserem Fall der SSC-Takt ist und dem durch den CPU-Takt eine Grenze gesetzt ist. Weiters sollte der SSC-Takt natürlich sinnvollerweise nur so hoch gewählt werden, wie man auch in der Lage ist Daten nachzuliefern – andernfalls resultieren große Lücken zwischen kleinen Zeichen und eine unattraktive Darstellung.

Einen ASCII-kompatiblen 5x7-Punktmatrix-Zeichensatz in Form eines **unsigned char**-Arrays (8-Bit) finden Sie in der Datei **font.h**, welche Sie in ihr Projekt einbinden. Der Zeichensatz wird dann im Flash des XC164 abgelegt.

Zeichen „A“, ASCII-Code 65 (0x41)

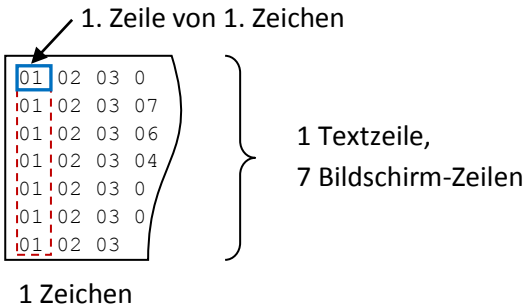




Sehen Sie sich die Bitmuster in jener Zeilen an, welche mit diesem Kommentar versehen ist, um den Aufbau des Zeichensatzes zu verstehen:

```
// 65 – A
```

Da das Laden von ASCII-Zeichen, Berechnen von Adressen und Laden von Bitmustern aus dem Flash für jedes Zeichen zu langsam wäre, müssen Sie eine **komplette Textzeile,** bestehend aus 7 Bildschirm-Zeilen, im RAM vorausberechnen, da zwischen 2 Bildschirmzeilen keine Rechenzeit bleibt – außer unter Inkaufnahme von Leerzeilen.



Die Ausgabe geschieht interrupt-gesteuert: Sobald ein Zeichen ausgegeben wurde, soll ein Interrupt ausgelöst werden und das nächste Bitmuster aus dem Puffer geladen und der SSC übergeben werden.

Versuchen Sie die SSC1 Interrupt-Routine so kurz wie möglich zu realisieren und nutzen Sie vorteilhafterweise Pointer statt (mehrdimensionaler) Arrays. Vermeiden Sie Berechnungen und reduzieren Sie if-Abfragen in Interrupt-Routinen auf ein Minimum.

Überlegen Sie, ob alles, das in der SSC1-Interrupt-Routine berechnet wird auch wirklich dort berechnet werden muss.

→ Konfiguration der SSC1:

- Wählen Sie eine adäquate Bitrate – am besten ca. 3-5 MHz – eine niedrigere Bitrate ist anfangs einfacher zu implementieren
- Die SSC1 arbeitet im Master-Modus
- Baudrate, Phase, Receive und Transmit Error (Fehlererkennung), Loop Back sowie Clock-Phase/Polarity sind irrelevant bzw. können auf der Standardeinstellung (0) verbleiben
- Wird zuerst das MSB oder das LSB herausgeschoben?
- Welche Sende-Wortbreite soll eingestellt werden?
- Es sollte ein Interrupt nach jedem übertragenen Bitmuster ausgelöst werden

Die SSC gibt im Idle-Zustand einen Logisch-1 Pegel aus. Was müssen Sie daher unmittelbar vor Beginn und unmittelbar nach Ende der Einblendung beachten?

Um den Text horizontal zu positionieren können Sie die Funktion aus der Datei **delay_horiz.c** übernehmen. Achten Sie darauf, dass gleich viele Programmbefehle zu Beginn jeder Zeile mit eingeblendetem Text ausgeführt werden.

4.2 „Fast external interrupts“

Die „Fast external interrupts“ sind wie normale Interrupt-Leitungen auf den meisten anderen Mikrocontrollern bzw. Mikroprozessoren.

Zur Verwendung der „Fast external interrupts“ ist es erforderlich, die entsprechenden Register zu entsperren, da sie standardmäßig für Zugriffe gesperrt sind. Übernehmen Sie dazu die in der Datei **unlock.c** definierte Funktion und rufen Sie diese vor Verwendung der „Fast external interrupts“ in ihrer main-Funktion auf. Die sog. „*magische Sequenz*“ zur Entsperrung der Register ist so von Infineon definiert und muss exakt in dieser Reihenfolge bestehen bleiben.

→ Konfiguration der „Fast external interrupts“:

- Welche Flanken werden bei EXICON eingestellt?
- Welche Pins und Modi müssen bei EXISEL0, EXISEL1 eingestellt werden?

Die genaue Funktion der „Fast external interrupts“ entnehmen Sie bitte dem XC164-Peripherie-Handbuch.

4.3 Analog Digital Converter

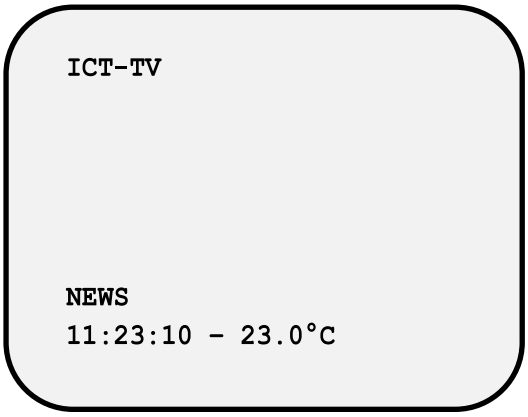
→ Konfiguration des Analog-Digital-Converters:

- Fixed Channel Single-Conversion Mode
- Conversion-Time, Sample Time und Auflösung können beliebig gewählt bzw. auf Standardeinstellung (0) belassen werden
- Nach Abschluss der Messung soll ein Interrupt ausgelöst werden

Wann starten Sie die ADC-Messungen am besten?

4.4 Funktionsbeschreibung / Aufgabenstellung

Erstellen Sie ein Programm, welches sich auf das Eingangs-Videosignal synchronisiert und textuelle Informationen ins Videobild einblendet. Bei der Gestaltung des Layouts dürfen Sie Ihrer Kreativität freien Lauf lassen – es sollen jedoch mindestens ein fester Text, eine simulierte Uhr und die aktuell gemessene Temperatur eingeblendet werden.



Wie können Sie eine Uhr auf einfachste Weise realisieren?

5. Empfohlener Übungsablauf

5.1 Aufbau des Programms

Bauen Sie Ihr Programm modular auf! Das Programm besteht aus mehreren Teilaufgaben, welche Sie nacheinander realisieren sollten.

→ Mögliche Lösungsschritte

- Synchronisation auf das Video-Signal
- Ausgabe von Bitmustern bzw. Pixeln
- Generierung von Text
- Messung der Zeit
- Messung der Temperatur

Nutzen Sie, insbesondere bei zeitkritischen Operationen das Schlüsselwort **__inline**

```
__inline void timecritical_function()
{
……….
}
```

Dadurch wird der Compiler angewiesen, soweit möglich, den Programmcode einer Funktion **direkt** dort einzufügen, wo die Funktion aufgerufen wird und einen Funktionsaufruf zu vermeiden. Dies sollte nur für kurze Funktionen, zeitkritische Stellen oder zur Optimierung der Geschwindigkeit benutzt werden, da hierbei der Programmcode dupliziert wird.

5.2 Testen des Programms

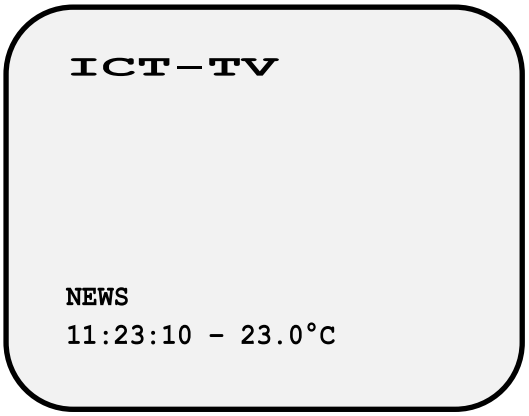
Sie können die LEDs auf dem Microcontroller-Board zu Test- und Debugging-Zwecken nutzen. Es ist insbesondere ratsam durch Toggling der LEDs die Erkennung von Zeilen und Halbbildern anzuzeigen.

Bei diesem Laborversuch können Sie aufgrund der Tatsache, dass Sie die viele Effekte direkt sehen können auch einiges einfach ausprobieren und sich so herantasten.

5.3 Variationen

Seien Sie darauf gefasst, dass während der Laborübung zusätzliche Teilaufgaben oder Features zu Ihrem Beispiel von den Tutoren hinzugefügt werden können.

- Realisieren Sie unterschiedliche Schriftbreiten



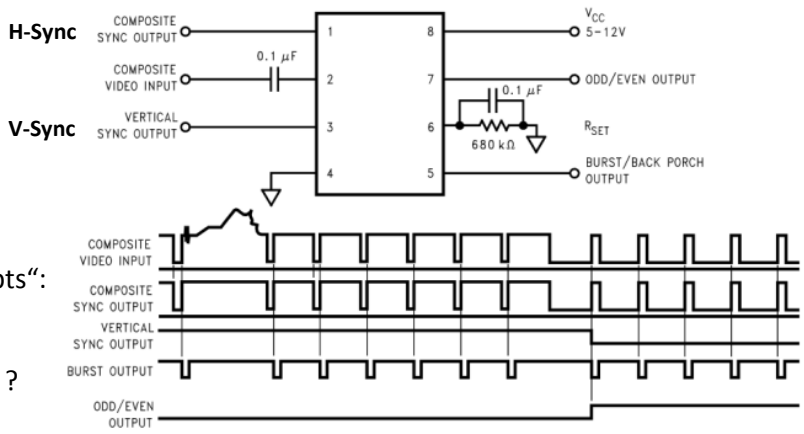
- Realisieren Sie eine Laufschrift

- Versuchen Sie, durch effiziente Programmierung mehr Zeichen in einer Zeile darzustellen. Eventuell können Sie, durch Austausch der Startup-Datei **START_V2.A66** durch jene im Turbo-Verzeichnis) den XC164 zu übertakten.

Zusammenfassung („Cheat Sheet“)

→ Mikrocontroller Pins:

- EX0IN = H-Sync
- EX4IN = V-Sync
- AN0 = Temperatursensor
- MTSR1 = Video-Ausgabe



→ Konfiguration der „Fast external interrupts“:

- Flanken in EXICON?
- Pins und Modi bei EXISEL0, EXISEL1 ?

→ Konfiguration der SSC1:

- Bitrate ca. 3-5 MHz
- Master-Modus
- Baudrate, Phase, Receive, Transmit Error, Loop Back sowie Clock-Phase/Polarity irrelevant
- LSB/MSB zuerst?
- Sende-Wortbreite?
- Interrupt auslösen

Die SSC gibt im Idle-Zustand einen Logisch-1 Pegel aus!

→ Konfiguration des Analog-Digital-Converters:

- Fixed Channel Single-Conversion Mode
- Conversion-Time, Sample Time und Auflösung beliebig
- Interrupt auslösen

Wann Messung starten?

Viel Erfolg!