Report for ARM Microcontroller Project

uLoader WiSe 2014

Appel, Dennis (s813783)
Voigt, Alexander (s814526)
Merrikhi, Pedram (s882217)

April 8, 2015

Contents

List of Figures

List of Tables

Listings

1. Introduction

Nowadays the usage of the ARM-MCU could be used in every aspect of everyday life. Additionally, the ARM processor is the number one architecture of choice in many market segments.
This project is based on the development of a bootloaders and its implementation inside a network.
The usage stm32f4-discovery Board is a prefered and viewed as an "'Allrounder"' for such a project. The reasoning behind this is the "value for money" and user-friendliness. This allows for an easy introduction into the world of ARM Microcontroller unit programming.
The ARM-Cortex-M4-Prozessor found on the STM32f4-discovery board prossesses the principel parts shown in the figure below.

1.1. The Aim

The aim of the project is to research the feasibility to create a quick, cheap and easy to use way of utilizing an STM32 Microcontroller to communicate between a user and a remote device.
The purpose of the application is meant to be a first step fundamental strategy to creating a product for future projects.
It is hoped that by way of HTML communication, a fast and light application could be used to fulfill the desire of a user to achieve a particular objective such as threeway handshake signal to verify a particular device in order to transmit information such as codes or messages, using a TCP/IP protocol stake.
As it can be demonstrated the different applications could be endless.

Es wird später ein weiteres unterschiedliches Diagram verwendet. Dies ist nicht falsch, sondern dient der sinngemä"sen Darstellung des Prozessors.

Zunächst soll ein Überblick üeber die behandelten Themenschwerpunkte gegeben werden

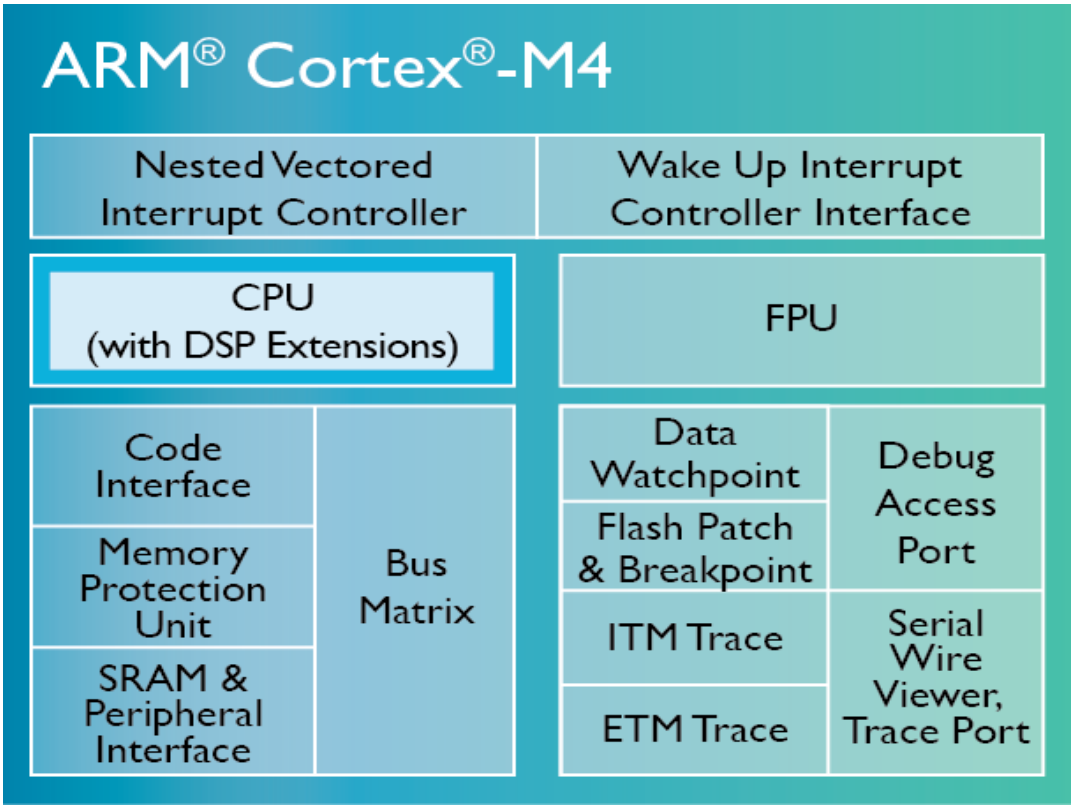

Figure 1.1.: Prinzipieller Aufbau

1.2. Bootloader

The purpose of a Bootloader program is to allow the installation and utilisation of any program that could be reloaded. Whereas the program that is currently loaded is also being run.
Next it is necessary to initialise the hardware, that would in turn be needed to load the program.
The "STM32f407 discovery board" offers three different methodes to boot up the hardware.
In order to switch between the three different boot methods, the Boot-Pins BOOT1 and BOOT2 could be set:

BOOT1	BOOT2	Boot-Mode	Adresse
x	0	Flash Memory (User Flash)	0x8000_0000
0	1	System Memory	0x1FFF_F000
1	1	SRAM	0x2000_0000

Der ROM, hier System Memory, beinhaltet den, vom Hersteller, mitgegebenen Boot-

loader.
Wie man den Speicher dann belegt ist einem freigestellt. Man muss dann darauf achten, dass man an die richtige Adresse springt, wenn man das Programm nachgeladen hat.
Als Beispiel-Vorgehensweise kann man sagen, dass die Reihenfolge, nachdem der Bootloader bereits geladen ist, folgende ist:

1. Verwendete Hardware initialisieren (USB/USART/RCC...)
2. Auf eingehendes Programm warten (wenn sonst keine Aufgabe ansteht)
3. Eingehendes Programm an Adresse XY schreiben.
4. An Adresse XY springen

Das eingegangene Programm ist dann natürlich selbst für die Initialisierung der verwendeten Hardware zuständig.

1.3. SWD

Bei der Entwicklung kam die Serial Wire Debug Technologie zum Einsatz. Hierbei handelt es sich um einen Debug-Port, der speziell dafür entwickelt wurde um MCU bzw. Projekte mit MCU, bei denen so wenig wie möglich Pins verwendet werden sollen.
Dieser Port besteht aus den Leitungen folgender Tabelle:

Pin	Signal	Type	Beschreibung
1	VTref	Input	This is the target reference voltage. It is used to check if the target has power, to create the logic-level reference for the input comparators and to control the output logic levels to the target. It is normally fed from Vdd of the target board and must not have a series resistor.
7	SWDIO	I/O	Single bi-directional data pin
9	SWCLK	Output	Clock signal to target CPU. It is recommended that this pin is pulled to a defined state of the target board. Typically connected to TCK of target CPU.
13	SWO	Output	Serial Wire Output trace port. (Optional, not required for SWD communication)
15	RESET	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".
19	5V-Supply	Output	This pin is used to supply power to some eval boards. Not all JLinks supply power on this pin, only the KS (Kickstart) versions. Typically left open on target hardware.

Die anderen Leitungen des bisherigen 20-poligen Anschlussen wurden weggelassen, weil sie entweder für SWD uinteressant sind oder sie auf GND gelegt sind. Egal was davon zutrifft, sie haben keinen Einfluss auf die SWD-Kommunikation.

Es ist eine neue, sehr interessante Weise zu debuggen. Bisher war JTAG das Debugger-Interface. Die Vorteile dieser Technologie sind (frei von der ARM-Website übersetzt):

- Nur 2 Pins werden belegt
- JTAG TAP controller kompatibel
- Erlaubt dem Debugger ein weiterer AMBA-Bus-Master zu werden um auf Register / Speicher zuzugreifen.
- High Datarates - 4Mbytes/sec @50MHz
- Low Power - keine zusätzlichen Versorgungsspannung
- gute "'built in"' Fehler-Erkennung
- Schutz vor Fehlern bei Kontaktverlust

1.4. startup

- stack, program counter, interrupt, vector table, initial system clock

1.5. CMSIS

Der ARM Cortex Microcontroller Software Interface Standard ist eine Händlerunabhängige Abstraktionsschicht für die Cortex-M Prozessoren.
Dabei ist CMSIS unterteilt in:

- CMSIS-CORE - API zum Zugriff auf den Prozessorkern sowie Perepherie-Register.
- CMSIS-Driver - Generischer Zugriff auf Perepherie für die Middleware (reusability).
- CMSIS-DSP - DSP Bibliothek mit über 60 Funktionen
- CMSIS-RTOS API - Standardisiertes (RTOS kompatibel)
- CMSIS-Pack - Beschreibung der wichtigen Bestandteile (Nutzersicht)
- CMSIS-SVD - Beschreibung der wichtigen Bestandteile (Systemsicht)

- CMSIS-DAP - Debug Access Port

Zusammengefasst erlaubt CMSIS einheitliche und simple Software Schnittstelle zu Prozessor und die Perepherie, sowie Echtzeit OS (RTOS) und Middleware.

1.6. Nested Vectored Interrupt Controller

Der NVIC bietet die Möglichkeit gewisse Interrupts zu konfigurieren (Priorität, aktiviert, deaktiviert...).
Neben vorgegebenen Interrupts sind hier auch die Implementierungsabhängigen Interrupts konfigurierbar. Da die ersten 15 Interrupts vorgegeben sind, können die implementierten Interrupts in der Anzahl von 0 bis 240 reichen.

1.7. Unterschiede

GNU, KEIL iar

1.8. Netzwerk

was sollte zum besseren verstehen hier einen platz finden.