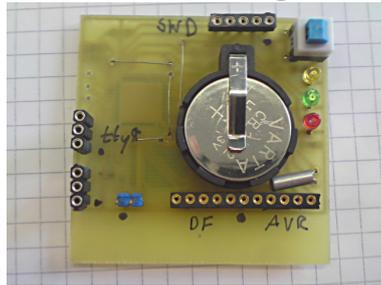


# MXE11: Unix auf dem Mikrocontroller

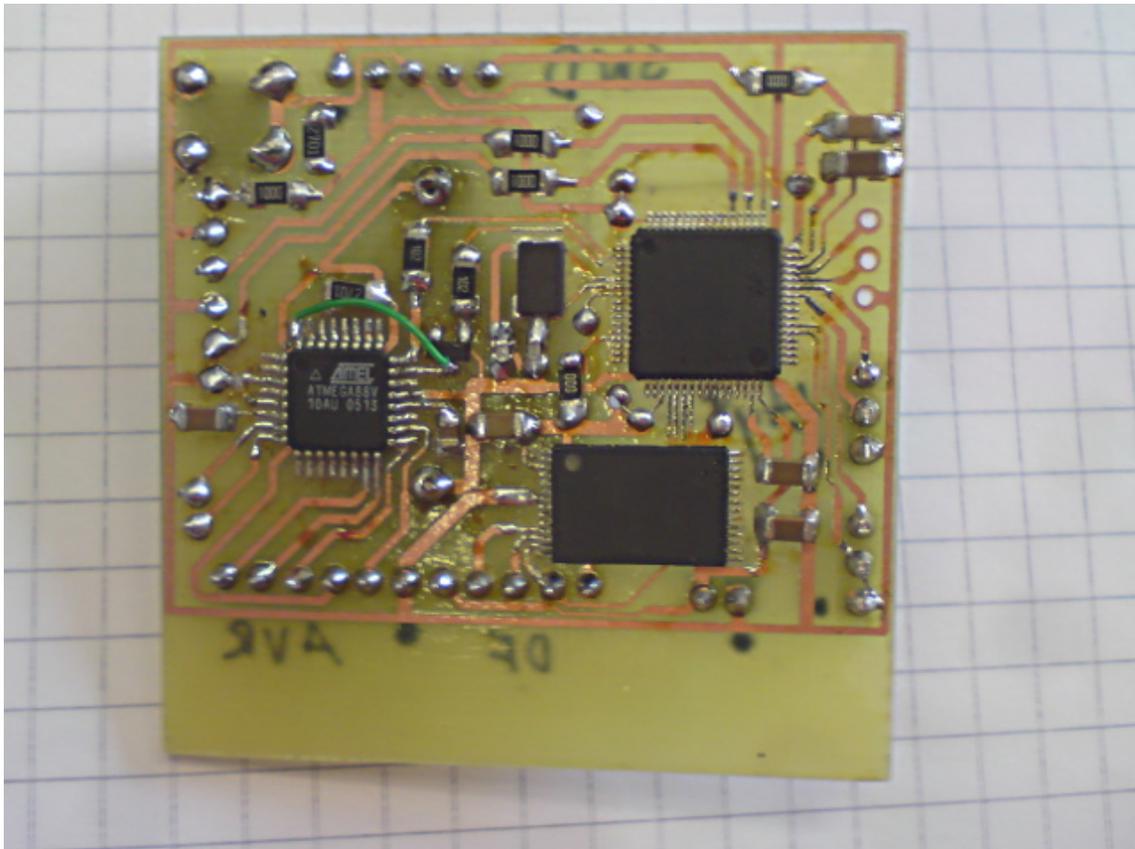
V1.71 (c) 2017-2018 Jörg Wolfram



[0.5cm]

## 1 Grundversion

Gleich vornweg: Eine 1:1 Nachbauanleitung für alle möglichen Varianten gibt es nicht, da ich meist die Bauteile nehme, die ich gerade da habe oder von alten Projekten und Leiterplatten „recyclen“ kann. Lediglich für die STM32F405-Variante gibt es ein relativ aktuelles Layout.



Das Layout ist einseitig (mit ein paar Drahtbrücken) und beinhaltet den STM32F405RG, einen AT45DB642 und einen ATmega88 als RTC. Es ist mit gEDA PCB erstellt und befindet sich im **doc** Archiv. Ich habe es zur Zeit nicht weiter dokumentiert, da meistens andere Layoutprogramme bevorzugt werden.

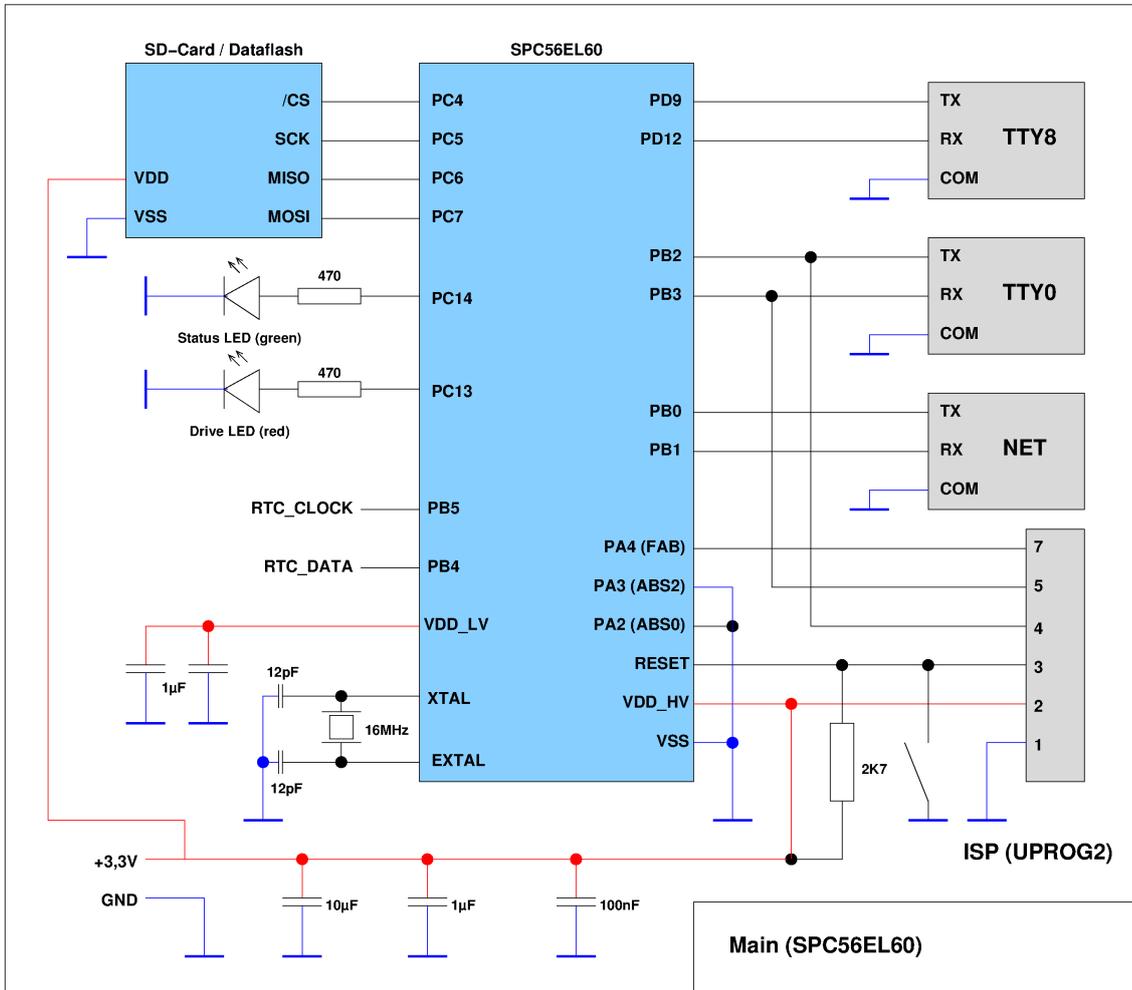
### 1.1 Schaltung

Im Schaltplan sind meist nur die Signalbezeichnungen angegeben. Als externer Datenträger können entweder ein AT45DB642 Dataflash (Binary page mode) oder eine SD-Karte verwendet werden. Die Anschlussbelegung am Controller ist in beiden Fällen gleich. Als serielle Schnittstellen sind momentan TTY8 (/dev/tty8) und TTY0 (/dev/tty0) vorgesehen, wobei TTY8 die primäre Konsole ist. Als ISP-Schnittstelle sind die Anschlüsse vom UPROG2 angegeben. Falls ein AT45DB642

eingesetzt werden soll, empfehle ich, anstatt eines Reset-Tasters einen Rastschalter aufzubauen. Damit lässt sich der Mikrocontroller während der Programmierung des Flashes deaktivieren und stört nicht die Programmierung.

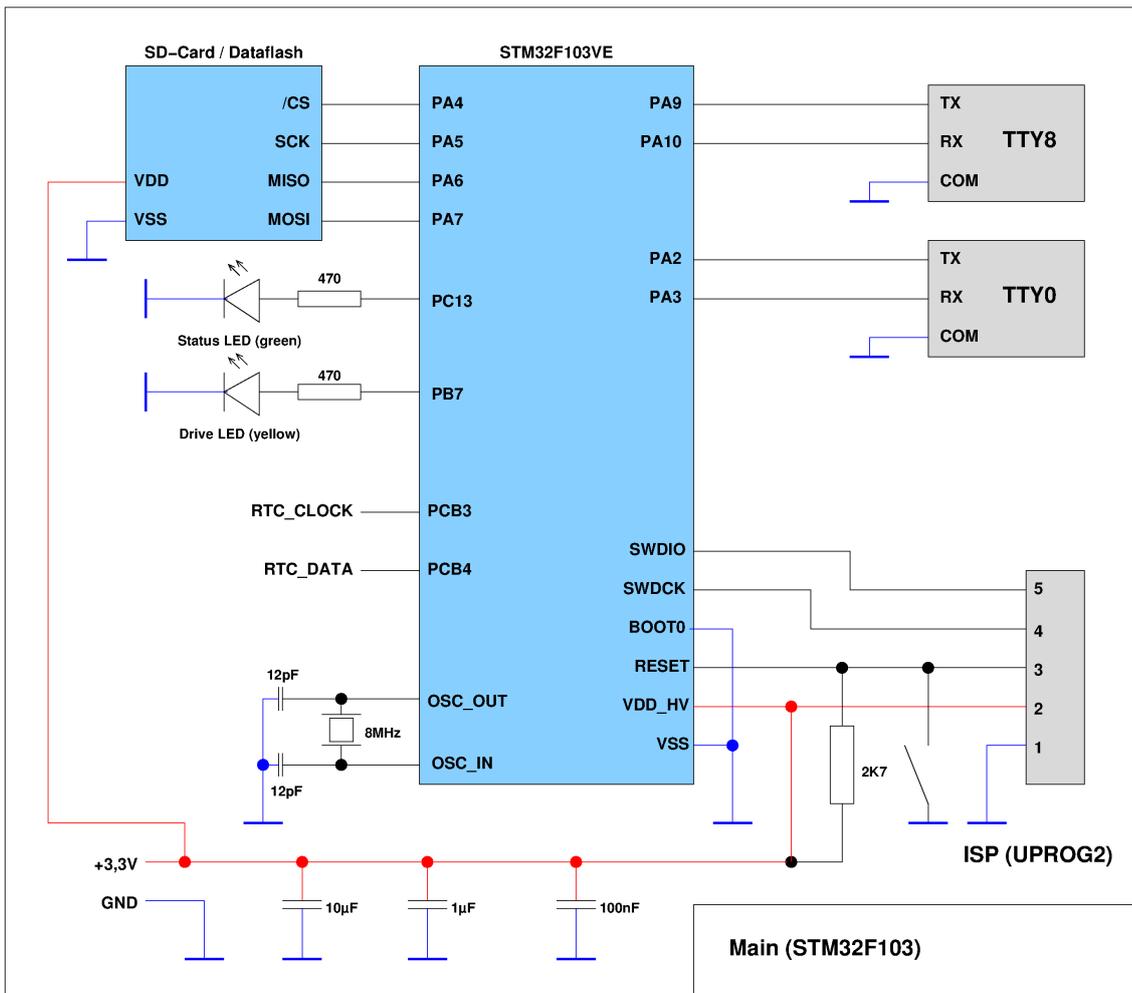
## 1.2 SPC56EL60

das ist die Ur-Version, die ich mit einem Testboard aufgebaut habe. Die Quarzfrequenz beträgt 16MHz, die CPU-Frequenz 120MHz. Zur Zeit wird nur 1 Core verwendet.



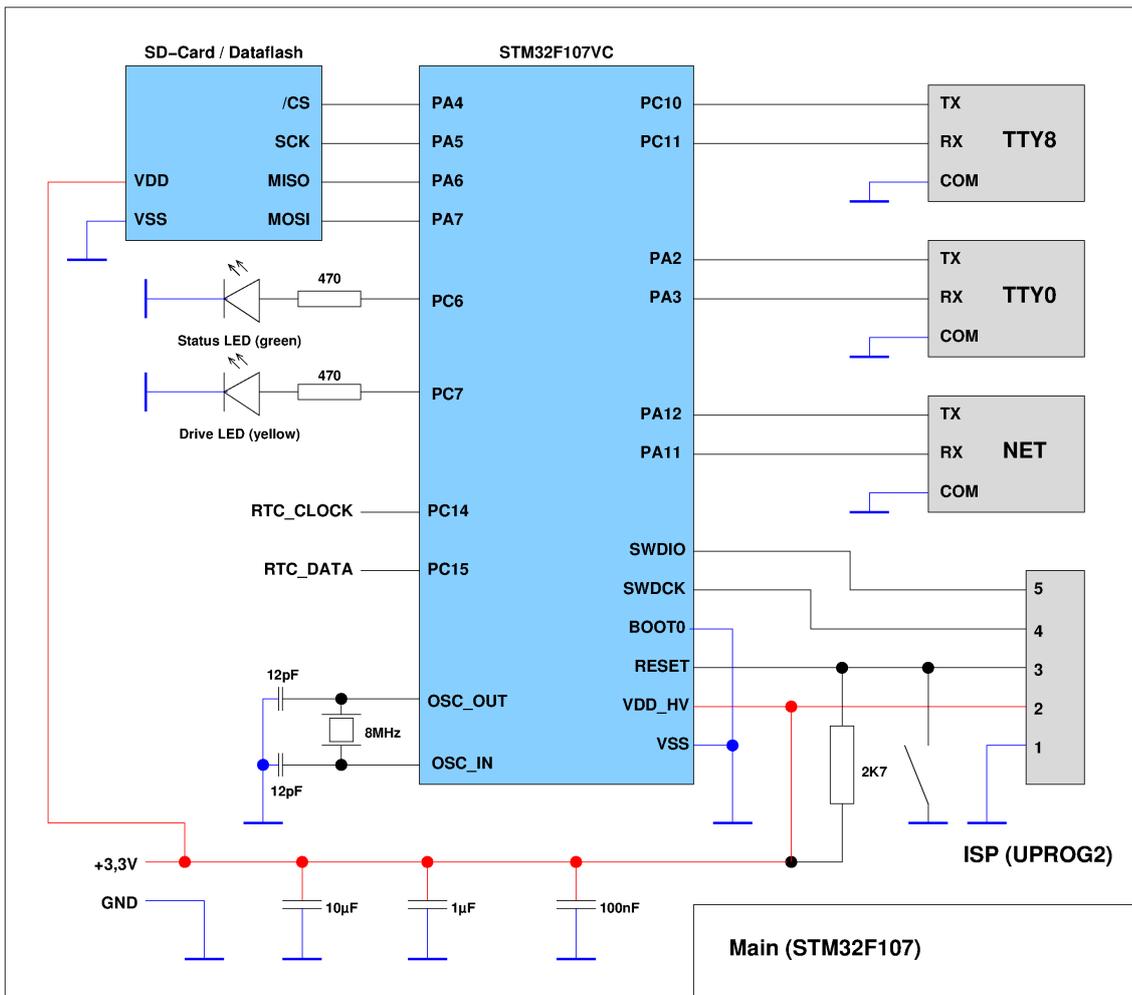
## 1.3 STM32F103

Testversion auf einem eigenen SMT32-duino kompatiblen Board, Quarzfrequenz liegt bei 8MHz. Der Controller wurde via Overclocking auf 120MHz gebracht, das lässt sich aber in der `board.h` abschalten (`#define F1_OVERCLOCKING` auskommentieren)



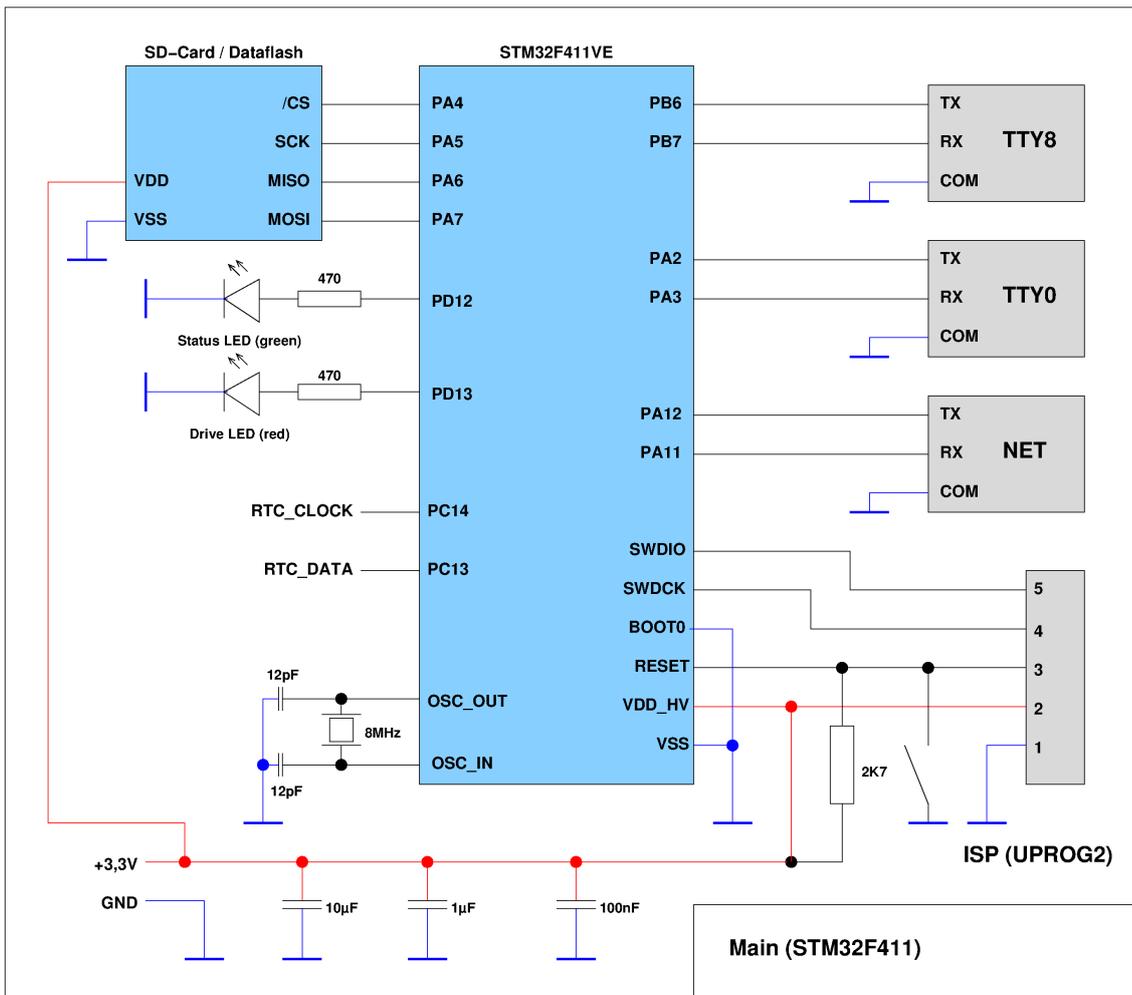
## 1.4 STM32F107

Als Board habe ich das Olimex STM32-H107 Board verwendet, Quarzfrequenz liegt bei 8MHz. Overclocking lief bei mir nicht stabil, daher 72MHz Taktfrequenz.



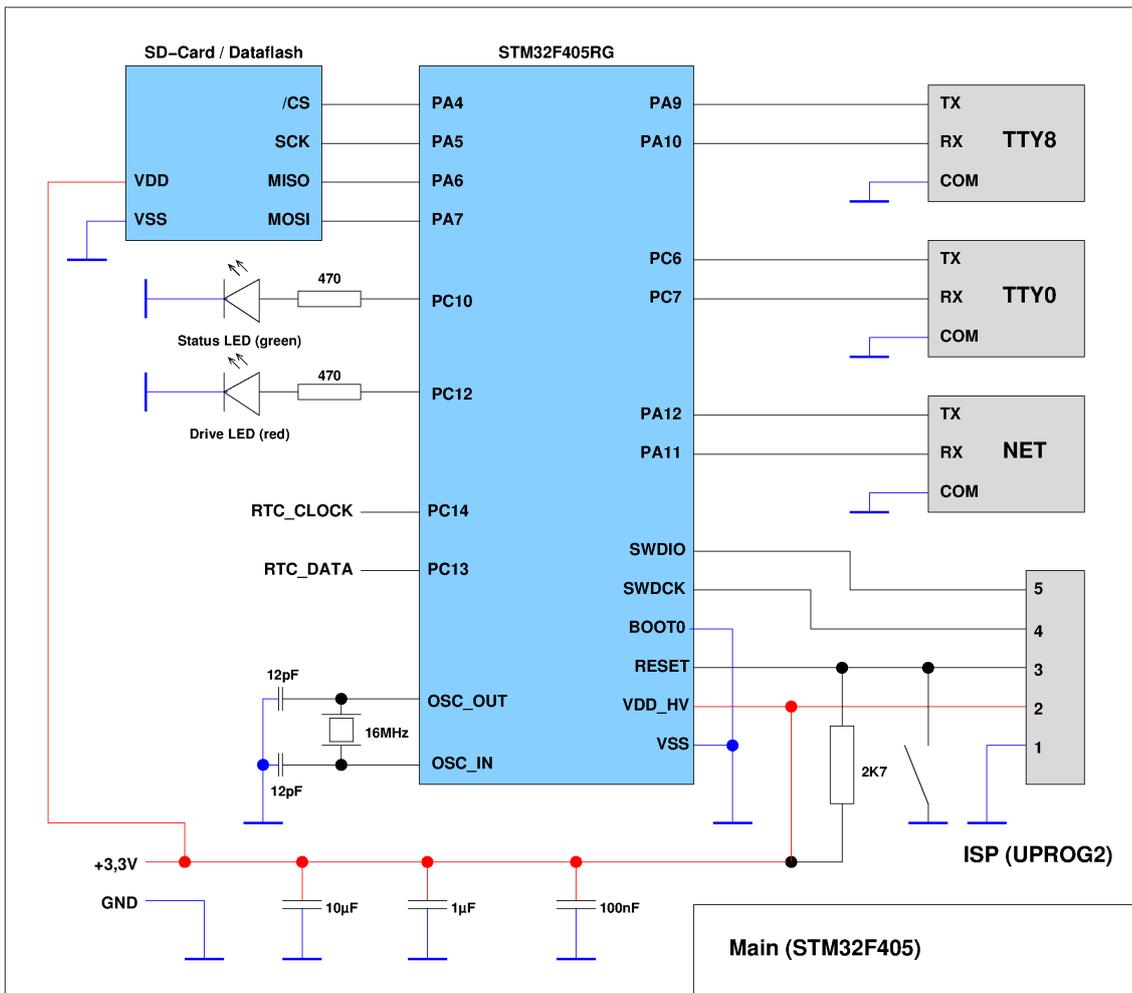
## 1.5 STM32F411

Als Board habe ich das STM32F411E-Discovery Board verwendet, Quarzfrequenz liegt bei 8MHz.



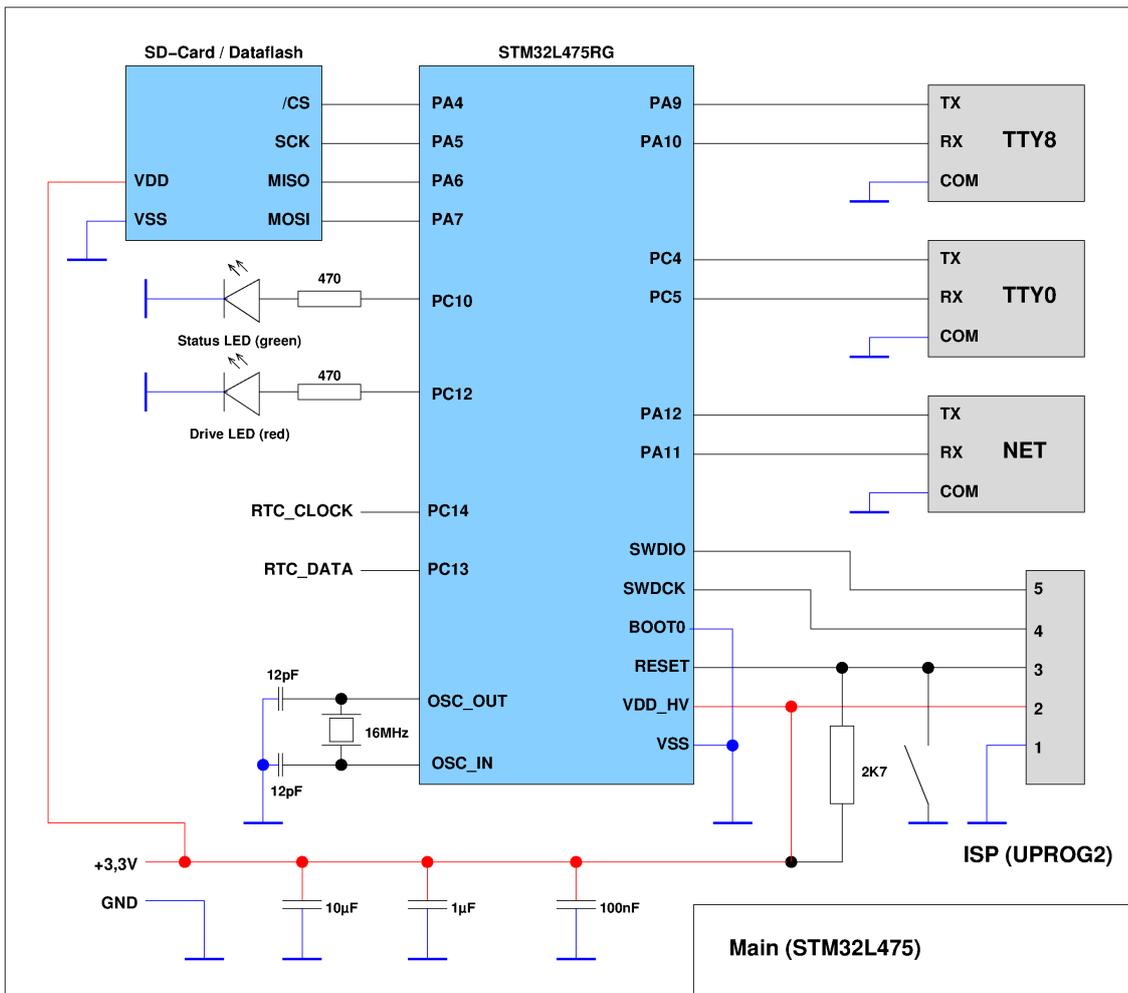
## 1.6 STM32F405

Board ist hier ein eigenes Board für den MXE11, Quarzfrequenz liegt bei 16MHz.



## 1.7 STM32L475

Board ist hier das MXE11-Board wie beim STM32F405, lediglich die Signale für TTY0 wurden parallelverdrahtet, da sie auf PC4/PC5 anstelle PC6/PC7 liegen. Die Quarzfrequenz liegt ebenfalls bei 16MHz.

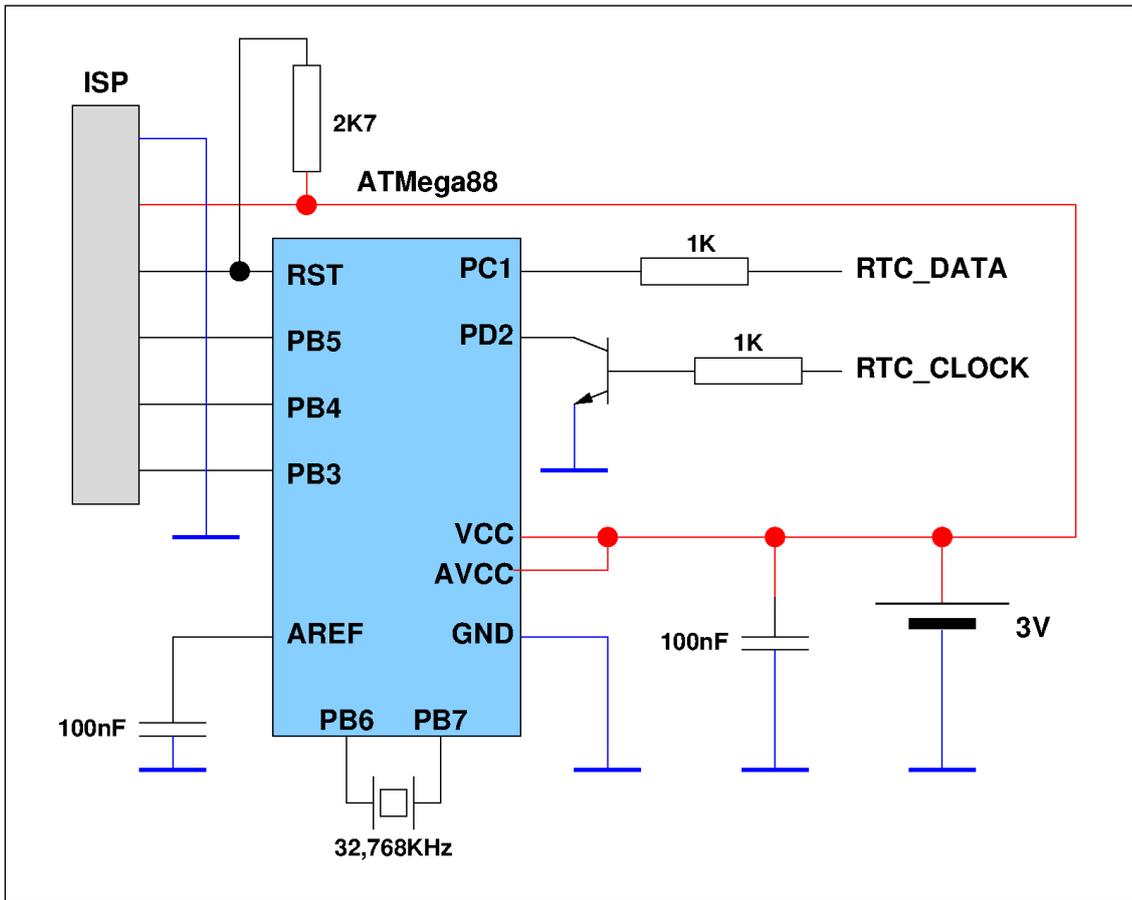


## 2 Die RTC

Auch wenn einige der verwendeten Mikrocontroller eine integrierte RTC besitzen (STM32), habe ich eine eigene (externe) Lösung entwickelt, die mit allen eingesetzten Controllern funktioniert. Über eine 2-Draht Schnittstelle sind neben dem Sekundenregister (32 Bit) noch 15 weitere batteriegestützte 32-Bit Register zur universellen Verwendung verfügbar.

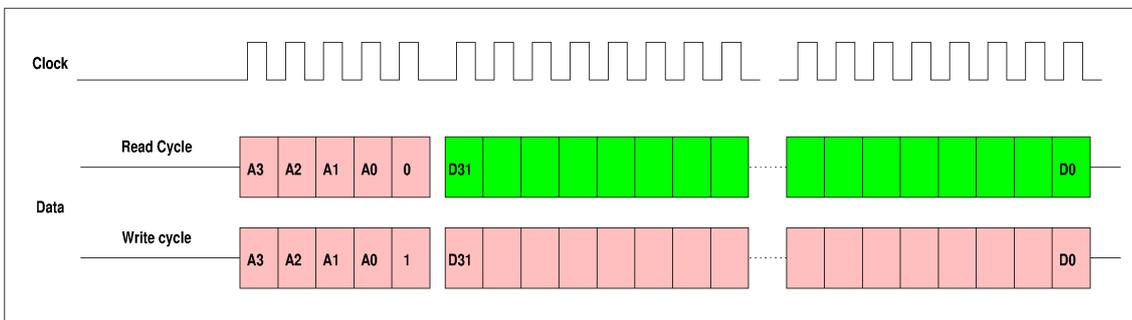
### 2.1 Schaltung

Kernstück der Schaltung ist im konkreten Fall ein ATmega48/88/168/328, der genaue Typ ist zweitrangig. Der Controller läuft mit internem RC-Oszillator, zusätzlich ist ein 32758 Hz Uhrenquarz notwendig. Die Kommunikation mit dem Host-Controller erfolgt über zwei Leitungen (Clock und Data), wobei die Clock-Leitung über einen NPN-Transistor invertiert wird. Die Datenleitung erhält einen Entkopplungswiderstand von 1KOhm, um Signalkonflikte während der Richtungs-umschaltung zu vermeiden. Versorgt wird der ATmega über eine 3V Lithium-Knopfzelle.



## 2.2 Protokoll

Die Kommunikation seitens des Host-Controllers erfolgt über 2 frei wählbare I/O Pins und softwareseitig über meine **Unilib**. Der RTC-Controller befindet sich die meiste Zeit im Sleep und wird in regelmäßigen Abständen über den Timer 2 aufgeweckt. Als weitere Interrupt-Quelle dient INT0, der über den externen Transistor mit der Clock-Leitung verbunden ist. Die maximale Taktfrequenz für die Übertragung beträgt ca. 16KHz. Da das Auslesen der RTC nur beim Start des Emulators erfolgt, spielt die für die Kommunikation erforderliche Zeit nur eine untergeordnete Rolle. Danach werden clocksynchron 4 Adressbits und ein R/W Bit zur RTC gesendet. Abhängig von R/W Bit folgen dann 32 Bits Daten entweder von der RTC zum Host oder vom Host zur RTC. Danach ist die Kommunikation abgeschlossen und die RTC geht wieder in den Sleep-Mode.



Bei einer unvollständigen Kommunikation bricht die RTC nach ca. 0,5 Sekunden mit einem Timeout ab und geht wieder in den Sleep-Mode. Auf diese Weise werden kurze Störimpulse ausgefiltert.

## 2.3 Software

Die Software auf dem ATmega ist in Assembler geschrieben und nutzt sowohl Timer 2 als auch Int0 als Wake-Up und Interrupt-Quellen.

Die Fuse-Einstellungen für einen ATmega88 lauten:

LOW-FUSE        0xD2  
HIGH-FUSE      0xDF  
EXT-FUSE        0xFF

### 3 Erweiterungen

Als Erweiterung sind später eine Netzwerkfunktion via CAN, 4 analoge Inputs, 4 analoge Outputs (via PWM) und jeweils 4 digitale Inputs/Outputs vorgesehen. Bei eigenen Layouts sollten die entsprechenden Pins bereits mit berücksichtigt werden.

Am sinnvollsten wird es sein, die I/O als Aufsteckplatine zu realisieren, damit wird auch das Layout der Basisplatine einfacher. Die nachfolgenden Tabellen enthalten lediglich die vorgesehenen Pins, die konkrete Schaltung (insbes. Entkopplung) ist noch nicht festgelegt.

#### 3.1 SPC56EL60

Signal	Pin
CAN TX	PB0
RAN RX	PB1
AIN_0	PB7
AIN_1	PB8
AIN_2	PB9
AIN_3	PB10
AOUT_0	PA10
AOUT_1	PA11
AOUT_2	PA12
AOUT_3	PA13
DIN_0	PB11
DIN_1	PP12
DIN_2	PB13
DIN_3	PB14
DOUT_0	PA0
DOUT_1	PA1
DOUT_2	PD10
DOUT_3	PD11

#### 3.2 STM32F103

CAN entfällt hier.

<b>Signal</b>	<b>Pin</b>
CAN TX	—
RAN RX	—
AIN_0	PA0
AIN_1	PA1
AIN_2	PB0
AIN_3	PB1
AOUT_0	PC6
AOUT_1	PC7
AOUT_2	PC8
AOUT_3	PC9
DIN_0	PC0
DIN_1	PC1
DIN_2	PC2
DIN_3	PC3
DOUT_0	PB12
DOUT_1	PB13
DOUT_2	PB14
DOUT_3	PB15

### 3.3 STM32F107

Board ist das Olimex STM32-H107 Board.

<b>Signal</b>	<b>Pin</b>
CAN TX	PA12
RAN RX	PA11
AIN_0	PA0
AIN_1	PA1
AIN_2	PB0
AIN_3	PB1
AOUT_0	PC6
AOUT_1	PC7
AOUT_2	PC8
AOUT_3	PC9
DIN_0	PC0
DIN_1	PC1
DIN_2	PC2
DIN_3	PC3
DOUT_0	PB12
DOUT_1	PB13
DOUT_2	PB14
DOUT_3	PB15

### 3.4 STM32F411

Board ist hier das STM32F411E-Discovery Board, Signale sind zur Zeit noch nicht definiert.

### 3.5 STM32F405

Board ist hier ein eigenes Board für den MXE11 (in Vorbereitung).

<b>Signal</b>	<b>Pin</b>
CAN TX	PA12
RAN RX	PA11
AIN_0	PC4
AIN_1	PC5
AIN_2	PB0
AIN_3	PB1
AOUT_0	PC6
AOUT_1	PC7
AOUT_2	PC8
AOUT_3	PC9
DIN_0	PC0
DIN_1	PC1
DIN_2	PC2
DIN_3	PC3
DOUT_0	PB12
DOUT_1	PB13
DOUT_2	PB14
DOUT_3	PB15

### 3.6 STM32L475

Noch kein Board spezifiziert, Signale sind zur Zeit noch nicht definiert.