STM32*breakout*v2

Dokument: 1.0 Alpha, Hardware 14/2, ED14101

STM32 Breakoutboard*v2* - Fx-Serie

[Manual]

Version mit STM32F105RBT6

Autor: C. Hediger, 08.11.2014

Inhaltsverzeichnis

Änderungsverzeichnis		
Editiert von	**Bemerkung**	**Datum**
C. Hediger	Erste Version erstellt 1.0 Alpha	08.11.2014

1. Einleitung

Dieses Dokument, richtet sich an all jene welche mit dem Breakoutboard-V2 entwickeln.

Es enthält alle wichtigen Informationen zum Board und zur ersten Inbetriebnahme.

Es wird ebenfalls auf die verschiedenen Bestückungsvarianten eingegangen.

2. Beschreibung

Dieses Breakoutboard wurde entwickelt, um die Verwendung der STM32-Mikrocontroller der Serie F1, F2, F4 zu vereinfachen.

Eines der Hauptmerkmale dieses Boards ist, dass es in ein doppel-Breadboard passt. Dies deshalb, da die Stiftleisten im 2.54mm Raster angeordnet sind.

Da alle Controller der STM32Fx-Serie im LQFP64 Gehäuse untereinander pinkompatibel sind, eignet sich dieses Board für sämtliche Controller der entsprechenden Linien.

Das Board bietet zudem die Möglichkeit, drei verschiedene Speisungen zu bestücken. W

2.2 Übersicht

Nachfolgend sind einige Hauptmerkmale des Boards aufgeführt.

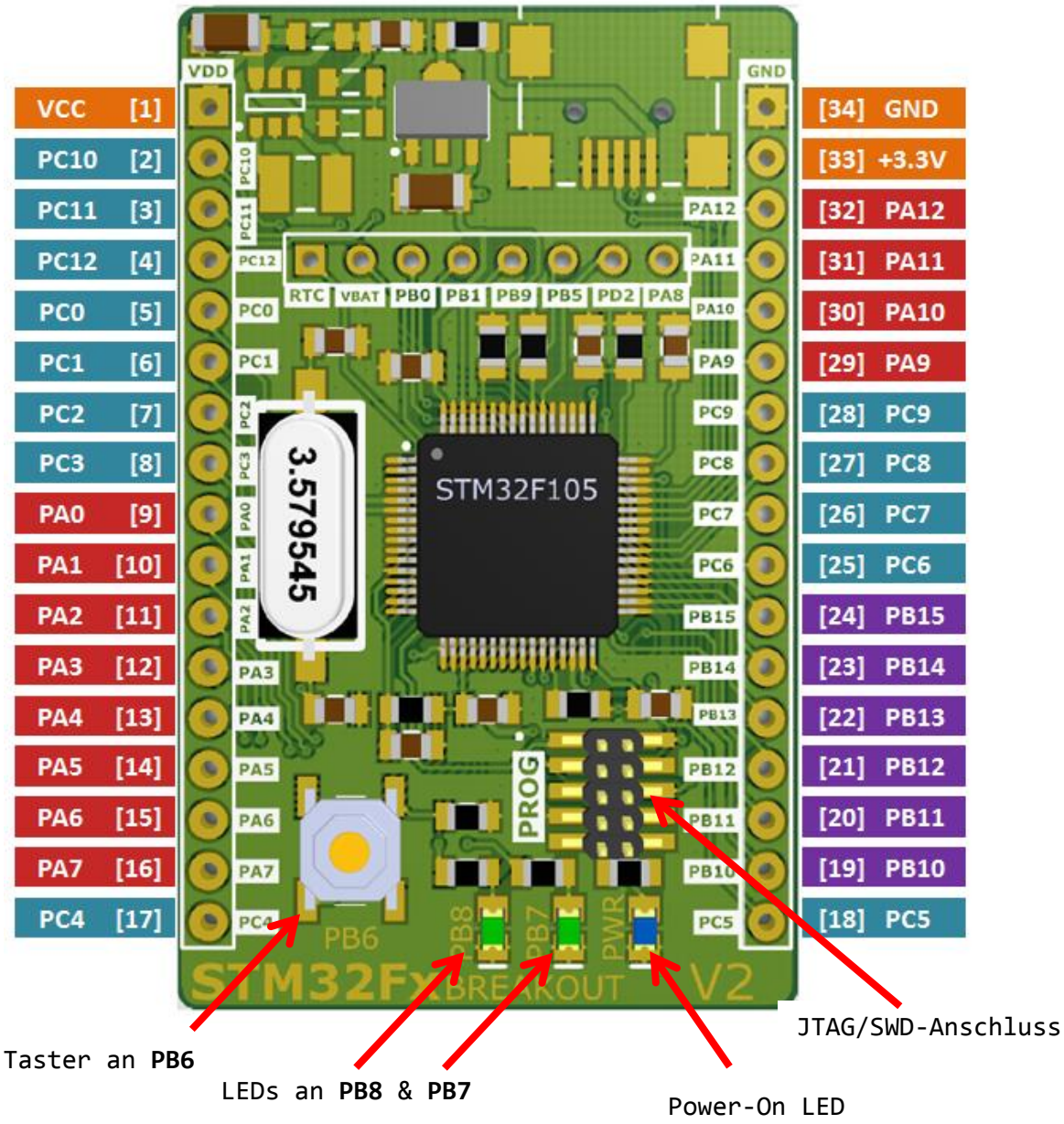
Stromversorgung:
Das Board kann extern mit **0.8-3.3V** versorgt werden.
Bei bestücktem LDO beträgt die Spannung **3.4V* (3.6V)-16V****

Anzahl I/O's	**37**
Anzahl ADC Eingänge	**16**
Anzahl UART	**3**
JTAG**	**Ja & SWD**
VCC	**0.8-16V**

* Die minimale Eingangsspannung ist Lastabhängig. 3.4V gilt bis ca. max. 50mA Last.
** 16V Eingangsspannung ist nur mit entsprechender Kühlung des LDO's möglich!

2.3 Pinbelegung, HW-Beschreibung

Nachfolgend ist die Pinbelegung des Boards ersichtlich.



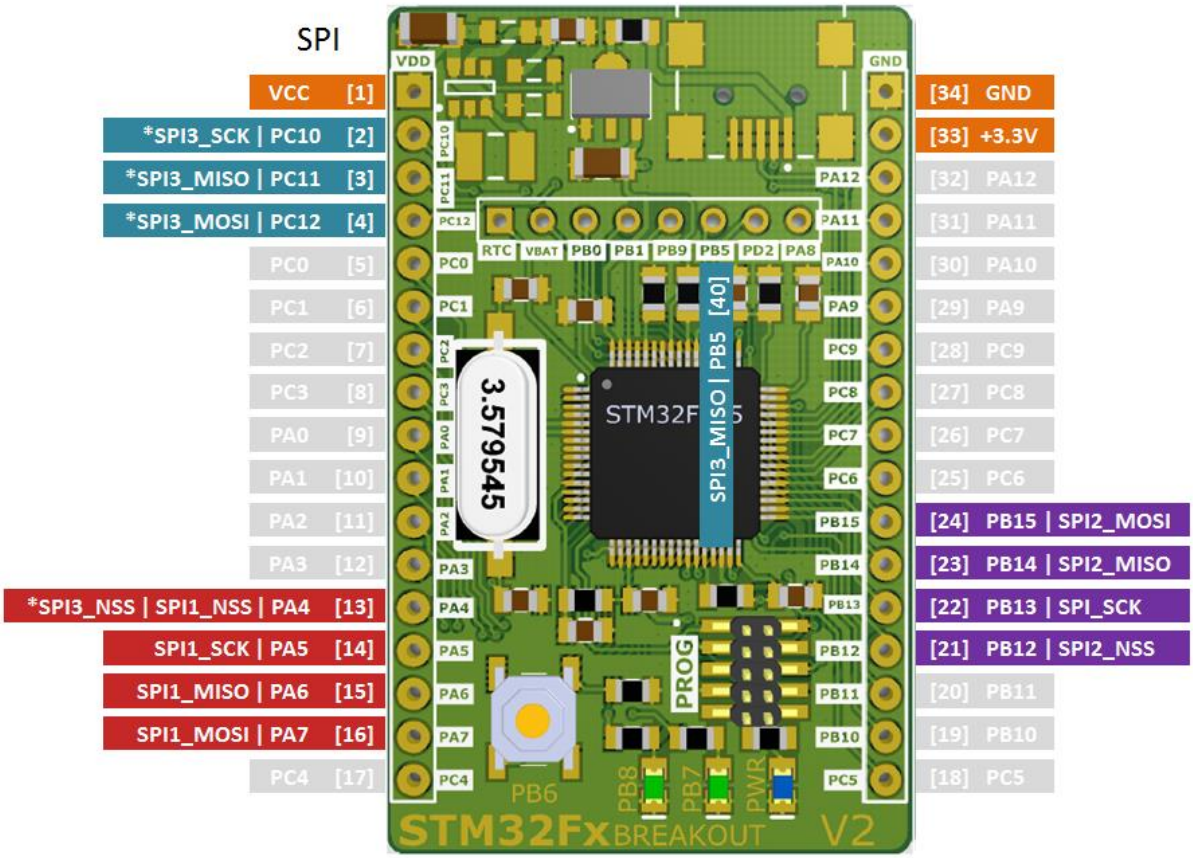
3. Hardwareschnittstellen

Hier findest du eine Übersicht, über alle Anschlüsse der verschiedenen Hardwareschnittstellen wie z.B. SPI, I^2C etc.

Beachte: Diese Schnittstellen sind eventuell nicht für alle Mikrocontroller im LQFP64 Gehäuse verfügbar!

Etwas weiter unten in dieser Anleitung, befinden sich Codesnippets zum einfachen ansteuern dieser Schnittstellen.

3.1 SPI-Schnittstellen

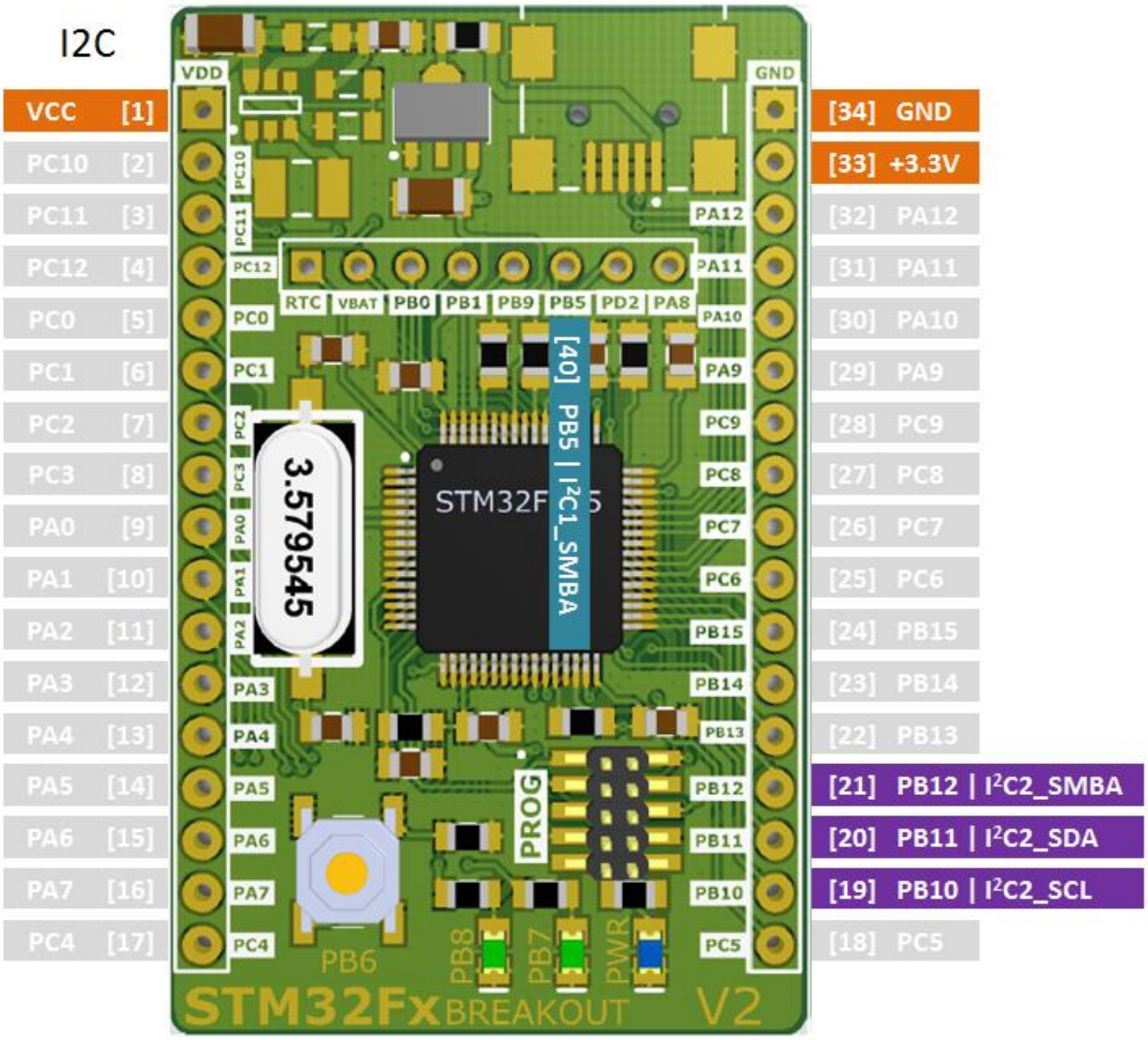


Wichtig! Die mit einem Sternchen versehenen Schnittstellenports, sind erst nach einem Remap der entsprechenden Pins verwendbar.

Wie dies funktioniert, wird weiter unten beschrieben.

Kann man auf den ChipSelect (NSS) von SPI3 oder SPI1 verzichten, stehen insgesamt 3 SPI-Schnittstellen zur Verfügung.

3.2 I²C-Schnittstellen



Es gibt leider lediglich eine I²C Schnittstelle auf dem Breakoutboard.

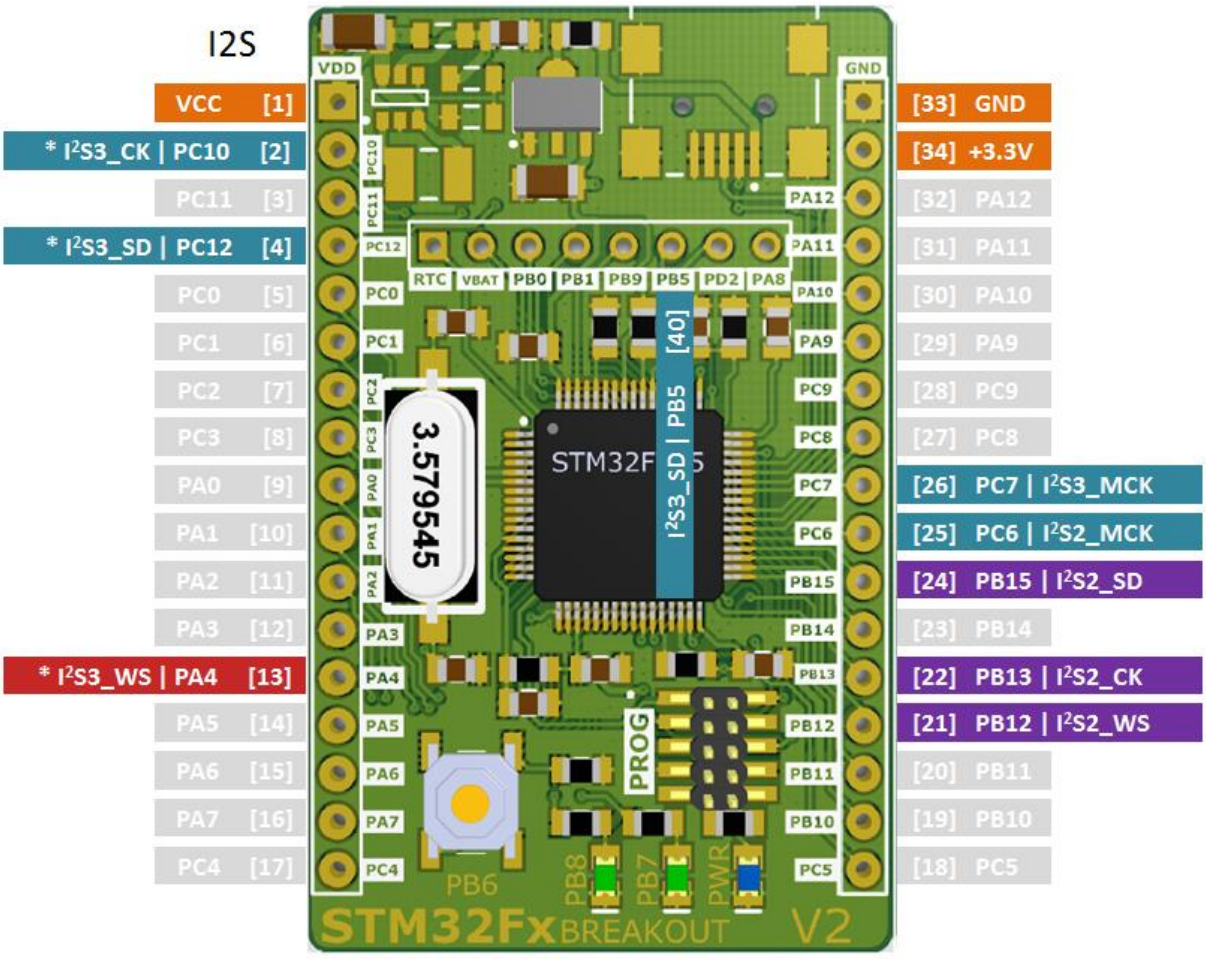
Ein Remap auf andere Pins ist nicht möglich.

3.3 I²S-Schnittstellen

I²S Schnittstellen, werden vorwiegend zur Übertragung von Audiodaten verwendet.

Viele Audio-DAC's wie beispielsweise der MAX9850 verwendet diese Schnittstellen.

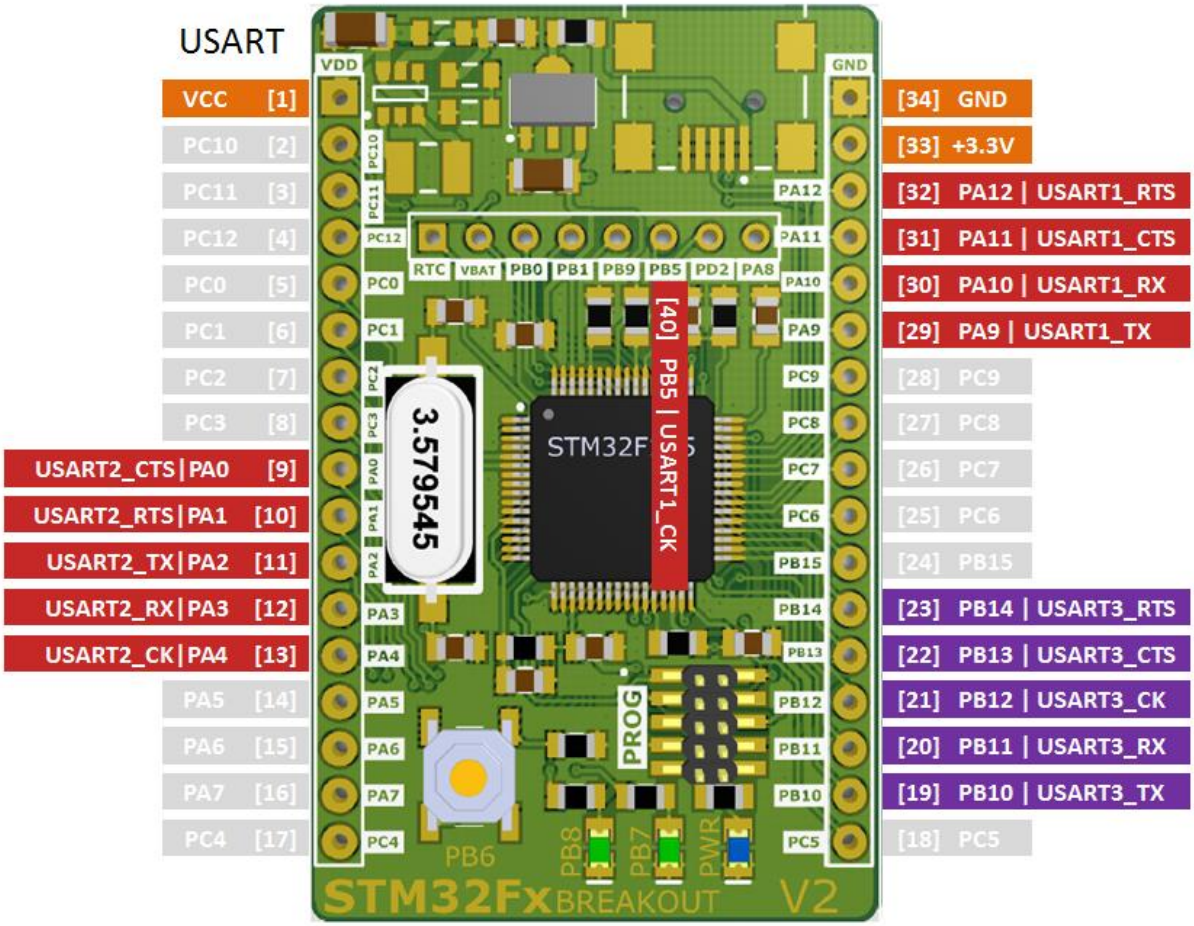
Das wichtigste bei I²S ist die Bereitstellung der richtigen Clocks im richtigen Verhältnis zu den übertragenden Daten. Hier helfen die internen PLL's der STM-Controller!



Die mit einem Sternchen versehenen Ports, sind erst nach einem Remap verfügbar!

Es sind bis zu zwei vollwertige I²S Schnittstellen möglich!

3.4 USART-Schnittstellen



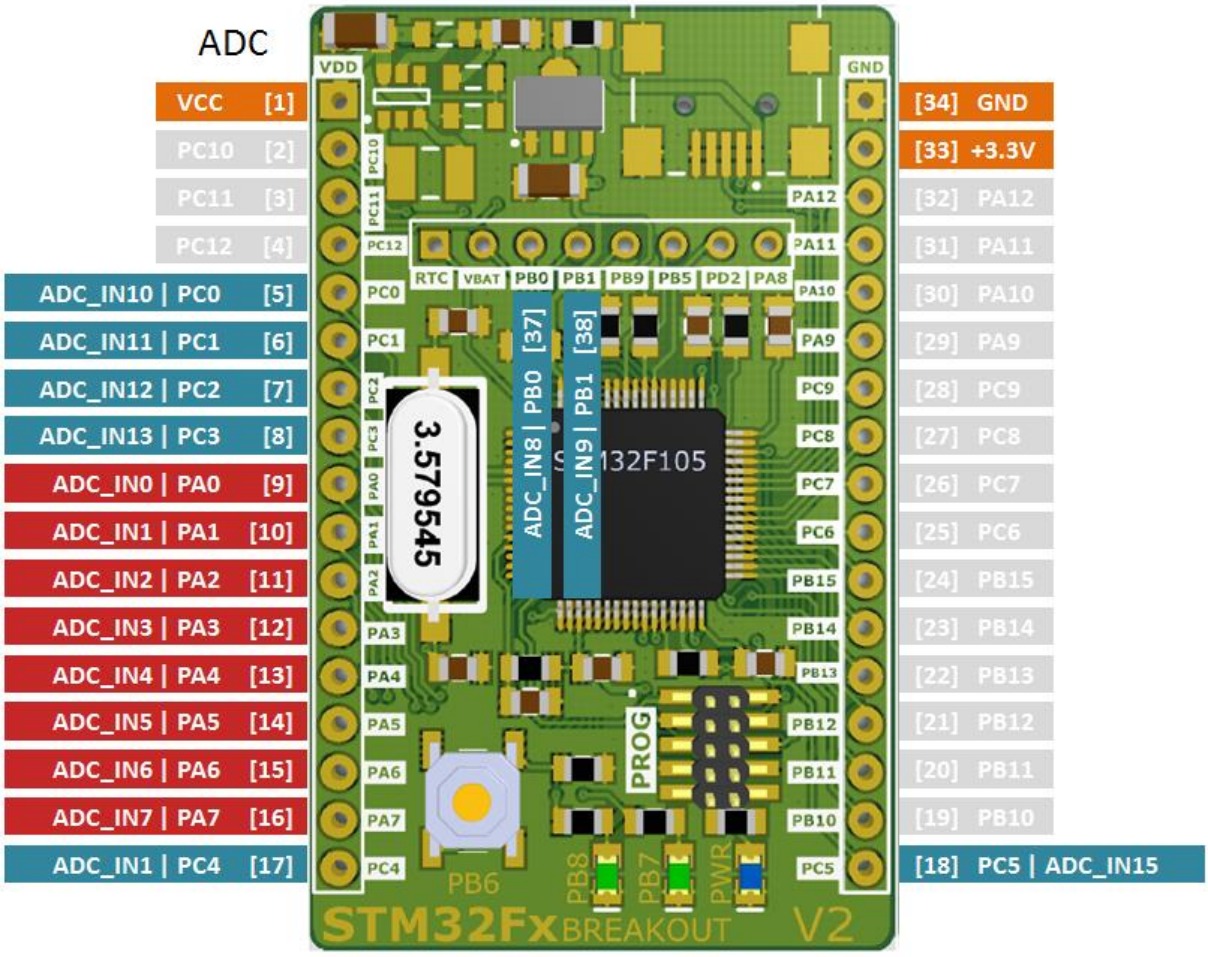
Ein remap ist für keine USART-Schnittstelle notwendig.

Insgesamt sind 3 vollwertige USART vorhanden.
Im Vergleich zum Breakout V1 ist nun auch für USART1 das CK Signal herausgeführt!

3.5 Analogeingänge

Im STM32F105RB befinden sich zwei 12bit AD-Wandler
Dieser hat eine maximale Sample-Rate von 2-MSPS im interleaved Modus.

Wandlungsbereich: 0-3.6V
Sample and Hold Möglichkeit.
Inkl. Temperatursensor zur Kompensation des Temperaturdrifts.



Insgesamt gibt es im Mikrocontroller 16 ADC Kanäle verteilt auf 2 Hardware-ADC's

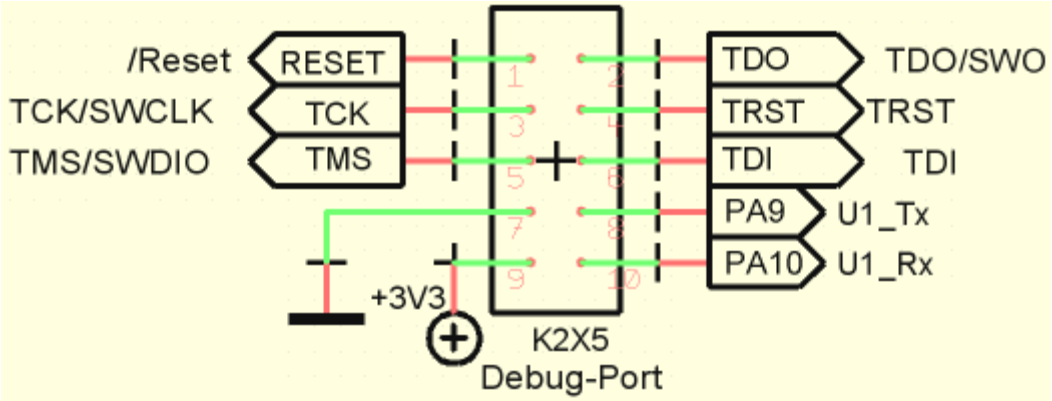
3.6 Mini-JTAG/SWD

Die auf diesem Board verwendete JTAG-Pinbelegung entspricht nicht dem Standard mit 20-Pol Stiftleisten.

Diese neue Pinbelegung mit nur 10 Pins, stammt von der mikrocontroller.net Webseite. (www.mikrocontroller.net/articles/JTAG)

Entworfen hat die neue Schnittstelle *mmvisual* (http://www.mmvisual.de/)

Nachfolgend nun die Pinbelegung.



(Bild von www.mikrocontroller.net/articles/JTAG)

Beim Breakoutboard, ist leider kein UART auf dem JTAG vorhanden. Insofern entspricht der JTAG nicht zu 100% dem Proprietären Standard.

Zum Verbinden, wird eine 1.27mm Stiftleiste 2x5 benötigt. Ein Passendes Modell, lässt sich bei RS-Online unter der Nr: 681-0692 finden.

Alternativ gibt es bei Reichelt eine 2x10 Stiftleiste.
Bestellnummer: SL 2X10G SMD1,27

4. Codefragmente

In diesem Abschnitt, finden sich diverse Codefragmente für die verschiedensten Breakoutboard-funktionen.

4.1 Minimalclock

Mithilfe des nachfolgenden Codefragments, wird das Breakoutboard mit einer ??????????

```
//Select HSI as PLL-Source divided by 2 = 4MHz multipl. 9 = 36MHz
RCC_PLLConfig(RCC_PLLSource_HSI_Div2,RCC_PLLMul_9);

//Activate PLL
RCC_PLLCmd(ENABLE);

//Wait until PLL is Ready
while (RCC_GetFlagStatus(RCC_FLAG_PLLRDY) == RESET);

//Set the PLLCLOCK as Source for the SYSCLK
RCC_SYSCLKConfig(RCC_SYSCLKSource_PLLCLK);
```

4.2 USART1 Codefragment für Copy&Paste

Nachfolgender Code ist für die Initialisierung des USART1.

```
/* Private variables ---------------------------------------------------------*/
USART_InitTypeDef USART_InitStructure;
GPIO_InitTypeDef GPIO_InitStructure;
NVIC_InitTypeDef NVIC_InitStructure;


/* Configure the NVIC Preemption Priority Bits */
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);

/* Enable the USARTy Interrupt */
NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
NVIC_Init(&NVIC_InitStructure);

/* Configure USART1 Rx as input floating */
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10; //USART1 Rx Pin
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
GPIO_Init(GPIOA, &GPIO_InitStructure); //USART1 GPIO Port. In this Case Port A

/* Configure USART1 Tx as alternate function push-pull */
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;  //USART1 Tx Pin
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
GPIO_Init(GPIOA, &GPIO_InitStructure); //USART1 GPIO Port. In this Case Port A


/* Enable USART1 Clock */
RCC_APB1PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);

USART_InitStructure.USART_BaudRate = 9600;
USART_InitStructure.USART_WordLength = USART_WordLength_8b;
USART_InitStructure.USART_StopBits = USART_StopBits_1;
USART_InitStructure.USART_Parity = USART_Parity_No;
USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;

/* Configure USART1 */
USART_Init(USART1, &USART_InitStructure);

/* Enable USART1 Receive and Transmit interrupts */
USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);
USART_ITConfig(USART1, USART_IT_TXE, ENABLE);

/* Enable the USART1 */
USART_Cmd(USART1, ENABLE);
```

Hier noch ein wichtiges Codefragment. Jenes für den Interrupthandler.

```
// USART Interrupt Handler

void USART1_IRQHandler(void)
{
  if(USART_GetITStatus(USART1, USART_IT_RXNE) != RESET)
  {
    /* Read one byte from the receive data register */
    USART_ReceiveData(USART1);   //This Function returns the Received Byte!
  }

  if(USART_GetITStatus(USART1, USART_IT_TXE) != RESET)
  {
    /* Write one byte to the transmit data register */
    USART_SendData(USART1, 0xAA);      //Send some Dummy Data (0xAA)
  }
}
```

Der Code ist voll funktionsfähig und kann direkt mittels Copy&Paste verwendet werden.

Der erste Teil, muss in die *main()* Funktion rein. Der zweite, ausserhalb.

Alle Codebeispiele wurden mit CooCox getestet!

5 – Aufbau

Das Breakoutboard lässt sich relativ leicht von Hand zusammenlöten.

Die meisten Bauteile um das Board aufzubauen, wurden hier in einem Reichelt-Warenkorb zusammengetragen.

https://secure.reichelt.de/index.html?&ACTION=20&AWKID=968416&PROVID=2084

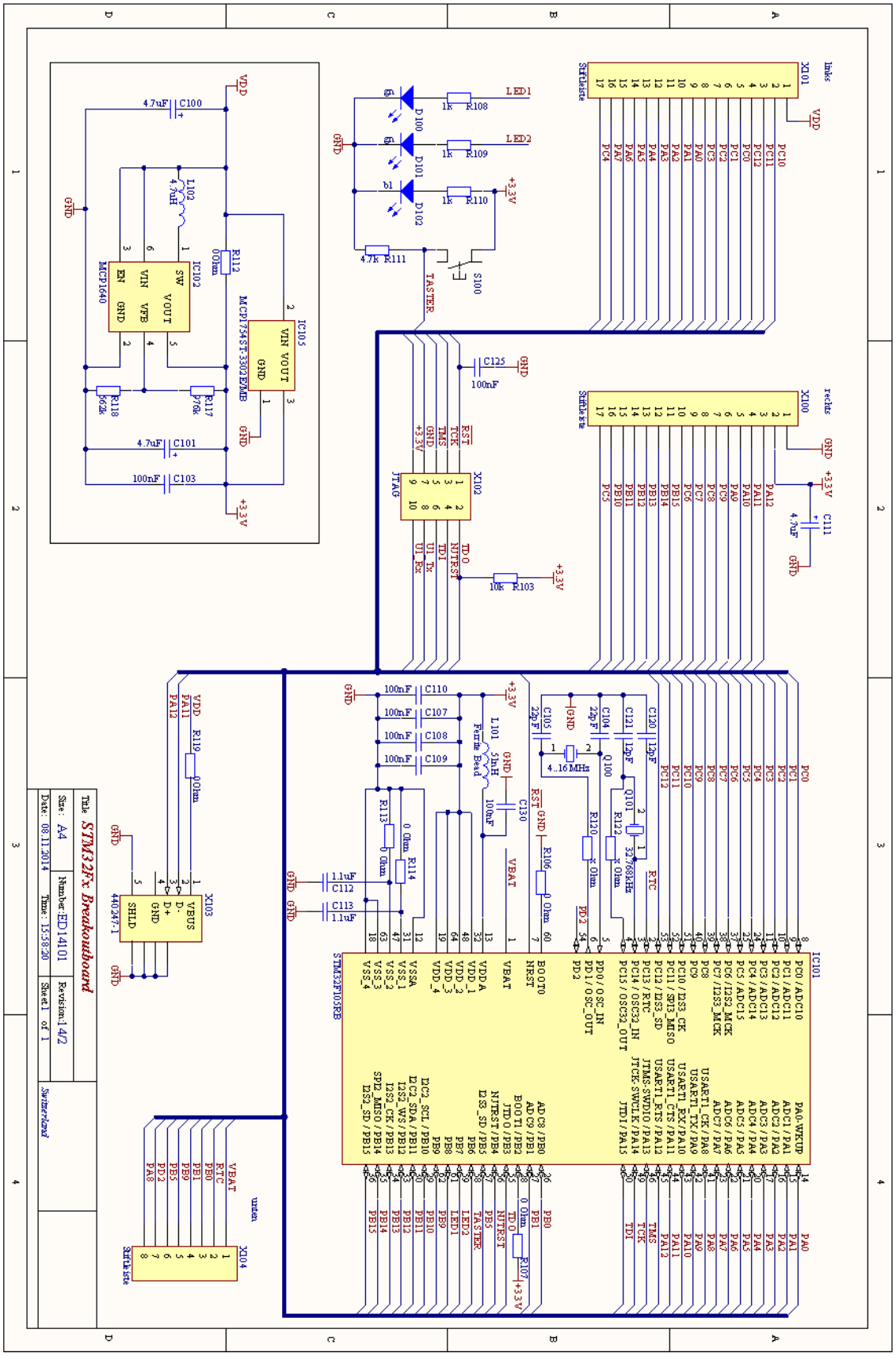
Es fehlen noch die Wiederstände für den Spannungsteiler des Stepup Converters. (R117 & R118)

Zudem muss man sich zuvor entscheiden, ob man das Board für die F1 oder die F2/F4 Serie verwenden möchte.

Davon hängt ab, ob R113/R114 bestückt werden muss oder für die Serie F2/F4 C112/C113.

5.1 – Schema

Das nachfolgende Schema entspricht der HW-Version 14/2



5.2 – Bestückungsplan