USB-Blaster

Es soll ein Programmieradapter für FPGAs von Altera(USB-Blaster) entstehen.

Vorgeschichte

Damals, vor vielen Jahren, hat sich Sebastian DL3YC mit dem Selbstbau von Logic Analyzern beschäftigt. Das war die Zeit als Nachbau-Versuche vom Salea Logic begannen erfolgreich zu werden. Dadurch bestärkt und auch durch die berufliche Auseinandersetzung mit dem USB-Schaltkreis FX2 von Cypress, entstand der Wunsch nach einem FX2-basierten Logic Analyzer, der die hervorragende Software von Salea nutzen kann. Der Trick ist dabei, dass die Firmware zur Laufzeit auf das Gerät geladen wird und man nur dafür sorgen musste, dass der Schaltkreis sich mit der richtigen Produkt- und Hersteller-ID per USB meldet. Dazu schreibt man sie in einen I²C-EEPROM, das der FX2(auch liebevoll „Zypresse“ genannt) beim Power-Up ausliest und sich mit den Daten versucht zu enumerieren.

Dies war alles im Jahre 2010, es entstanden selbst gebaute Leiterplatten von Sebastian und auch Winni DL2AWT(Datecode auf dem PCB: 5010, also KW50 2010). Vorallem die Leiterplatte von Winni hatte Potenzial, da man mit dieser auch einen schnellen AD-Wandler angebunden hat. Alle Leiterplatten hatten leider gemein, dass sie Probleme beim Enumerieren als Hi-Speed-Gerät haben. Dieses Problem zeigte sich auf sehr unterschiedliche und schlecht reproduzietrbare Weise. „Manchmal geht's“ ist für einen Logic Analyzer, den man vorallem in problematischen Situationen einsetzen will, keine Option. So vergingen 2 Jahre bis man sich noch einmal aufrappelte und eine Kopie von scheinbar funktionierenden Klonen aus dem Internetz in China fertigen lies(Datecode „0612“, also Juni 2012). Aber auch diese Leiterplatten zeigten die gleichen Probleme, die sich auch durch intensive Analyse nicht beheben wollten. Enttäuscht wurde das Projekt beendet.

Jetzt liegen also noch die Leiterplatten und vorallem auch die teuren USB-Schaltkreise rum. Um den leider fehlgeschlagenen Projekt noch einen Sinn zu geben, werden auf Grundlage dieser Leiterplatten USB-Blaster hergestellt. Dazu wird die Leiterplatte für eine übliche JTAG-Verbindung umgebaut. Als Software läuft auf der Zypresse eine freie Firmware, bei der nur das Pinout angepasst werden muss.

LOGIC

Zur Referenz die initiale Leiterplatte, wie sie als Logic-Analyzer angedacht war:
Schaltplan
Layout
EEPROM-Inhalt
Anleitung zur Inbetriebnahme

Umbau der Leiterplatte

Es wird die Leiterplatte größtenteils laut Schaltplan von LOGIC aufgebaut. Es wird die Minimalbeschaltung beschrieben und die zu bestückenden BE in der empfohlenen Aufbaureihenfolge

aufgelistet:

1. X1, U1, C13, C14, C12, C21, R19
    - R19 mit 1M bestücken
    - Spannung über C12 soll 3,3V betragen
2. U2, U3, C1-C11, Q1, C17, C18, R8, R1, R2, R4, R6
    - R4 und R5 mit 4k7 bestücken
    - R8 mit 1M bestücken
    - C18 mit 12p bestücken
    - Nach Anstecken von USB wird ein neues Gerät mit VID = 0x04B4 und PID = 0x8613 erkannt
3. RN1..4, D5, D7, D4, D8
4. Datenpins 1,2,4,5 von U6 Eingang zu Ausgang verbinden
5. H2 Anode mit C0 der Zypresse verbinden

Software-Anpassung

Die Software wurde hinsichtlich der Zuordnung MCU-Pin und JTAG-Pin angepasst:

Pin	**JTAG**	**Steckverbinder**
PB4 Out	TCLK	1
PB3 In	TDO	3
PB2 Out	TMS	5
PB0 Out	TDI	9
PC0 Out	JTAG_EN	-

Die Firmware unterstützt nicht die Modi Active Serial und Passive Serial.

EEPROM-Programmierung

Zuerst lädt man sich fx2fw aus meinem Github und kompiliert es:

```
# git clone https://github.com/dl3yc/fx2fw-sdcc.git
# cd fx2fw-sdcc
# make
```

Danach wird das Gerät angeschlossen und die Bus- und Gerätenummer ermittelt:

```
# lsusb
Bus 004 Device 006: ID 04b4:8613 Cypress Semiconductor Corp. CY7C68013 EZ-
USB FX2 USB 2.0 Development Kit
```

In diesem Beispiel ist die Gerätenummer 006 und der Bus 004.
Mit diesen Angaben wird die kompilierte Firmware im Hex-Format auf das Gerät bespielt. Direkt danach enumeriert sich das Gerät als USBBlaster.

```
# fxload -I usbjtag.hex -D /dev/bus/usb/004/006 -t fx2lp
```

```
# dmesg
[75309.252135] usb 1-5: new high-speed USB device number 13 using ehci-pci
[75309.384450] usb 1-5: config 1 interface 0 altsetting 0 bulk endpoint 0x81
has invalid maxpacket 64
[75309.384463] usb 1-5: config 1 interface 0 altsetting 0 bulk endpoint 0x2
has invalid maxpacket 64
[75309.384937] usb 1-5: New USB device found, idVendor=09fb, idProduct=6001
[75309.384944] usb 1-5: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[75309.384951] usb 1-5: Product: USB-Blaster
[75309.384957] usb 1-5: Manufacturer: EPFL
[75309.384963] usb 1-5: SerialNumber: 00000000
```

Nun ist das Gerät bereit als USBBlaster zu fungieren
Zu beachten: Dieser Prozess muss bei jedem erneuten Einstecken des USBBlasters gemacht werden. Um die Firmware dauerhaft in das EEPROM zu schreiben, fehlt noch ein 2nd Stage Bootloader, *TODO*.

Inbetriebnahme

From:
https://loetlabor-jena.de/ - **Lötlabor Jena**

Permanent link:
https://loetlabor-jena.de/doku.php?id=projekte:usbblaster:start&rev=1423952876

Last update: **2015/02/14 22:27**