Man könnte vielleicht Jumper oder so etwas verwenden um eventuell zwischen Bitstreams zu tauschen.

Denke auch, dass das praktisch sein könnte.

Können simple Mechanismen verwenden um Bitstreams per COM in den Arduino zu Pumpen und von dort dann in das FPGA?

Ich habe ein Python Skript und einen zugehörigen Arduino Sketch mit welchen man via COM Bitstreams auf den FPGA schreiben können sollte (heißt im Repo: *„Ard_Host_to_FPGA_Adapter“)*.

Alternativ habe ich auch ein kleines Python Skript geschrieben mit dem man ein Header-File mit dem Bitstream erzeugen kann und das dann in den FLASH schreiben kann. Beim Start den µC wird das dann in den FPGA programmiert (heißt im Repo: *„Ard_FlashBitstream“*).

Beides sollte man mal ausprobieren 😊

Wieviel Platz haben wir?
Wie groß darf die Konfiguration für den RAM und wie groß für den Flash sein?

Lt. Arduino Nano Schachtel total:

SRAM: 2KB

FLASH: 32KB

Was der Compiler/Linker zum Platzverbrauch sagt:

- Wenn ich das COM-Beispiel kompiliere, womit man Daten via Python vom PC an den Arduino senden kann und dieser diese dann gleich auf den FPGA programmiert (heißt im Repo: *„Ard_Host_to_FPGA_Adapter“):*

  ```
  Der Sketch verwendet 1890 Bytes (6%) des Programmspeicherplatzes. Das Maximum sind
  30720 Bytes.
  Globale Variablen verwenden 190 Bytes (9%) des dynamischen Speichers, 1858 Bytes für
  lokale Variablen verbleiben. Das Maximum sind 2048 Bytes.
  ```

- Wenn ich das FLASH Beispiel kompiliere (mit **4bit_counter Bitstream**). Hier kann man beim Kompilieren einen Bitstream als Headerfile mit-includen und dieser wird dann programmiert wird wenn der Arduino startet (heißt im Repo: *„Ard_FlashBitstream“*):

  ```
  Der Sketch verwendet 9010 Bytes (29%) des Programmspeicherplatzes. Das Maximum sind
  30720 Bytes.
  Globale Variablen verwenden 220 Bytes (10%) des dynamischen Speichers, 1828 Bytes für
  lokale Variablen verbleiben. Das Maximum sind 2048 Bytes.
  ```

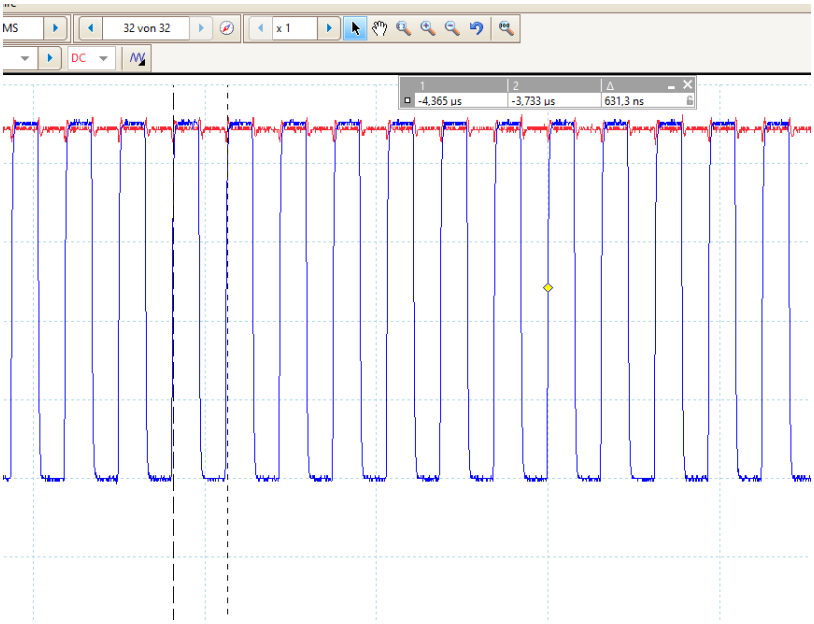
- Wenn ich ein minimales Beispiel kompiliere. Hier wird einfach nur periodisch immer wieder dasselbe Byte auf den FPGA programmiert (heißt im Repo: *„ArduinoSketch“*):

  ```
  Der Sketch verwendet 664 Bytes (2%) des Programmspeicherplatzes. Das Maximum sind
  30720 Bytes.
  Globale Variablen verwenden 9 Bytes (0%) des dynamischen Speichers, 2039 Bytes für
  lokale Variablen verbleiben. Das Maximum sind 2048 Bytes.
  ```

SRAM kann man wohl auch ganz elegant mit der *freeMemory()* Funktion messen aber bisher sind die Arduino Sketches nicht sehr Memory hungrig (https://learn.adafruit.com/memories-of-an-arduino/measuring-free-memory)

Wenn der Bitstream drin ist, wie schnell ist der Clock?

(Duty-Cycle braucht nicht symmetrisch zu sein)



1 Clock Cycle dauert ca. 631ns -> D.h. ca. 1,584MHz

Ganz alle Interrupts abklemmen könnte vielleicht ein Problem sein, wenn ein neuer Konfigurationsvorgang ansteht, aber da wird Dir schon was einfallen.

Ich lasse sie stand jetzt einfach an. Ich sehe bisher keinen Grund warum ein auftretender Interrupt ein Problem sein sollte bei der Übertragung von Daten an den FPGA. Aber sonderlich elegant wirkt es auch nicht.