Kurzdokumention zu den Signalen des DEMOCOM

Im Folgenden sollen die Signale des Prozessor-Designs kurz in der Reihenfolge besprochen werden, in der sie in der Parameterliste auftreten, und zu den wichtigen Signalen einige Hinweise bezüglich ihrer praktischen Verwendung gegeben werden, die allerdings ohne eine gewisse erste praktische Erfahrung im Umgang mit der Beschreibungssprache AHDL und den Simulationstools leider etwas kryptisch bleiben müssen.

reset (Eingang - wichtig)

Low-aktives Reset-Signal, das es ermöglicht, den Prozessor in einen definierten Anfangszustand zu bringen. Das Signal sollte an alle Register und insbesondere den Automaten für das Steuerwerk geführt werden.

Hinweis:
Man beachte, daß die normalerweise in AHDL verwendeten DFFE-Flipflops einen low-aktiven Reset-Eingang CLRN haben, während der Reset-Eingang für die Beschreibung von Automaten (machine with states) high-aktiv ist. Das heißt, in AHDL müßte es z.B. Statements folgender Form geben:

```
VARIABLE     REG0[15..0]:   DFFE;
             ............
             STEUERWERK:    MACHINE WITH STATES
                  (init, read0, .....);
             .......
BEGIN
REG0[15..0].CLRN= RESET;
......
STEUERWERK.RESET= NOT RESET;      % !!!!!!!! %
......
END;
```

Man beachte, daß das Steuerwerk auf diese Weise in ersten Zustand in der Liste der Zustände (hier also **init**) gebracht werden kann. Dieser Zustand sollte einzig und allein der Initialisierung dienen und damit insbesondere danach nicht mehr angesprungen werden.

clk1, clk2 (Eingänge - wichtig)

Komplementäre Takte, die vom Hilfs-FPGA zur Verfügung gestellt werden. Üblicherweise nimmt man eins der beiden Signale als Takt für das Steuerwerk und das andere für das Operationswerk. In AHDL wird es also Statements geben wie:

```
VARIABLE     REG0[15..0]:   DFFE;
             ............
             STEUERWERK:    MACHINE WITH STATES
                  (init, read0, .....);
             .......
BEGIN
REG0[15..0].CLK= CLK1;
......
STEUERWERK.CLK=  CLK2;
......
END;
```

Hinweis für die Simulation:
Es ist günstig, sich automatisch einen dauernd laufenden Takt erzeugen zu lassen. In der Datei DEMO.STI, die sich als Beispiel auf dem Directory befindet, ist ein solcher Prozess (clkgen) angegeben.

abus_in[19..0] (Eingänge - unwichtig)

Für (das unten erklärte Signal) **abus_write= 1** ist **abus_in= abus_out**. Das ist normalerweise völlig bedeutungslos. Das Signal kann damit ignoriert werden.

dbus_in[15..0] (Eingänge - wichtig)

Wie der Adressbus wird der Datenbus im Rahmendesign ALTERA1500.TFF in zwei Teile aufgespalten; einen **dbus_in**, auf dem Daten in den Prozessor hineingehen, und einen **dbus_out**, auf dem Daten aus dem Prozessor herauskommen. Alle Daten, die z.B. vom RAM oder aus der Peripherie (PIO, SIO) kommen, liegen also auf dem **dbus_in**. Wenn man den Prozessor in das vorgegebene Rahmendesign ALTERA1500.TTF einbindet, ist für (das unten erklärte Signal) **dbus_write= 1** über dort eingebaute Logik **dbus_in**

= **dbus_out**. Sollte der Prozessor diese Möglichkeit nutzen, sind bei der Simulation besondere Maßnahmen notwendig. In der Beispieldatei DEMO.STI findet sich ein entsprechender VHDL-Prozess.

halt_n (Eingang - wichtig)

Low-aktives Signal, um den Prozessor vom Hilfs-FPGA aus anzuhalten. Dies Signal ist essentiell wichtig für die Kommunikation auf der Mikrorechner-Platine und muß unbedingt sauber bedient werden! Das heißt, es muß sichergestellt sein, daß das Hilfs-FPGA zumindest bei jedem Lesen eines neuen Befehls die Möglichkeit hat, den Prozessor durch Setzen von **halt_n= 0** anzuhalten. Der Prozessor muß dann (und nur dann) unbedingt das Ausgangssignal **halt_n_ack= 0** setzen und er muß auch die Busse über die Signale **dbus_write= 0** und **abus_write= 0** freigeben, ebenso muß dann das Ausgangssignal **output_ena_n= 1** sein, damit nicht das RAM den Datenbus treibt.

Hinweis:
Das oben gesagte gilt in jedem Fall; also auch dann, wenn sich der Prozessor möglicherweise in einem Zustand für einen ungültigen Operations-Code befindet, oder dann, wenn er in einem besonderen Zustand auf einen Interrupt wartet. Ohne eine richtige Behandlung des Signals **halt_n** läuft auf der Platine so gut wie nichts mehr!

reg_req_n (Eingang - kann genutzt werden)

Bietet die Möglichkeit, sich Register des Prozessors anzusehen, sofern dies implementiert ist. Für **reg_req_n= 0** wartet das Hilfs-FPGA, bis der Prozessor **reg_ack_n= 0** setzt und liest dann den Daten- und Adressbus aus, um die Werte an den Rechner (PC) weiterzuleiten. Das heißt, der Prozessor kann dann in Abhängigkeit der Eingänge **req_select** Werte zu Debugzwecken auf **dbus_out** oder **abus_out** legen und sie durch Setzen von **dbus_write= 1** bzw. **abus_write= 1** durchschalten. Es ist klar, daß dann nicht gleichzeitig **output_ena_n= 0** sein darf.

Hinweis:
Es ist nur sinnvoll, sich das Signal **reg_req_n** anzusehen, wenn der Prozessor über das Signal **halt_n** angehalten worden ist, weil nur dann das Hilfs-FPGA überhaupt diese Debugmöglichkeit zuläßt.

reg_select[3..0] (Eingänge - können benutzt werden)

Bietet die Möglichkeit, Register auszuwählen (siehe Signal **reg_req_n**), die man sich zu Debugzwecken ansehen möchte. Wenn dies nicht benutzt werden soll, können die Eingänge ignoriert werden.

irq_n[3..0] (Eingänge - können benutzt werden)

Lowaktive Interrupt-Request-Leitungen aus der Peripherie, d.h. ein Gerät möchte bedient werden. Im Moment ist **irq_n[0]** der PIO, **irq_n[1]** der SIO und die beiden anderen Leitungen möglichen anderen Geräten zugewiesen. Wenn nicht mit Interrupts gearbeitet wird, können die Signale ignoriert werden; ansonsten sollte eine Rücksprache mit den Betreuern erfolgen.

micro_d[39..0] (Eingänge - können benutzt werden)

Daten aus dem Mikroprogrammspeicher. Wenn der Prozessor nicht mikroprogrammiert ist, können die Eingänge ignoriert werden.

bus_grant_n (Eingang - unwichtig)

Kann ignoriert werden. Ursprünglich gedacht, um einem externen Busmaster die Gelegenheit zum Zugriff auf die Busse zu geben (z.B. DMA). Da es so etwas aber (noch) nicht gibt, unwichtig.

jumper[7..0] (Eingänge - unwichtig)

Eingänge eines Jumperfeldes, die normalerweise ignoriert werden können.

abus_write (Ausgang - wichtig)

Schaltet im Rahmendesign die internen Adressleitungen **abus_out** auf den externen Adressbus. Es kann normalerweise außer im Zustand, der das Signal **halt_n** behandelt, einfach **abus_write= 1** gesetzt werden.

abus_out[19..0] (Ausgänge - wichtig)

Interne Adressleitungen, die im Rahmendesign über **abus_write= 1** auf den externen Adressbus geschaltet werden.

astrobe_n (Ausgang - unwichtig)

Bietet die Möglichkeit, ein asynchrones Busprotokoll zu fahren. Wenn das nicht genutzt wird, was wohl der Normalfall ist, kann man global **astrobe_n= 1** setzten.

dbus_write (Ausgang - wichtig)

Für **dbus_write= 1** werden (in Rahmendesign) die internen Datenleitungen **dbus_out** auf den externen Datenbus und den **dbus_in** geschaltet. Der Prozessor hat also dafür zu sorgen, daß dies geschieht, wenn er z.B. in den Speicher schreiben will. Wenn er dagegen aus dem Speicher liest, muß unbedingt **dbus_write= 0** sein, damit Speicher und Prozessor nicht gleichzeitig den **dbus_in** benutzen.

Hinweis:
Sollte der Prozessor die Möglichkeit **dbus_in= dbus_out** für **dbus_write= 1** nutzen, müssen bei der Simulation entsprechende Maßnahmen ergriffen werden. In der Beispieldatei DEMO.STI findet sich ein entsprechender Prozess.

dbus_out[15..0] (Ausgänge - wichtig)

Interne Daten, die aus dem Prozessor herausgehen. Man beachte das zu **dbus_write** Gesagte.

dstrobe_n (Ausgang - unwichtig)

Bietet die Möglichkeit, ein asynchrones Busprotokoll zu fahren. Wenn das nicht genutzt wird, was wohl der Normalfall ist, kann man global **dstrobe_n= 1** setzten.

halt_ack_n (Ausgang - wichtig)

Essentiell wichtiges Signal für die Kommunikation mit dem Hilfs-FPGA. Man beachte unbedingt das oben zum Eingang **halt_n** Gesagte.

reg_ack_n (Ausgang - wichtig)

Man beachte das zum Eingang **reg_req_n** Gesagte. Wenn diese Debugmöglichkeit nicht genutzt wird, sollte **reg_ack_n= reg_req_n** gesetzt werden.

irq_ack_n[3..2] (Ausgänge - unwichtig)

Vorgesehen für ein lowaktives Interrupt-Acknowledge-Signal für spätere Erweiterungen der Platine. Sollte man global auf **1** setzen.

output_ena_n (Ausgang - wichtig)

Für **output_ena_n= 0** wird aus dem Speicher oder den Registern der Peripherie (PIO, SIO) gelesen. Das macht nur Sinn, wenn gleichzeitig eine gültige Adresse auf **abus_out** liegt, **abus_write= 1** und **dbus_write= 0** ist. Wenn nicht gelesen werden soll, muß man <u>unbedingt</u> **output_ena_n= 1** setzen. Ebenso ist in jedem Fall gleichzeitiges **output_ena_n= 0** und **dbus_write= 1** bzw. **output_ena_n= 0** und **write_ena_n= 0** zu vermeiden!

Hinweis:
1.) Normalerweise läßt sich das Lesen in einem Prozessortakt durchführen. Man beachte aber, daß für das Auslesen der Register für die Tastatur das Signal **output_ena_n** <u>zwei</u> Takte lang aktiv (**0**) sein muß.
2.) Ein (primitives) Speichermodell für die Simulation findet sich in der Datei DEMO.STI.

write_ena_n (Ausgang - wichtig)

Für **write_ena_n= 0** wird in den Speicher oder die Register der Peripherie (PIO, SIO) geschrieben. Das macht nur Sinn, wenn gleichzeitig eine gültige Adresse auf **abus_out** liegt, **abus_write= 1** und **dbus_write= 1** ist. Wenn nicht geschrieben werden soll, muß man <u>unbedingt</u> **write_ena_n= 1** setzen. Ebenso ist in jedem Fall gleichzeitiges **output_ena_n= 0** und **write_ena_n= 0** zu vermeiden!

Hinweis:
Es sei hier daran erinnert, daß eine Schreiboperation (mindestens) drei Takte benötigt.
1. Takt: Adresse und Daten auf **abus_out, dbus_out**, **abus_write= 1, dbus_write= 1, write_ena_n= 1**.
2. Takt: Wie erster Takt, aber **write_ena_n= 0**
3. Takt: wie erster Takt.

micro_a[14..0] (Ausgänge - können benutzt werden)

Für mikroprogrammgesteuerte Prozessoren die aktuelle Adresse in Mikroprogrammspeicher. Andere Prozessoren sollten alle Leitungen mit **0** belegen.

led[15..0] (Ausgänge - können benutzt werden)

Leuchtdioden, die vom Prozessor aus angesteuert werden können, um sich z.B. den aktuellen Zustand oder das Auftreten eines ungültigen Operationscodes anzeigen zu lassen. Können auch alle mit **0** belegt werden.

bus_ack_n (Ausgang - wichtig)

Ist im Grunde bedeutungslos, schaltet aber zusammen mit **abus_write** den **abus_out** im Rahmendesign auf den externen Adressbus. Darum sollte global **bus_ack_n= 1** gesetzt werden.