Beginn : 15:30 Uhr
Ende : 17:15 Uhr
Raum : C2.227

1 Allgemeines

Es wurde bemängelt, daß die beiden Rechner *pollux* und *prometheus* mit jeweils 128 MB ständig von anderen Usern benutzt werden und so z.B. die speicherintensive Synthese größerer Schaltungen viel zu lange läuft.
Als Lösung zu diesem Problem gab es 2 Vorschläge :

- Einen Rechner komplett für andere User sperren
- Die anderen Benutzer bitten, ihre speicherintensiven Prozesse nachts laufen zu lassen

Desweiteren wurde besprochen, daß es unumgänglich ist, später beide Boards auf dem Simulator zu benutzen. Das Problem, was dabei auftreten wird, ist die Verbindung zwischen den einzelnen Boards, da uns nur eine bestimmte Anzahl von Verbindungsleitungen zur Verfügung steht (512 Pins). Erste Schätzungen zeigten jedoch, das unter Benutzung des Auto-Partionierers diese Zahl überschritten wird und uns so keine eigentlichen I/O-Leitungen mehr zur Verfügung stehen werden.
Bis zur nächsten Woche wollen wir uns dann überlegen, ob man die gesamte Schaltung so auf die beiden Boards verteilen kann, daß jeweils 12 Bit auf dem einen und 12 Bit auf dem anderen Board integriert werden. D.h., daß man nicht die Schaltung in zwei Teile aufteilt, sondern jedes einzelne Modul, um so einige Verbindungsleitungen zu sparen.

2 Interface–Gruppe

Die Minimalkonfiguration der Interface–Gruppe ist soweit fertig und konnte bereits komplett in INCA eingelesen werden. Probleme gibt es allerdings noch beim automatischen Partionieren mit den Tristates. Eine Möglichkeit wäre, die einzelnen Gatter per Hand zu partionieren, was aber angesichts der hohen Anzahl (9000 Gatter mit Testumgebung bzw. 5500 ohne Testumgebung) schnell wieder verworfen wurde. Auch die Möglichkeit einen Chip zu integrieren, der unidirektionale Signale in Tristates umwandeln kann, erwies sich ebenfalls als unpraktikabel.
Weiterhin wurde daran erinnert, daß man evtl. die FPU sockeln muß, da von dieser ein Signal durchgetrennt wird.

3 ALU

- **Multiplizierer :** Der Multiplizierer ist soweit fertig synthetisiert. Die Größe des Multiplizierers wird auf ca. 4100 Gatter geschätzt, was aber evtl. noch verbessert werden kann.
- **Dividierer :** Bei dem Dividierer gibt es noch kleinere Probleme mit dem Takten. Außerdem wird eine evtl. Division durch Null nicht berücksichtigt (wahrscheinliches Ergebnis : größtmögliche Zahl). Die Algorithmusgruppe verwies jedoch darauf, daß eine Division durch Null nicht vorkommen kann. Aus diesem Grund wird der Sonderfall nicht weiter berücksichtigt.

- **Addierer :** Der Float–Addierer konnte bereits fertig synthetisiert werden. Die Module, die für die Normalisierung zuständig sind, werden noch zu einem kompakten Modul zusammengefaßt, um so ein Einbinden der Normalisierungseinheit in andere Module zu ermöglichen.

Der Versuch, eine Testumgebung aus SpeedChart mit einem Float–Addierer zu synthetisieren verlief ebenfalls nach einigen Anfangsproblemen erfolgreich.

4 Algorithmus

Bei der Algorithmusgruppe wurde angemerkt, daß SpeedChart lediglich bei der Addition die für die Synthese notwendigen Logicvektoren benutzt. D.h., daß für alle anderen Operationen diese per Hand umgestrickt werden müssen.
Danach gab es eine kleine SpeedChart–Vorführung, wobei u.a. angemerkt wurde, daß es notwendig ist, bei einer kleinen Anzahl von Stützstellen das Register Nr. 1 mit einem Dummy-Wert zu beschreiben, da es sonst nicht weiter genutzt werden kann. Der Grund liegt darin, daß normalerweise im Register Nr. 0 die Anzahl der Stützstellen gespeichert wird und danach im 2. und 3. Register die erste Stelle, im 4. und 5. Register die nächste, usw. Sind dann alle Register verbraucht, so wird mit dem 0. und 1. Register fortgefahren. Existieren nun aber nur wenige Stützstellen, so kann es vorkommen, daß alle auf einmal übergeben werden können und so das 1. Register nie beschrieben wird. Die Folge ist, daß man dieses Register bei der Spline-Berechnung nicht nutzen kann.