Kategorie	Problem	Beschreibung	Funktion	Handling	Skalierung	Kommentar
Arbeitstechnik	Grafische Eingabe nicht wirklich unterstützend	Matlab unterstützt mit dem Blockset nicht wirklich die Generation von VHDL sondern bietet nur die Möglichkeit, dies grafisch zu erledigen. Grundsätlziche Themen des timing und der Verschaltung bleiben unverändert bestehen. Das Ganze wird allerdings umständlicher, naturgemäß laufwändiger und es entsteht - anders, als man dies vermuten könnte - auch keinesweg automatisch mehr Übersicht.	Nur auf den ersten Blick "einfacher"	zusätzlicher Aufwand	einfaches Skalieren eines Design erschwert	Grundsätlzich hinterfragenswert
Arbeitstechnik	Paralleles Arbeiten unterdrückt	Während einer laufenden Simulation oder Synthese kann nicht am Design weitergearbeitet werden. Selbst kleine grafische Änderungen werden teilweise nicht angenopmmen. Es müsste erst eine manuelle Kopie des Desingns erzeugt-, mit MATLAB geladen und gestartet werden, um Simulieren UND Arbeiten zu können.	starke Einschränkung	Erschwernis, Mehraufwand		Mehr Freizeit für den Designer :-)
Arbeitstechnik	Teamwork erschwert	Grafische Tools erschweren generell das team work. Grafische Sourcefiles lassen sind schwer bis garnicht gleichzeitg bearbeiten. Damit wird z.B. auch ein konsistentes handling von Derivate durch / mit Kollegen schwieriger	nicht vorhanden	Erschwernis	Erschwernis, Mehraufwand	nicht überall einsetzbar
Arbeitstechnik	Versionierung erschwert	Grafische Tools erschweren generell die Versionsverwaltung. Grafische Sourcefiles sind schwerer bis garnicht automatisch zu versionieren und im Zuge von Quervergleichen auf Änderungen hin untersuchbar. Konkret in Matlab Simulink fehlt eine Derivateverwaltung völlig.	starke Einschränkung	Erschwernis, Mehraufwand		nicht überall einsetzbar
Allgemein	Records nicht nutzbar	Die in VHDL häufig verwendete Funktion der Records ist in Simulink nicht verfügbar. Damit werden designs unübersichtlich, es besteht ein großer Mehraufwand bei Portierung und eine Skalierung ist unmöglich.	nicht vorhanden	zusätzlicher Aufwand	einfaches Skalieren eines Design erschwert	Verhindert sicheres Handling großer Designs
Allgemein	Generics kaum nutzbar	Die in VHDL standardmäßig verwendete Funktion der Generics ist in Simulink nicht konsistent nutzbar. Dadruch werden designs statisch, es besteht ein Mehraufwand bei der Portierung und eine Skalierung wird sehr viel aufwändiger.	nicht vorhanden	zusätzlicher Aufwand	Durchgängiges Skalieren eines Design unmöglich	Eine vollkommen unakzeptable und massive Einschränkung !
Konzept	Counter haben indirekten Takt	Counter laufen ohne echten Takt. Sie müssen über eine virtuelle Periode parametriert werden, was bei kurmmen Frequenzen zu Diskrepanzen im Laufverhalten führen kann und es ferner nötig macht, den Counter jedesmals anzufassen, wenn die Taktfrequenz geändert wird.	Problem	Hinderniss	Skalierung ist unterbrochen	ein überdenkens-wertes Konzept
Konzept	Konstanten müssen parametriert werden	Die Variablendimension muss bei z.B. Comparatoren manuell angepasst werden, statt daß sie automatisch berechnet wird.	mögliche Fehlerquelle	Erschwernis, Mehraufwand	Skalierung ist unterbrochen	vollkommen unnötig und widersinnig

Konzept	Daten-IO-müssen manuell parametriert werden	Die virtuellen Ports sowie die realen Schnittstellen zu Core-Elementen müssen bezüglich der Dateninterpretation manuell parametriert werden. Die im Design erforderliche Genauigkeit wird nicht von MATLAB durch das design durchgetrieben, sondern muss wie in VHDL auch punktuell definiert und verifiziert werden. Dabei ist Simulink aber nicht durchgängig: Einige Blöcke verarbeiten die Interpretation, andere nicht. Dies führt dazu, daß künstliche und z.T. unsinnige Interpretationen entstehen - bzw. aufgesetzt werden müssen, um Module überhaupt aneinander anschließen und verbinden zu können. Viele in VHDL sehr simple Aktionen erfordern sehr viele zusätzliche Eingaben. Das Design dauert länger und wird sehr unübersichtlich.	Problem	zusätzlicher Aufwand	Unterbrochen	Der Designer muss Simulink helfen, zu rechnen, statt umgekehrt.
Konzept	Pipelining muss per Hand erzeugt werden	Um Signale, welche aus unterschiedlichen Zeitebenen stammen mit einander verrechnen zu können, muss das durch die blocksets entstandene Delay manuell korrigiert werden, obwohl es aus den Cores bekannt ist, sein müsste. Dieser Schritt wäre leicht automatisierbar. Durch die manuelle Einstellung der Delays müssen Änderungen unter Umständen kettenförmig durch das Design durchgetrieben werden. In VHDl erfolgt eine derartige Anpassung durch Zeilenverschieben in getaktete Prozesse oder parametrierte Synthese von Delay FFs.	mögliche Fehlerquelle	zusätzlicher Design-Aufwand	nicht automatisch erzeugt und auch nicht durch Berechung skalierbar	Dies wäre das EIGENTLICHE Ziel der Übersetzung von Matlab in VHDL
Konzept	Darstellung der Simulink-Werte ungeschickt	Die simulierten Werte werden in Simulink-Scopes nicht kontinuierlich dargestellt. Ein repaint nach verlorenem Windows-Fokus führt zu einer Ausschnittdarstellung. Erst nach dem Ende der Simulation ist das komplette Bild verfügbar. Die Bedienung der Scroll - und Zoomfunktionen ist mangelhaft	Entschleunigte Entwicklung	umständliche Bedienbarkeit		Angeblich soll es doch einfacher sein, als mit ModelSIM :-)
Konzept	Darstellung der Wavescope-Werte ungeschickt	Werte werden im WAVE SCOPE nicht kontinuierlich dargestellt, sondern erst nachdem die Simulation durchgelaufen ist oder abgebrochen wurde. Die Bedienung der Scroll - und Zoomfunktionen ist eine Katastrophe	Entschleunigte Entwicklung	Sehr schlechte Bedienbarkeit		WAVESCOPE ist die übleste Komponente insgesamt!!

Realisation	Latenz in Cores verhindert Simulation	Aus einigen Cores kommen infolge von pipelining und Lazenz zu Beginn einer Simulation naturgemäß unbekannte Werte. Diese werden von Simulink unterschiedlich interpretiert und gehandelt. Teilweise ist das tool nicht in der Lage, damit umzugehen und simuliert nicht, weil nachgeschaltete Einheiten keine vernünftigen Werte bekommen. Dies führt bereits beim check der Simulation zu einem Fehler und der Unmöglichkeit, zu simulieren. Damit tut die Simulation nicht, was sie eigentlich sollte: Die Information über Existenz sowie die Dauer dieser Latzenz präzise nach Außen darstellen. Es müssen daher zusätzliche Komponenen nachgeschaltet werden, die zum startup den "U"-Datenstrom unterdrücken, damit die Schaltung simulieren kann, auch wenn dies real nicht von Nöten wäre.	Problem	zusätzlicher Aufwand	Unterbrochen	Typisches Problem solcher tools. Die Simulation sollte einfach laufen und "unknown" zeigen.
Realisation	Blockmodule komplex und versteckt parametriert	Viele Funktionen wie Addierer, Multiplizieren oder allgemeine mathematische Blöcke müssen sehr aufwändig parametriert werden, um simple Funktionen zu erzielen. Die genaue Funktion ist ihnen jedoch im grafischen Design nicht anzusehen. Daher müssen zusätzliche Kommentare eingefügt werden, um die Funktion zu beschreiben. Der Anspruch, das Design durch Grafik lesbarer zu machen, wird völlig konterkarriert.	mögliche Fehlerquelle	zusätzlicher Aufwand	Unterbrochen	Die Übersicht und Lesbarkeit des Designs wird damit nicht besser, sondern schlechter!
Realisation	embedded VHDL wrapper umständlich und statisch	Um eingebettetes VHDL im Design nutzen zu können, muss es umständlich eingefasst werden. Der dazu zu erstellende Wrapper wird nicht immer automatisch erzeugt, hat teilweise Fehler und bedarf oft der Nacharbeit. Die Portdeklaration ist bezüglich der Vektoren nicht konsistent (einmal std_logic, ein anderes mal std_logic_vector), bezieht sich zudem auf veraltete Synompsis-Bibliotheken und ist nicht ausreichend sprachtolerant. (VHDL-Standard > 2002). Problem: Sobald z.B. die Portbreite im Design geändert wird, muss der Wrapper manuell angefasst werden. Unterkomponenten müssen per Hand angegeben werden, statt sie dem design zu entnehmen, generics werden unvollständig und falsch gehandhabt.	Einschränkung		Unterbrochen	Es ist nicht einsehbar, daß ein Tool, welches vorgibt, aus einfacher Mathematik VHDL erzeugen zu können, hier nicht ohne Hilfen durch den user klarkommt
Realisation	Mathematisches Abbild und VHDL Core nicht konsisten	Das von Simulink während der Simulation auf dem PC genutzte Modell mancher Blocksets stimmt nicht bitgenau mit dem überein, was hinterher in VHDL oder als CORE-Parameter realisiert wird. Dies führt sowohl zu scheinbar funktionierenden Designs die, später versagen, wie auch zu scheinbar nicht funktionierenden Designs, die lauffähig sind.	Problem	fatal error!		dies konterkariert die Vorgehensweise, M/S zu nutzen in vollkommenster Weise

Realisation	BitGen-Timingannahme nicht aussagefähig	Wird, wie in den meisten Fällen, kein vollständiges VHDL-Design angelegt, sondern nur eine Art drop-in des algorithmischen Teils erzeugt, welches in einem größeren Design eingefasst werden soll, wie z.B. mehrere Instanzen einer Recheneinheit, so versagt die vom SysGen erzeugte Timingvorhersage mangels ausreichender Isolation (IO-FFs zu den Ports) von einem potenziellen Restdesign. Das spätere Delay wird automatisch immer überschätzt.	Funktion nutzlos	zusätzlicher Synthese-aufwand		Eine recht peinliche Schwäche. Gerade ein FPGA Hersteller sollte um diesen Effekt wissen.
Realisation	Pipelining-Anahme nicht aussagefähig	Die Funktion "check pipelining", die anhand der Zielfrequenz die nötigen FFs nach einer Unit abschätzen will, arbeitet unzuverlässig. Das Delay wird in Einzelfällen unterschätzt.	Funktion nutzlos	zusätzlicher Synthese-aufwand	nicht automatisch skalierbar	
Realisation	LoLevel timing und Signale sehr schlecht beobachtbar	Um Delays und Timingaspekte präzise einstellen zu können, ist es erforderlich, Single auf Taktebene zu verfolgen. Das in Simulink angebotene Tool "WaveScope" arbeitet konzeptionell ungeschickt, hat eine Reihe von Bedienschwächen, die das Anzeigen verhindern und kommt überdies mit extrem vielen bugs: Werte werden gar nicht, oder nicht kontinuierlich angezeigt, Signale verschwinden, sie lassen sich infolge eines Programmierfehlers im popup-Window nach einer Weile nicht mehr skalieren oder sie ändern scheinbar ihre Namen. So werden falsche und verwirrende Informationen erzeugt, die es in Einzelfällen verunmöglichen, die korrekte Funktion der Signale zu prüfen.	Funktion weitgehend wertlos, teilweise fehlerhaft	zusätzlicher Verfikations-Aufwand	mehrdimensionale Signale nicht anzeigbar	Das Tool ist eine Zumutung und gelinde gesagt, eine Frechheit. Etwas anfängerhafteres habe ich selten gesehen.
Realisation	Port-numerierung nicht konsistent	Die Nummerierung der Ports, die die Reihenfolge der Pins in Symbolen bei Subsystemen und Co-Simulationsblöcken definiert, wird nicht konsisten abgebildet. Dies führt zu vollkommen unnötigem Mehraufwand bei der Generation von CO-Simulations-Designs und Alternativen und erschwert Änderungen und Updates		sehr viel zusätzlicher Zeichen-Aufwand		typisches Xilinx Chaos!

Realisation	Bitlevel Operationen unnötig kompliziert	Bei Zusammenfügen von Bitvektoren muss deren mathematische Wertinterpretationen manuell auf "neutral" gestellt- also die Information zerstört werden, um sie aneinanderbinden zu können, obwohl z.B. Bitvektoren grundsätzlich nicht notwendigerweise Zahlen darstellen, folglich keine Interpretation besitzen (müssen). Das manuelle und ausdrückliche Zerstören der Information ist also im Fall unspezifizierter Werte (z.B. wenn aus einer Zahl ein Wert und gleichzeitig ein Takttiming abgeleitet wird) entweder unnötig oder redundant, da die Informationsverfälschung implizit bereits durch die Bitoperation erfolgt. Ein grundsätzliches "unknown" am Ausgang des Concat würde völlig reichen. In den Fällen, in denen hingegen spezifizierte Werte entstehen und die Interpretation eindeutig ist, wie z.B. bei einem Concat aus zwei tatsächlich mathematischen Werten, ist diese vorherige Zerstörung komplett unsinnig, da hinterher wieder eine händische Reinterpretation aufgesetzt werden muss, die volllkommen eindeutig ist, also durch MATLAB leistbar wäre.	weniger Übersicht, mangelnde Funktion, große Fehlerquelle	viel zusätzlicher und unnötiger Zeichen-Aufwand, stark verschlechtere Übersicht durch mehr Symbole	Eine System-skalierung durch simplen Aufrechterhalt von Verknüpfungen wird aufgrund ausdrücklicher Interpretatione n unterwegs unnötigerweise verhindert!	Notwendigkeit ist unsinning sowie hinderlich und aus konzeptioneller Sicht Irrational: Bitoperationen brauchen keine mathematische Interpretation. Dort hingegen, wo sie mal vorhanden ist und nutzbar wäre, wird sie ignoriert.
Realisation	Bitlevel Operationen unnötig kompliziert	Bei der Realisation von Ram-Anschlüssen muss die mathematische Wertinterpretationen manuell auf "neutral" gestellt werden. Durch das Zerstören der Information ist es unmöglich, den Wert aus dem (dual ported) RAM beim Auslesen zu reinterpretieren. Bei dem simplen Auslesen aus einem RAM, in das vorher ein gleich breiter Vektor eingeschrieben wurde, ist diese Zerstörung komplett unsinnig, da wieder händisch reinterpretiert werden muss, was durch MATLAB leistbar wäre. Wo also in VHDL ein simples Zusammenfügen oder Anschalten eines Vektors gereicht hätte, muss die Stimmingkeit jedesmal künstlich (wieder-) hergestellt werden.	weniger Übersicht, mangelnde Funktion, mögliche Fehlerquelle	viel zusätzlicher und unnötiger Zeichen-Aufwand, stark verschlechtere Übersicht durch mehr Symbole	Skalierung des Systems durch simples Fortschreiben der Interpretation der Werte nicht möglich, obwohl durchaus leistbar	Siehe oben! Was soll aus einem Ram denn bitte rauskommen, wenn nicht das, was man reinschreibt?
Realisation	Parameter verändern sich oder werden nicht ordnungsgemäß gespeichert	Parameter verändern sich oder werden nicht ordnungsgemäß gespeichert. Ein Hinweis auf Inkonsistenz sind fehlerhafte Grafikdarstellungen, die beim öffnen eines kürzlich geänderte Designs hochkommen: Hier sitzen Linien und Obkejte an anderen Stellen, als ursprünglich plaziert. Siehe auch den Effekt bei Wavescope.	fatal!			So ziemlich die größte Katastrophe!
Realisation	Fehler in VHDL wird mit falscher Zeilennummer angegeben	Gibt es bei der Übersetzung / Einbindung von VHDL einen Fehler, so wird die Zeilennummer des übersetzten Codes angegeben. Dieser enthält in einem großen file u.a. auch die Quell-VHDL-files. Damit wird eine Zeilennummer in der Größenordung von Tausenden ausgegeben, die nicht direkt auf die fehlergafte Zeile im Code hindeutet.	eingeschränkte r Nutzen	zusaätlzicher Suchaufwand		Peinlicher Denkfehler!