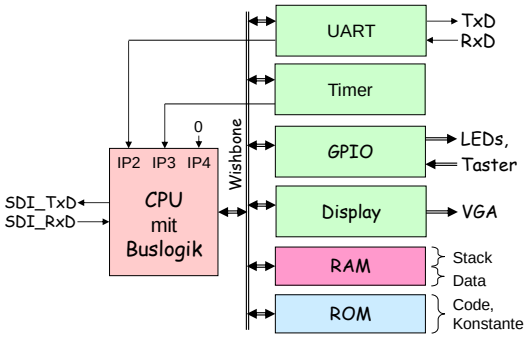
Praktikum Digitaler Komponenten

Übung 4: Einbinden von Peripheriekomponenten, Realisierung einer Applikation

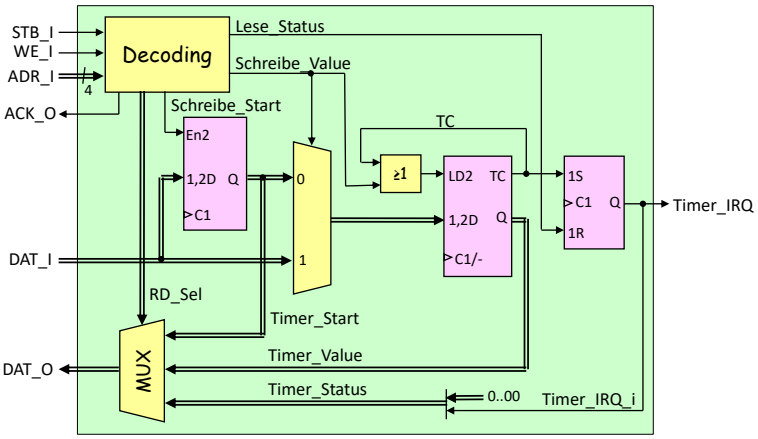
Das in der Vorlesung vorgestellte Beispielrechner-System wird um eine Timer-Komponente und um die Komponente zur Ansteuerung eines VGA-Displays erweitert. Für das erweiterte System wird eine Applikation erstellt, welche eine Uhr realisiert. Die Uhrzeit soll sowohl über die serielle Schnittstelle als auch auf dem VGA-Display ausgegeben werden. Das Stellen der Uhr soll über Taster und alternativ über die Zeichen 'h', 'm' und 's', die von der seriellen Schnittstelle empfangen werden, möglich sein.



Aufgabe 1: Implementierung und Integration einer Timer-Komponente

Implementieren Sie die in der Vorlesung vorgestellte Timer-Komponente. Führen Sie dazu die folgenden Schritte aus:

- Ausgangspunkt dieser Übung ist das Beispielrechner-System aus Übung 3.
- Kopieren Sie die bereitgestellten Dateien so in das Arbeitsverzeichnis von Übung 3, dass die Verzeichnisse `Hardware` und `Software` in die gleichnamigen Verzeichnisse integriert werden.
- Vervollständigen Sie in der Datei `Hardware\Timer\Timer.vhd` die VHDL-Beschreibung der in der Vorlesung vorgestellten Timer-Komponente entsprechend dem folgenden Blockschaltbild:



Die Positionen der von Ihnen zu erstellenden Programmteile in der VHDL-Datei sind mit TODO markiert. Beachten Sie dabei die nachfolgenden Hinweise:

- **Dekodierung** der Bussignale im Prozess „Decoding“: Die benötigten Adress-Offsets der Register können Sie der folgenden Tabelle entnehmen:

Offset	Register
0x0	Timer_Value
0x4	Timer_Start
0x8	Timer_Status

  Bei der Dekodierung muss den Signalen „ACK_O“, „Lese_Status“, „Schreibe_Value“ und „Schreibe_Start“ ein Wert zugewiesen werden.

- **Lesedatenmultiplexer** im Prozess „Mux_Lesedaten“: Für die Auswahl des Eingangssignals wird das Signal „RD_Sel“ vom Aufzählungstyp „RD_Mux_Type“ verwendet.

- **Register** für den Startwert des Zählers im Prozess „Reg_Timer_Start“: Das Register soll den Wert vom Eingang DAT_I übernehmen, wenn das Signal „Schreibe_Start“ aktiv ist.

- **Zähler** im Prozess „Reg_Timer_Value“: Sie können die Funktionen des vorgeschalteten Multiplexers und des Oder-Gatters mit einbeziehen. Aus dem Wert von „Schreibe_Value“ und den alten Werten von „Timer_Value“ und „TC“ muss der neue Wert für „Timer_Value“ bestimmt werden:

Schreibe_Value	TC (alt)	Timer_Value (alt)	Timer_Value (neu)
1	-	-	DAT_I
0	1	-	Timer_Start
0	0	>0	Timer_Value - 1

  Das Signal „TC“ wird auf ‚1‘ gesetzt, wenn Timer_Value = 0 ist.
- **Register** im Prozess „Reg_Timer_IRQ_i“: Das Signal „Timer_IRQ_i“ soll auf den Wert 0 gesetzt werden, wenn das Signal „Lese_Status“ den Wert 1 hat. Falls das Signal „TC“ den Wert 1 hat, soll das Signal „Timer_IRQ_i“ auf den Wert 1 gesetzt werden.

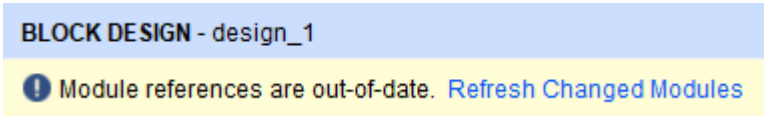
- Verifizieren Sie die Funktion der Komponente mit dem VHDL-Simulator (QuestaSim oder ModelSim). Setzen Sie dazu dessen Arbeitsverzeichnis auf das Verzeichnis `Hardware\Timer`. Nutzen Sie dabei die Testbench `Timer_tb.vhd` mit dem Simulationsskript `test_Timer.do`.

Aufgabe 2: Integration der Timer-Komponente in das Beispielrechner-System

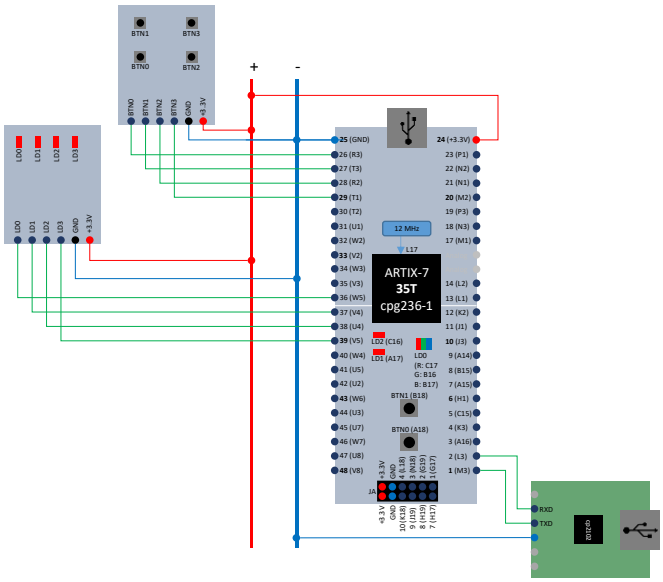
Integrieren Sie die in Aufgabe 1 erstellte Timer-Komponente in das Beispielrechner-System aus Übung 3. Führen Sie dazu die folgenden Schritte aus:

- Instanziieren Sie dazu in der Datei `Hardware\Beispielrechner_System.vhd` den Timer mit dem Instanznamen „Timer_Inst“. Deklarieren Sie die für die Einbindung benötigten Signale. Vervollständigen Sie die Busanbindung und legen Sie den Adressraum der Timer-Komponente dabei auf 0x00008300 – 0x0000830F fest. Verbinden Sie den Port „Timer_IRQ“ der Timer-Komponente mit dem Signal IP3.
- Starten Sie Eclipse und verwenden Sie wie in Übung 3 das Verzeichnis `Software` als Workspace. Importieren Sie das neue Projekt `TestTimer` in den Workspace. Dieses enthält ein Programm, welches die LEDs im zeitlichen Abstand von 10 µs blinken lässt. Erneuern Sie nach erfolgter Übersetzung die VHDL-Speicherbeschreibungen durch Ausführung der Batch-Datei `Hex2VHDL.bat` im Projekt `TestTimer`.
- Simulieren Sie das Gesamtsystem im VHDL-Simulator. Nutzen Sie dazu die Testbench `Beispielrechner_System_V4_testbench.vhd` in Verbindung mit dem Simulationsskript `test_Beispielrechner_System_V4.do`. Überprüfen Sie im Wave-Diagramm, ob periodisch in Abständen von 10 µs Interrupts angefordert und daraufhin die LEDs umgeschaltet werden. Falls dies nicht der Fall ist, suchen und beheben Sie die Fehler.

- Öffnen Sie in Vivado das bereits in Übung 3 erstellte Projekt `Synthese`, fügen Sie diesem die Datei `Hardware\Timer.vhd` hinzu („Add Sources“). Vivado schlägt vor, die Änderungen in den VHDL-Dateien für das Blockdesign zu verwenden. Folgen Sie diesem Vorschlag (Refresh Changed Modules).



- Lassen Sie Vivado den Bitstream neu erzeugen („Generate Bitstream“).
- Verwenden Sie wieder den bereits aus Übung 3 bekannten Hardwareaufbau:

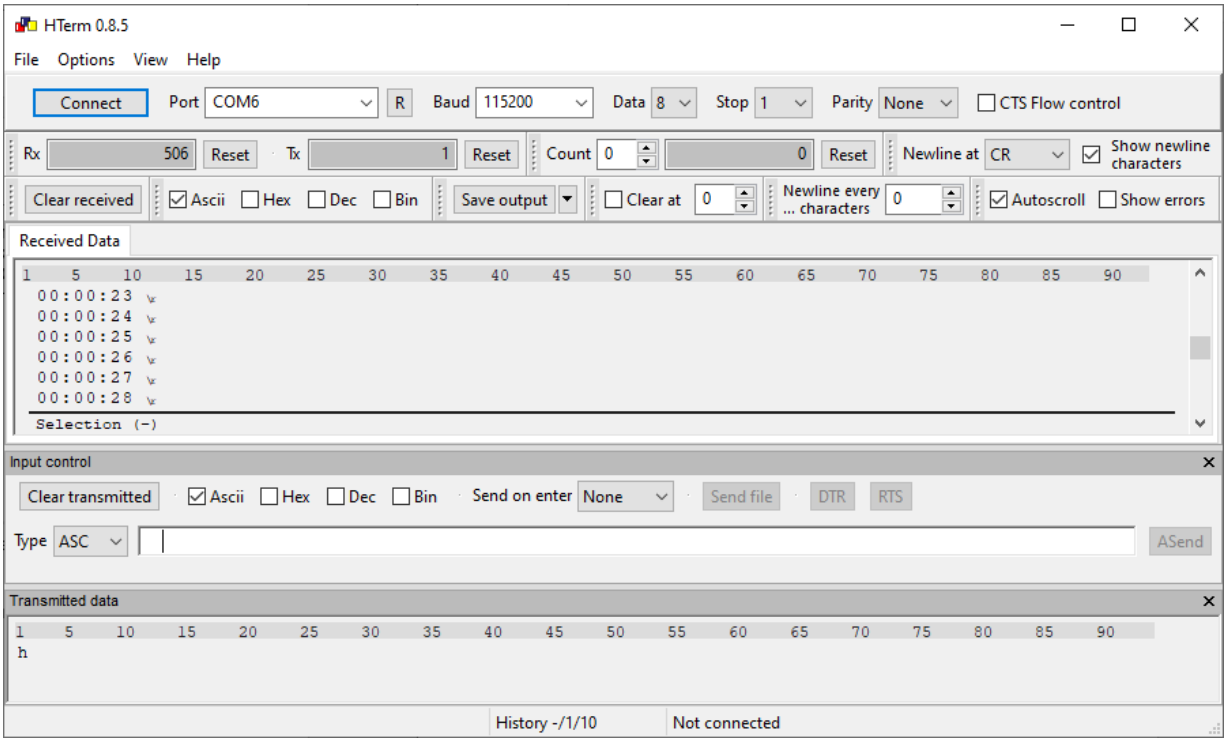


- Programmieren das FPGA Sie mit dem soeben erzeugten Bitstream. Nutzen Sie dazu das Batch-File `Tools\program_fpga.bat`.
- Wechseln Sie wieder zu Eclipse. Passen Sie die zeitliche Verzögerung in der Software in der Datei `main.c` an, so dass eine Blinkfrequenz von 1 Hz erreicht wird.
- Starten Sie den GDB-Remote-Server mit dem Batch-File `Tools\run_remote.bat`, erzeugen Sie eine Debug-Konfiguration für das Projekt `TestTimer` und führen Sie damit das Programm aus. Nun sollten die LEDs mit einer Frequenz von 1 Hz blinken.

Aufgabe 3: Erstellung einer Software-Applikation (Uhr)

Erstellen Sie eine Software-Applikation zur Anzeige der Uhrzeit nach dem in der Vorlesung vorgestellten Verfahren zum zeitgesteuerten Arbeiten. Die dafür benötigten Programmfragmente finden Sie in den Vorlesungsunterlagen. Führen Sie dazu die folgenden Schritte aus:

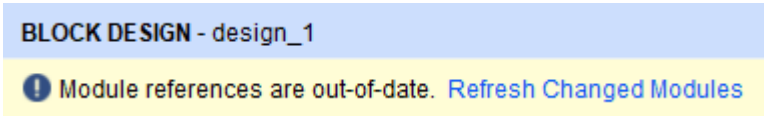
- Importieren Sie das neue Projekt „Uhr“ in den Eclipse-Workspace. Dieses enthält ein Programmfragment, mit dem die in der Vorlesung vorgestellte Applikation „Uhr“ realisiert werden kann.
- Ergänzen Sie in dem vorgegebenen Programm in der Datei `main.c` die folgenden Funktionen. Die zu ergänzenden Stellen sind mit „TODO“ markiert. Orientieren Sie sich dazu an den Vorlesungsunterlagen:
  - Timer_Init
  - Timer_Handler
  - main
  - increment_clock
  - show_clock (dabei – falls vorhanden – die Zeilen zur Display-Ausgabe auskommentieren)
  - check_buttons
  - check_inbyte
- Starten Sie den GDB-Remote-Server mit dem Batch-File `Tools\run_remote.bat`, erzeugen Sie eine Debug-Konfiguration für das Projekt `Uhr` und führen Sie damit das Programm aus.
- Starten Sie auf dem PC ein Terminal-Programm (z.B. HTerm) und stellen Sie dort den COM-Port für die serielle Übertragung und die Übertragungsparameter wie im Programm vorgegeben ein (115200 Baud, 8 Datenbits, 1 Stoppbit, kein Paritätsbit). Prüfen Sie, ob Sie durch das Drücken der Taster und durch das Senden der Zeichen h, m und s die Uhrzeit verändern können.



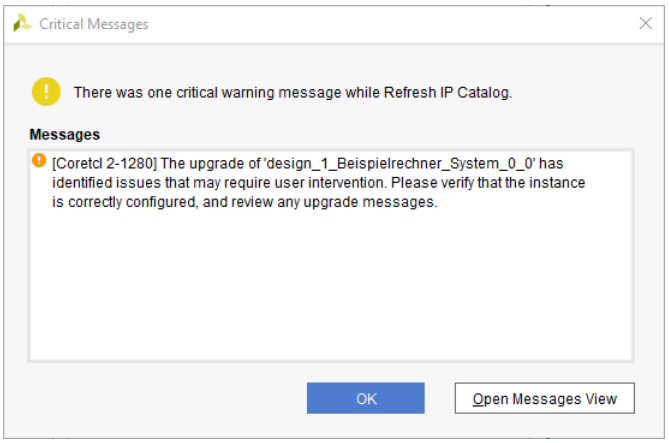
Aufgabe 4: Integration einer Display-Komponente (falls VGA-Display vorhanden)

Integrieren Sie die vorgegebene Display-Komponente in das Beispielrechner-System. Lassen Sie die Uhrzeit auf einem angeschlossenen VGA-Display anzeigen. Führen Sie dazu die folgenden Schritte aus:

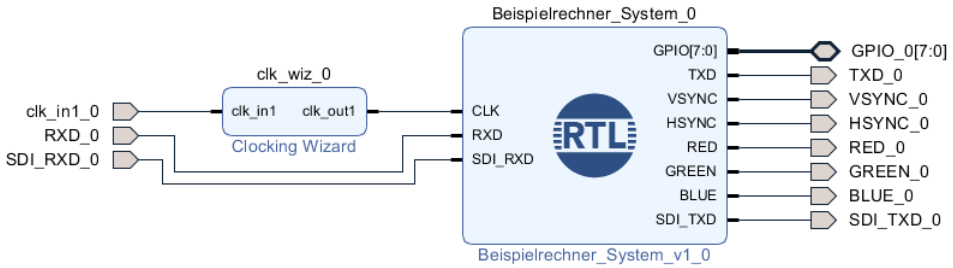
- Im Verzeichnis `Hardware\Display` finden Sie die Hardware-Bestandteile der Display-Komponente.
- Fügen Sie in der Datei `Hardware\Beispielrechner_System.vhd` folgende Ausgangs-Ports zur entity hinzu: VSYNC, HSYNC, RED, GREEN, BLUE (alle vom Typ std_logic).
- Instanziieren Sie die Komponente „Display“ mit dem Instanznamen „Display_Inst“. Deklarieren Sie die für die Einbindung benötigten Signale. Vervollständigen Sie die Busanbindung und legen Sie den Adressraum der Display-Komponente dabei auf 0x00010000– 0x0001FFFF fest.
- Öffnen Sie wieder das Vivado-Projekt `Synthese`, fügen Sie diesem alle Dateien aus dem Verzeichnis `Hardware\Display` hinzu („Add Sources“). Falls das Block Design nicht geöffnet ist, öffnen Sie es (Open Block Design). Vivado schlägt Ihnen vor, die Änderungen in den VHDL-Dateien für das Blockdesign zu verwenden. Folgen Sie diesem Vorschlag (Refresh Changed Modules).



Vivado weist nun mit einer Meldung daraufhin, dass sich die Schnittstelle des Beispielrechnersystems geändert hat. Das sollte auch so sein, es wurden schließlich neue Signale hinzugefügt. Das Fenster mit der Meldung können Sie also einfach mit „OK“ schließen.



Die neuen Signale sollen aus dem FPGA herausgeführt werden, deshalb sollen externe Ports für sie angelegt werden. Am einfachsten gelingt dies, wenn man die kurze Linie neben der Portbezeichnung rechtsklickt und im Kontextmenü das Kommando „Make External“ auswählt (Shortcut Strg-T). Die Namen der mit „Make External“ erstellten Ports werden automatisch durch das Suffix „_0“ ergänzt:

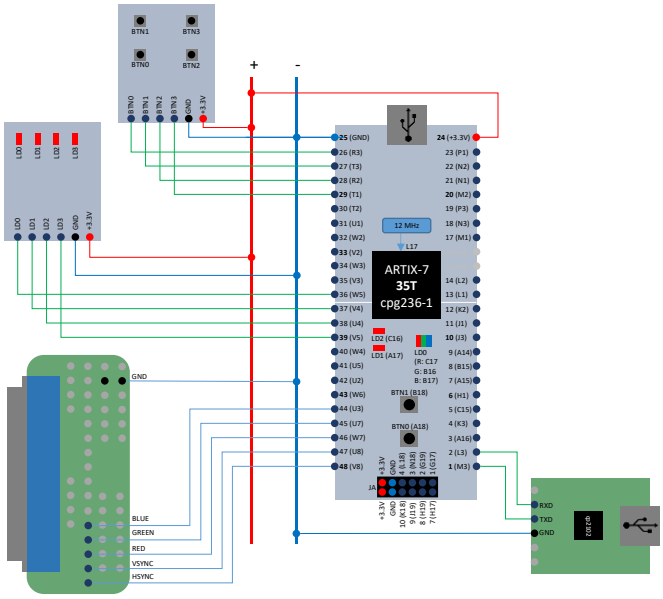


- Um die FPGA-Pins für die neuen Ports festzulegen, müssen Constraints ergänzt werden. Ergänzen Sie dazu in der Datei `Beispielrechner_System.xdc` die folgenden Zeilen:

```
# Display
set_property -dict {PACKAGE_PIN U3 IOSTANDARD LVCMOS33} [get_ports {BLUE_0 }];
set_property -dict {PACKAGE_PIN U7 IOSTANDARD LVCMOS33} [get_ports {GREEN_0}];
set_property -dict {PACKAGE_PIN W7 IOSTANDARD LVCMOS33} [get_ports {RED_0  }];
set_property -dict {PACKAGE_PIN U8 IOSTANDARD LVCMOS33} [get_ports {VSYNC_0}];
set_property -dict {PACKAGE_PIN V8 IOSTANDARD LVCMOS33} [get_ports {HSYNC_0}];
```

- Lassen Sie Vivado daraufhin den Bitstream neu erzeugen („Generate Bitstream“).

- Der Hardwareaufbau wird mit einer Platine erweitert, die einen sehr einfach aufgebauten Digital-Analog-Wandler zum Anschluss eines VGA-Monitors enthält:



- Programmieren Sie mit dem aktualisierten Bitstream das FPGA. Nutzen Sie dazu das Batch-File `Tools\program_fpga.bat`. Ein angeschlossener Monitor sollte nun ein blaues Bild zeigen.
- Wechseln Sie wieder zu Eclipse. Das Software-Projekt Uhr enthält bereits die zur Display-Komponente gehörenden Dateien `display.c` und `display.h`. In der Datei `config.h` muss noch die Konfiguration für die Display-Komponente ergänzt werden:

```
// Display configuration
#define DISPLAY_BASE     0x00010000
#define DISPLAY_WIDTH    80
#define DISPLAY_HEIGHT   30
```

- Ergänzen Sie die Datei `main.c` um eine Include-Anweisung für die Header-Datei `display.h`:

```
#include <display.h>
```

- In der Funktion „show_clock“ können Sie nun die auskommentierten Zeilen zur Anzeige der Zeit auskommentieren (falls schon vorhanden) oder ergänzen:

```
display_set_cursor(0, 0);
display_puts(buffer);
```

- In der Funktion „main“ fehlt noch die Initialisierung der Display-Komponente:

```
display_init(DISPLAY_BASE, DISPLAY_WIDTH, DISPLAY_HEIGHT);
display_clear();
```

- Starten Sie den GDB-Remote-Server mit dem Batch-File `Tools\run_remote.bat` und führen Sie das Programm mit der bereits erstellten Debug-Konfiguration erneut aus.
- Nun sollte auch auf Ihrem Monitor die Uhrzeit zu sehen sein.