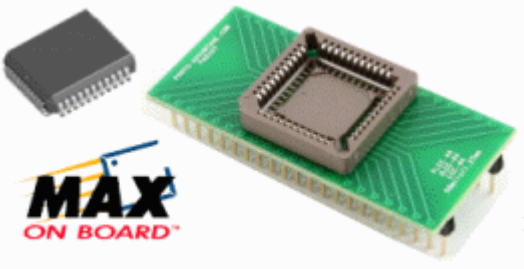
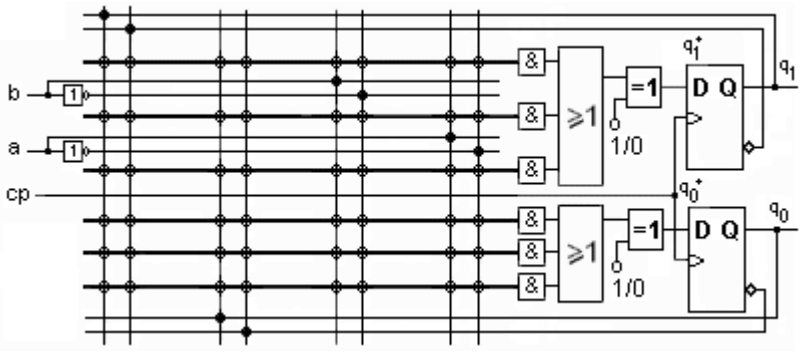
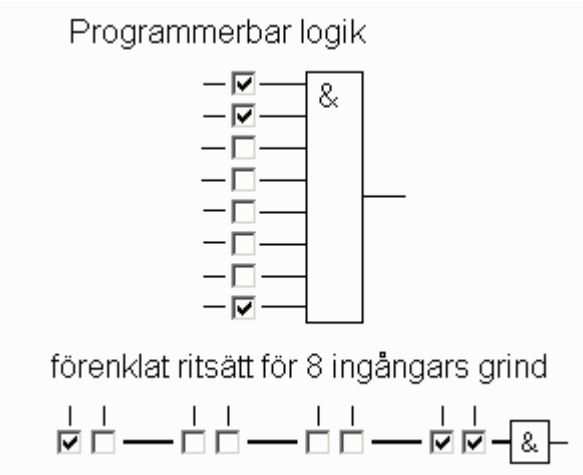
Moore-automat med Programmerbar logik och VHDL

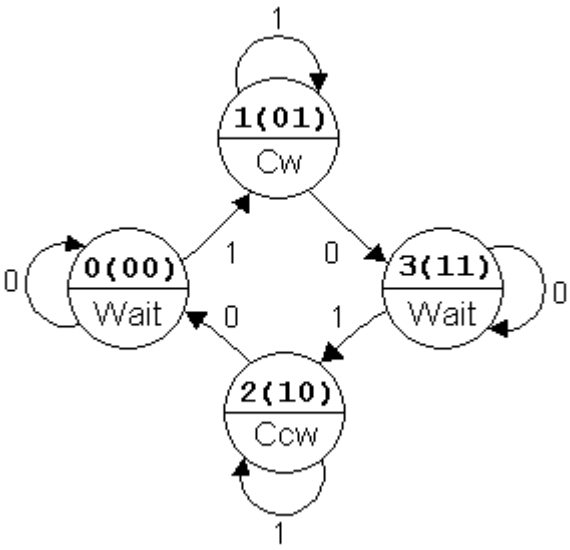


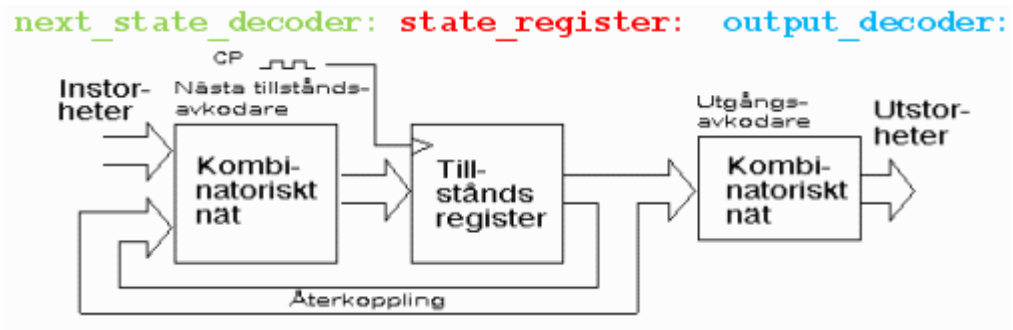


Programmerbara logikkretsar innehåller vippor med förberedda logiknät. Genom att specifiera vilka signaler man vill ha anslutna till grindarna konfigurerar man den önskade funktionen.



VHDL Hårdvarubeskrivande språk





Motor-styrningen är utformat som en Moore-atomat. Moore-modellens olika delar är utmärkta med rubriker (labels) i programmet.

```
next_state_decode:
output_decode:
state_register:
```

Förr skrev man konfigureringsinformationen för hand. Numera anger man kretsens önskade beteende med ett hårdvarubeskrivande programmeringsspråk tex. **VHDL**. Koden har som synes vissa likheter med ett vanligt C-program.

```
-- VHDL
-- example for IL131V PIC processors
-- code by Johan Wennlund KTH

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;

entity toggledir is
port (    I: in std_logic ;
        CLK: in std_logic ;
          Q: out std_logic_vector ( 2 downto 1 ) );
end toggledir ;

architecture behavior of toggledir is
subtype state_type is integer range 0 to 3 ;
signal present_state, next_state : state_type;

begin
next_state_decode:
process ( present_state, I )
begin
    if I = '1' then
      case present_state is
        when 0 => next_state <= 1;
        when 1 => next_state <= 1;
        when 3 => next_state <= 2;
        when 2 => next_state <= 2;
      end case;
    else  -- I = '0'
      case present_state is
        when 0 => next_state <= 0;
        when 1 => next_state <= 3;
        when 3 => next_state <= Ooops;
        when 2 => next_state <= Ooops;
      end case;
    end if ;
end process ;

state_register:
process ( CLK )
  begin
    if rising_edge( CLK ) then
        present_state <= next_state ;
    end if ;
end process ;

output_decode:
Q <= conv_std_logic_vector (present_state, 2);

end architecture behavior ;
```

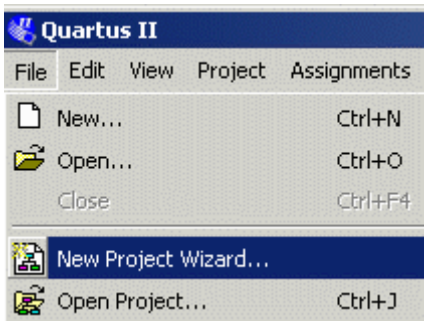
VHDL-program med Quartus

Logga in på labdatorn. På skolans centralt administrerade labdatorer har Du inte rättigheter att installera program. **Quartus II** finns redan installerad. Du har inte heller åtkomst till mappar under `C:\`. Vid laborationer ska Du därför använda din "server"-mapp `H:\`.

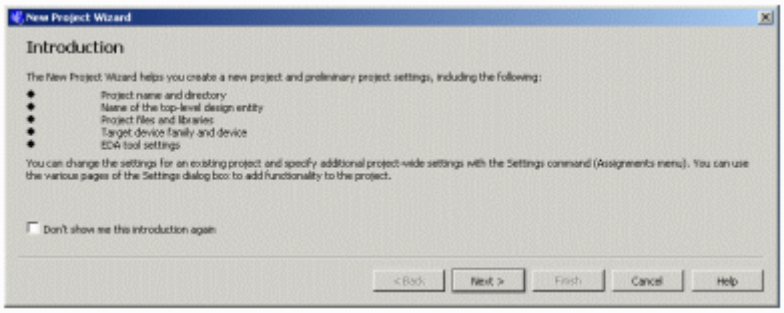
Skapa en mapp `H:\MAXWork\` för filerna i denna lab. Viktig operativsystem-inställning. Visa filnamnstillägg bör vara inställt vid alla programmeringskurser!

- Windows 7 show fileextensions

Starta Quartus. Blir Du inte direkt erbjuden att starta **New Project Wizard**, så kan Du även välja det alternativet från **File**-menyn.



Introduction
Klicka på **Next**.



Project Name and Directory

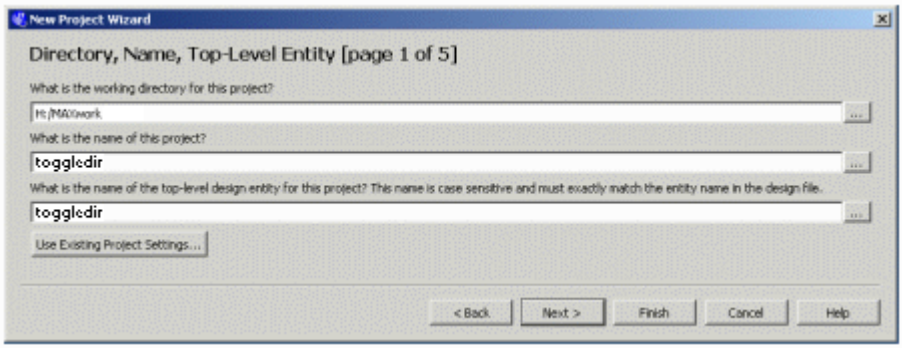
I skolan måste hela projektet ligga i en mapp på ditt `H:\`, tex. `H:\MAXwork`

Name: `toggledir`

Top-Level Entity: `toggledir`

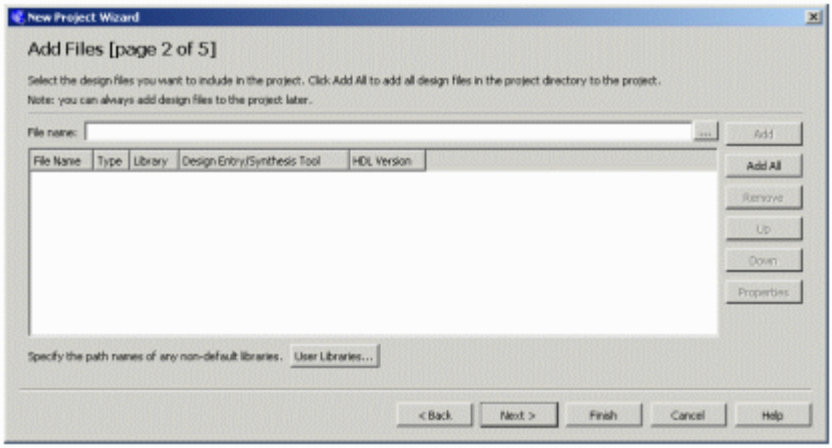
(OBS namnet `toggledir` måste "matcha" det namn Du senare anger som entity i din VHDL-fil)

Gå vidare med **Next**.



Add files

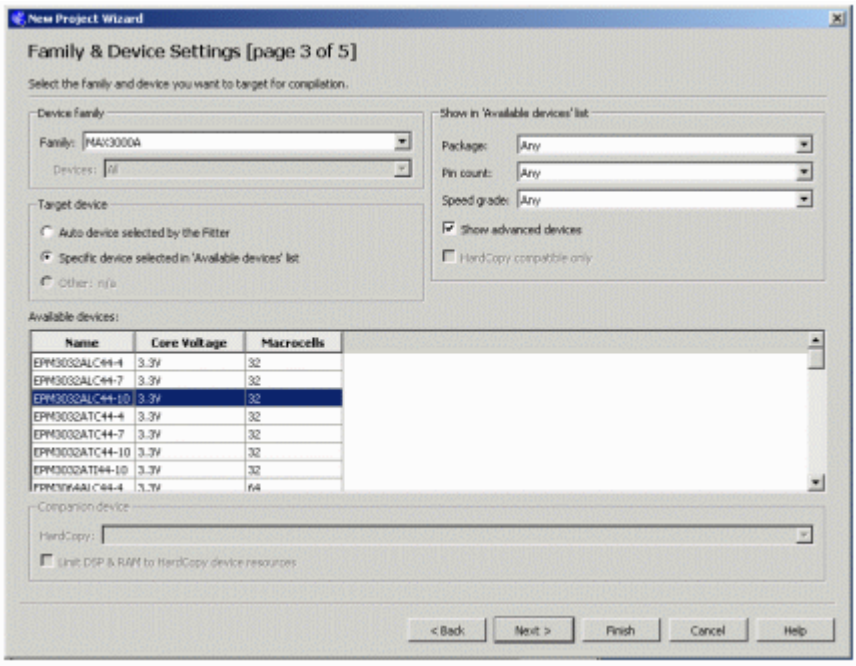
Vi har inga filer att lägga till projektet, så vi går vidare med **Next**.



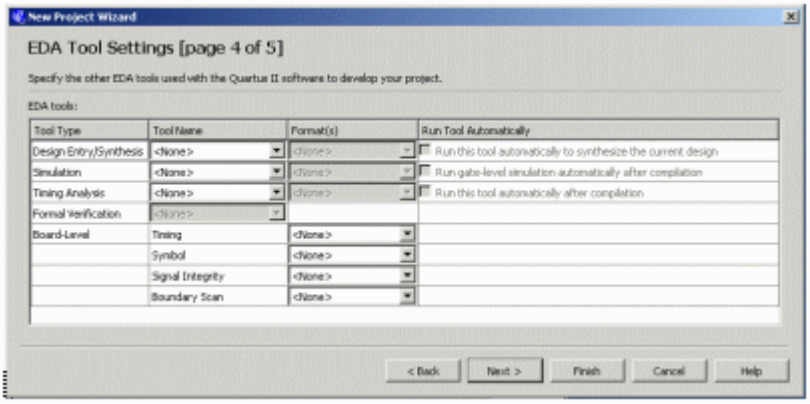
Family and Device Settings
Här anger vi vilket chip vi tänker använda under laborationen.

Family: `MAX3000A` **Available devices:** `EMP3032ALC44-10`

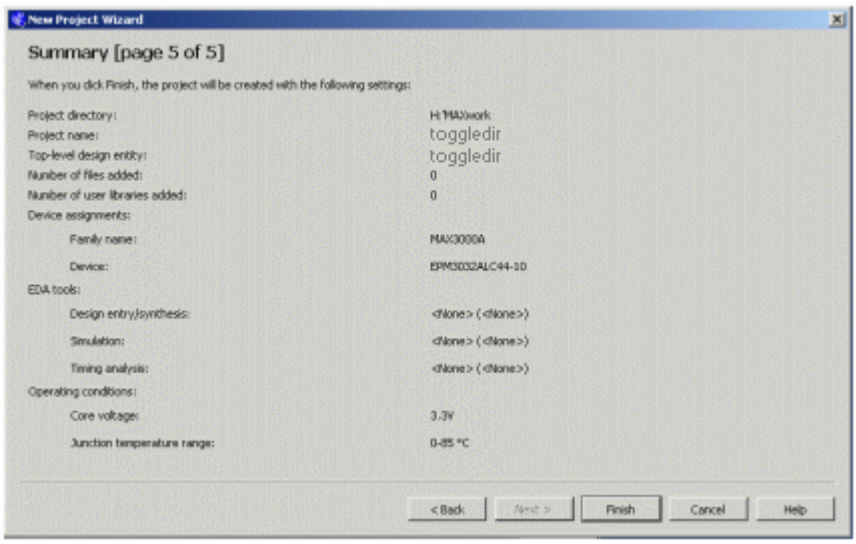
Gå vidare med **Next**.



EDA tools setting
Här kan man skapa sammanhang med programverktyg från andra leverantörer. Vi använder inga sådana.
Gå vidare med **Next**.



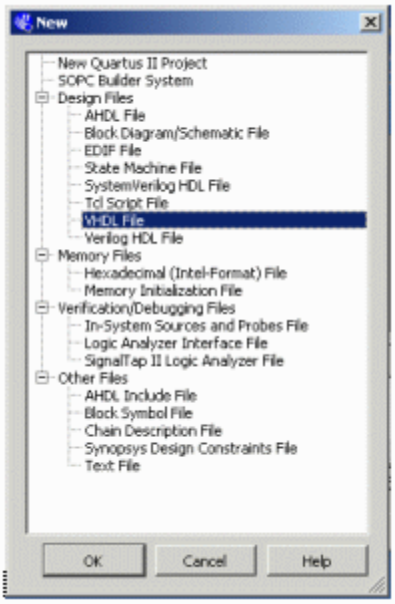
Summary, sammanfattning.
Här ser Du en sammanfattning av dina val, avsluta "Wizard" med **Finish**.



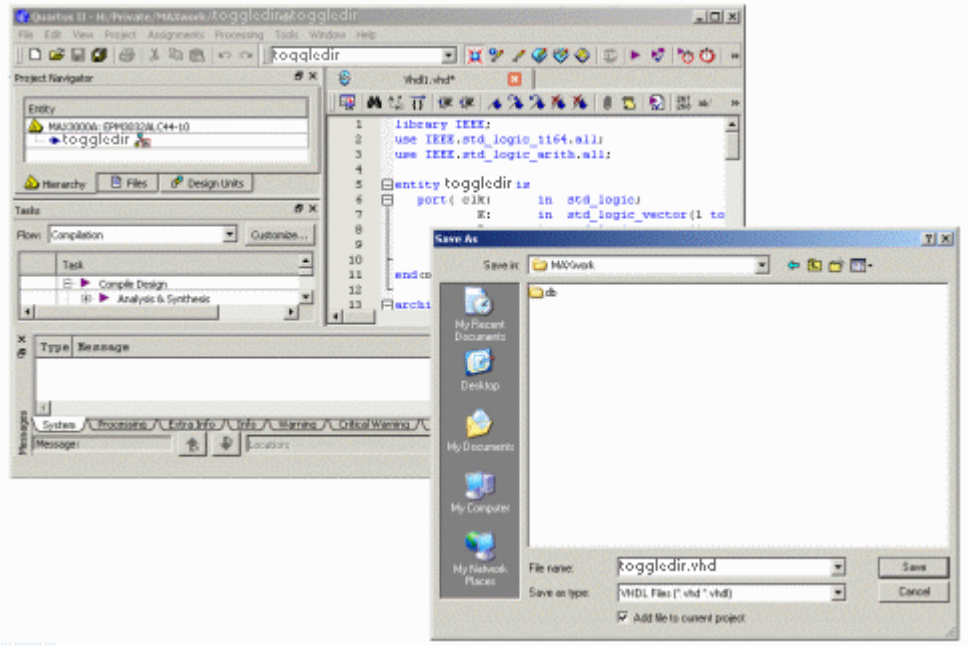
Projektet har skapats

VHDL-koden

Skapa en blank fil för VHDL-koden. **File**, **New**, **VHDL File**.



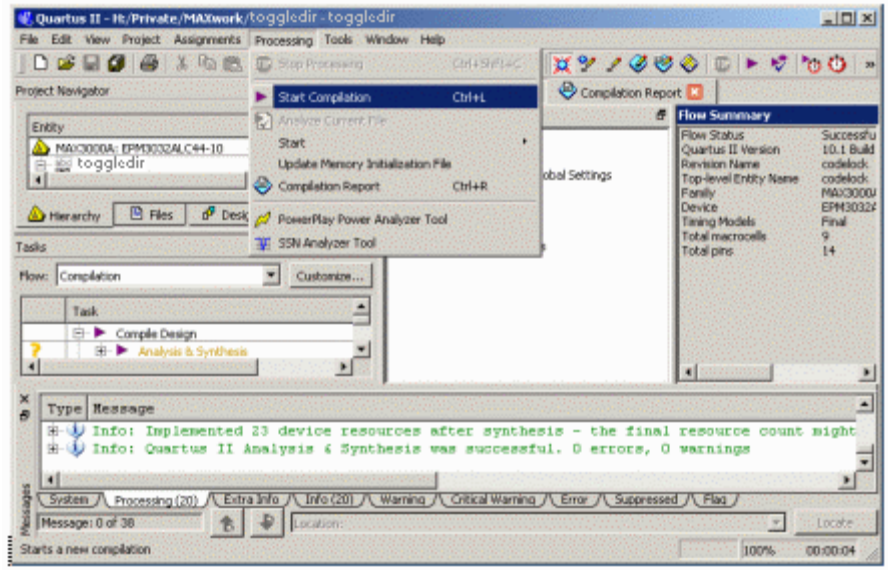
- Kopiera Mall-programmet `toggledir_mall.vhd` och klistra in det i Quartus texteditor.
- Rätta programmet! Du behöver åtgärda två ställen med texten Ooops!



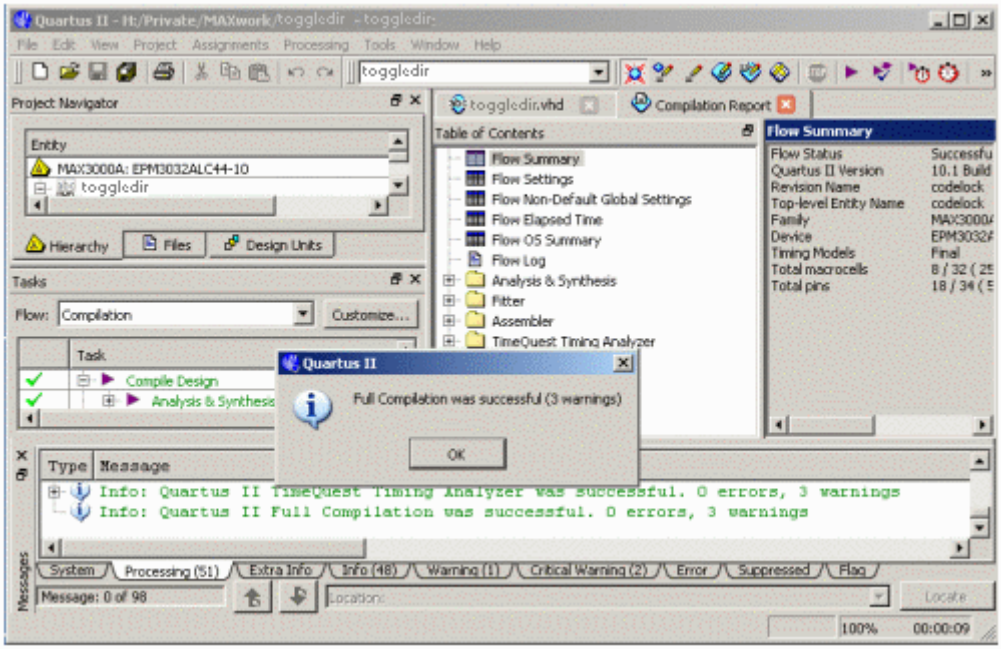
Observera att entity i VHDL-filen ska matcha projektets Top Level Entity!

Spara filen med: **File**, **Save As** och som VHDL-fil. Namnet kan vara `toggledir.vhd` (eller annat).
Add File to current project skall vara förbockat!

Start Compilation



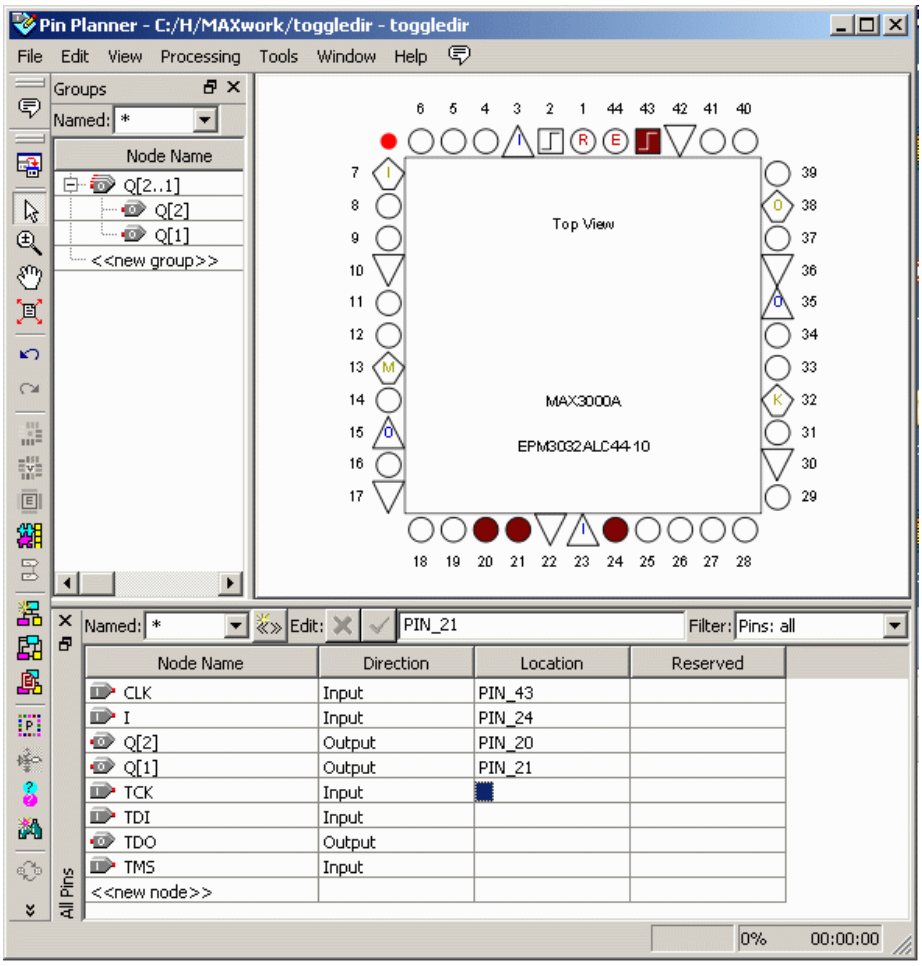
Start Compilation kör en hel kedja av verktyg.



Varningarna handlar om "verktyg" som saknas i vår programversion men som vi inte behöver.
(Antalet varningar kan variera lite med programversion, detta är ingen anledning till oro).

Pin-planering

Starta Pin Planner. Meny **Assignements**, menyval **Pin Planner**, eller genom att klicka på ikonen för Pin Planner.



I Pinnplaneraren finns en bild över det valda chipets pin-layout, samt en lista över de variabler/signaler som förekommer i VHDL-koden. Man skriver dit önskade pin-nummer i kolumnen **Location**. TCK, TDI, TDO, TMS är anslutningarna till JTAG-kontakten.

- CLK=PIN_43, I=PIN_24, Q[2]=PIN_20, Q[1]=PIN_21.

Stäng därefter Pin-planerar fönstret.

Start Compilation

Start Compilation kör hela verktygskedjan en gång till för att kompilera om, nu med den nya pin-informationen.

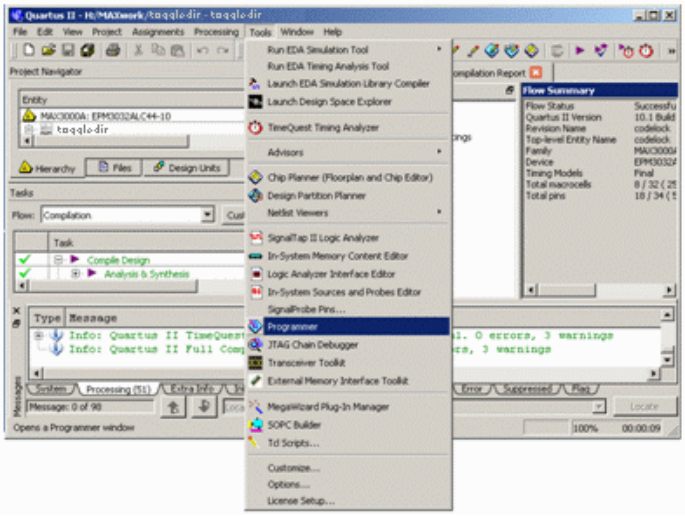
Även denna gång ska kompileringen gå bra, med undantag för varningarna (som vi ignorerar).

Chip-programmering med Quartus

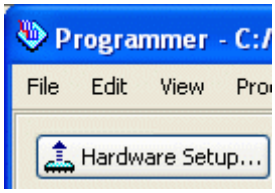
Val av programmeringsdon - USB-Blaster

Anslut USB-blastern till datorn. Anslut USB-Blasterns JTAG-kontakt till labutrustningen. Labutrustningens spänning skall var frånslagen när du sätter i eller drar ur JTAG-kontakten. Labutrustningens spänning skall vara påslagen när Du programmerar chippet - USB-strömmen ensam räcker inte till detta.

Inifrån Quartus väljer Du menyalternativet **Tools** och **Programmer**, eller så klickar Du på ikonen Programmer.

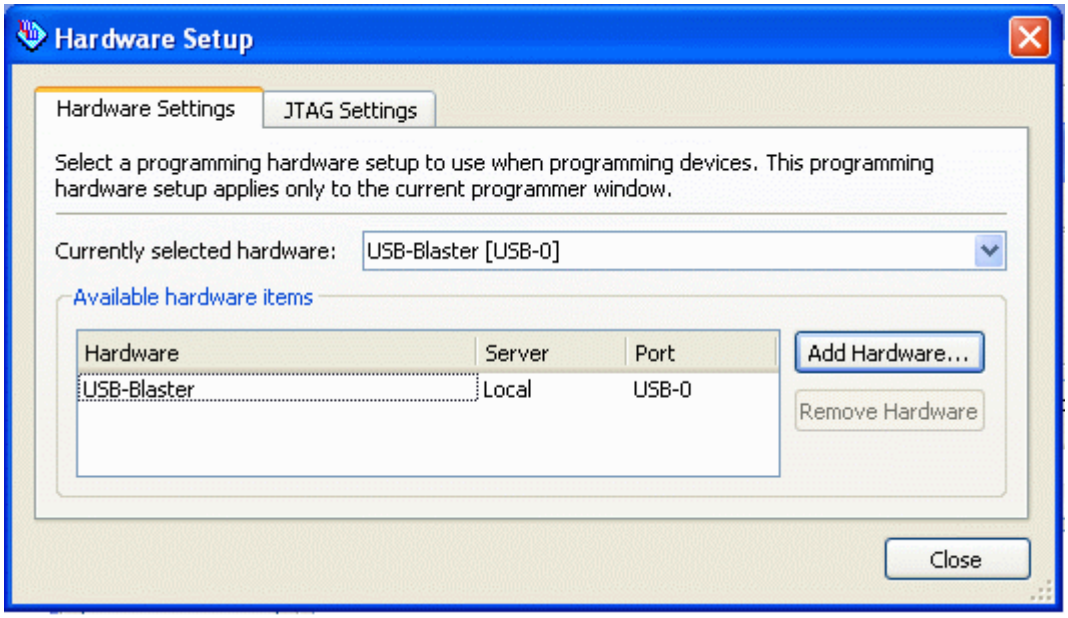


I fönstret **Programmer** klickar man på **Hardware Setup** för att kunna välja USB-blastern som programmeringsdon. (Datorn kommer sedan ihåg denna inställning)

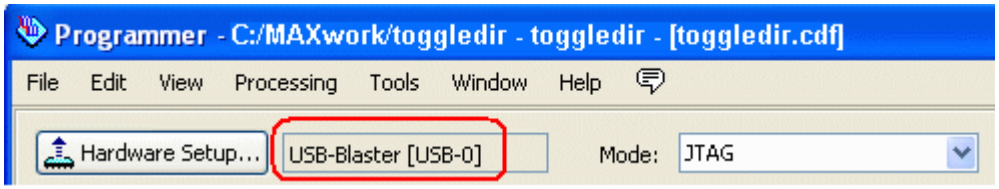


I fönstret **Hardware Setup** finns en lista över "Available hardware items". Där står USB-blastern. (Om inte så har Du kanske glömt att ansluta den?) Markera USB-blastern och klicka på **Add Hardware**.

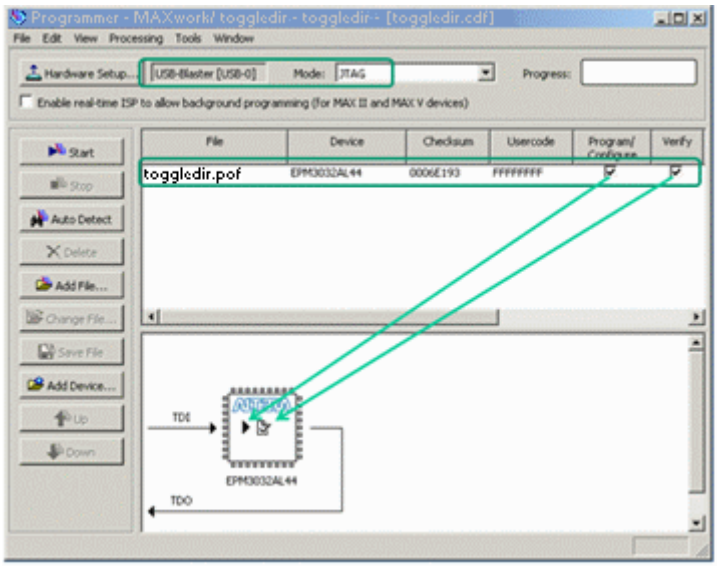
Nu skall USB-blaster vara din "Currently selected hardware". klicka på **Close**.
Quartus kommer att komma ihåg ditt val, så förmodligen behöver Du inte upprepa detta någon mer gång.



I fönstret **Programmer** ser man vilken hårdvara som valts.

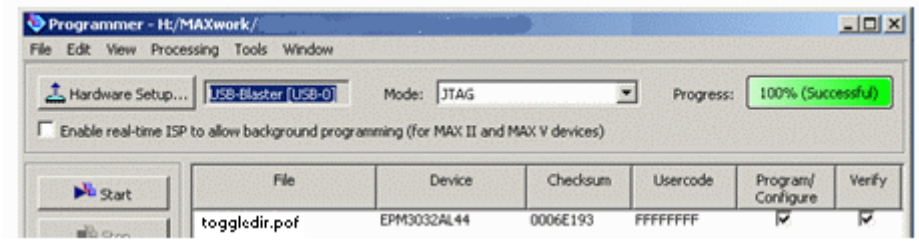


Ladda ned koden

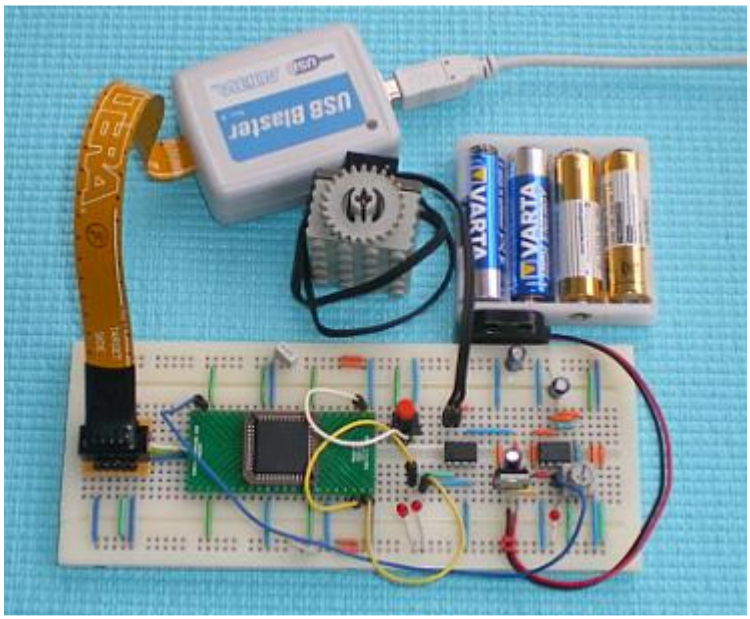


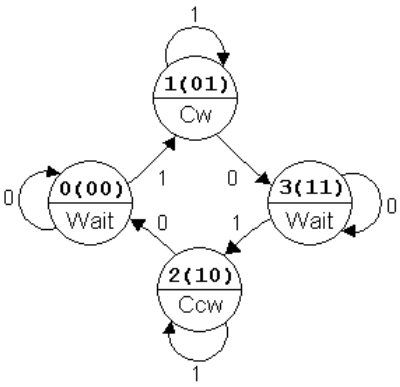
Det kompilerade projektets namn står under rutan **File**. Bocka för **Program/Configure** och **Verify**. I bilden med chippet växer det fram symboler för dina val.

Starta programmeringen genom att klicka på **Start**-knappen. Observera! Glöm inte att spänningen till labkortet ska vara på under programmeringen!



Prova funktionen!





Labassistenten har ett kopplingsdäck med en MAX-krets. Med det kan Du kontrollera att funktionen blir den förväntade.

- Quartus-programmet finns tillgängligt i en gratis web-version.

Du får gärna behålla detta häfte!