

USCALE XC800

Example application „Blinky“



Embedding Software Quality

Application “Blinky” for USCALE XC800 with SDCC

Project environment and definition



Project Environment

- Hardware is USCALE, connected via USB to a PC
- DAvE Tool to configure XC800 peripherals
- SDCC XC800 Compiler
- HiTOP IDE for USCALE

Project Definition

USCALE hosts 3 XC800 derivatives. Port 1 bit 5 is connected to a LED.

Write a program, which toggles that LED exactly every 500 milliseconds.

Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Workflow



- **Step 1: Generate an application frame using DAvE and SDCC**
- **Step 2: Generate a HiTOP project framework**
- **Step 3: Import DAvE´s application framework into HiTOP IDE**
- **Step 4: Add user code to “BLINKY” using HiTOP**

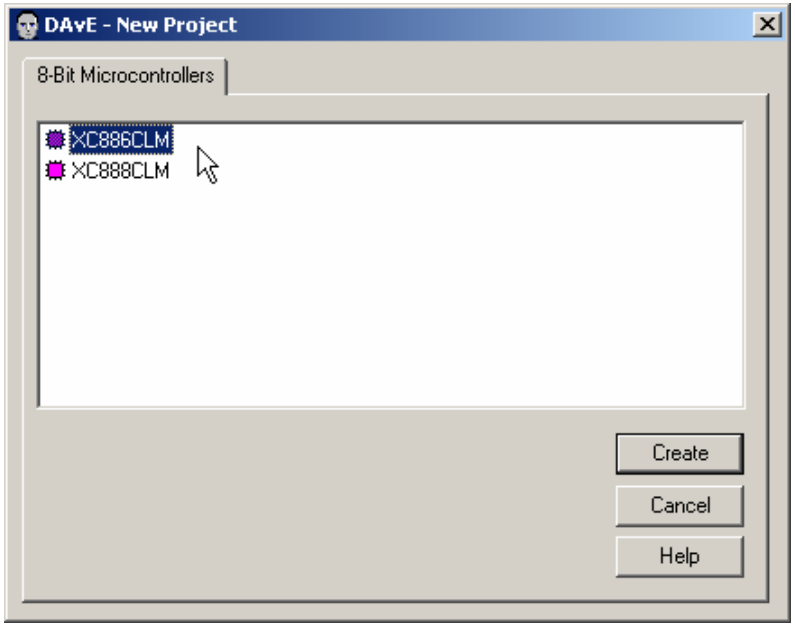
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- Create a new DAvE project
- Select XC886CLM, create...



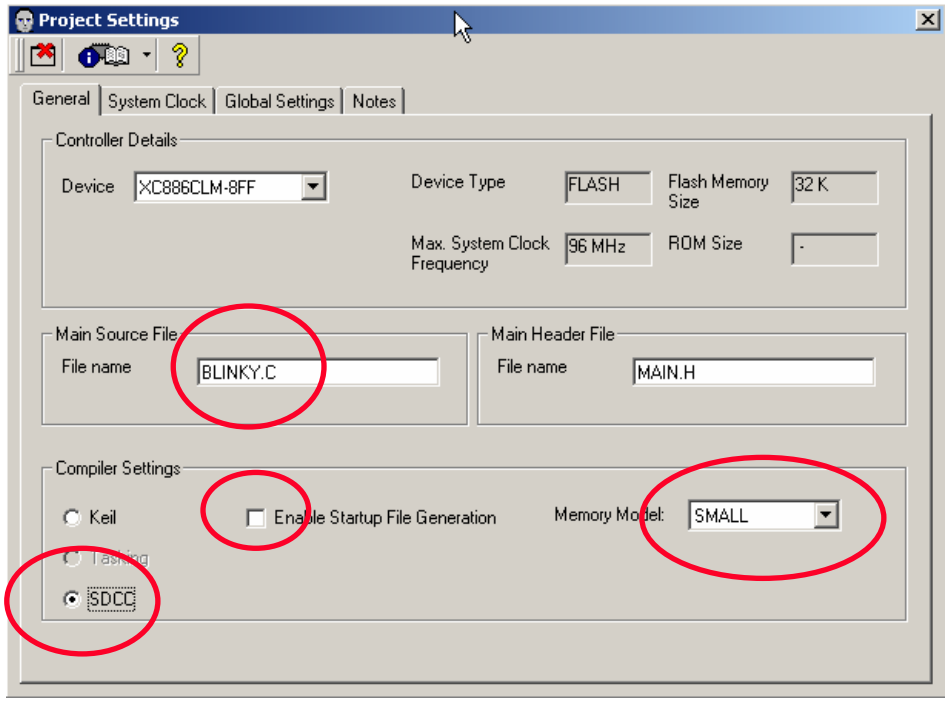


Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** General Project Settings



- Main code module is called BLINKY.C
- Compiler SDCC
- Large Memory Model
- No need for Startup File Generation

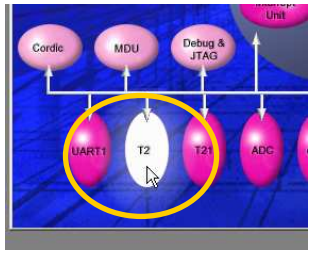
... close that window...

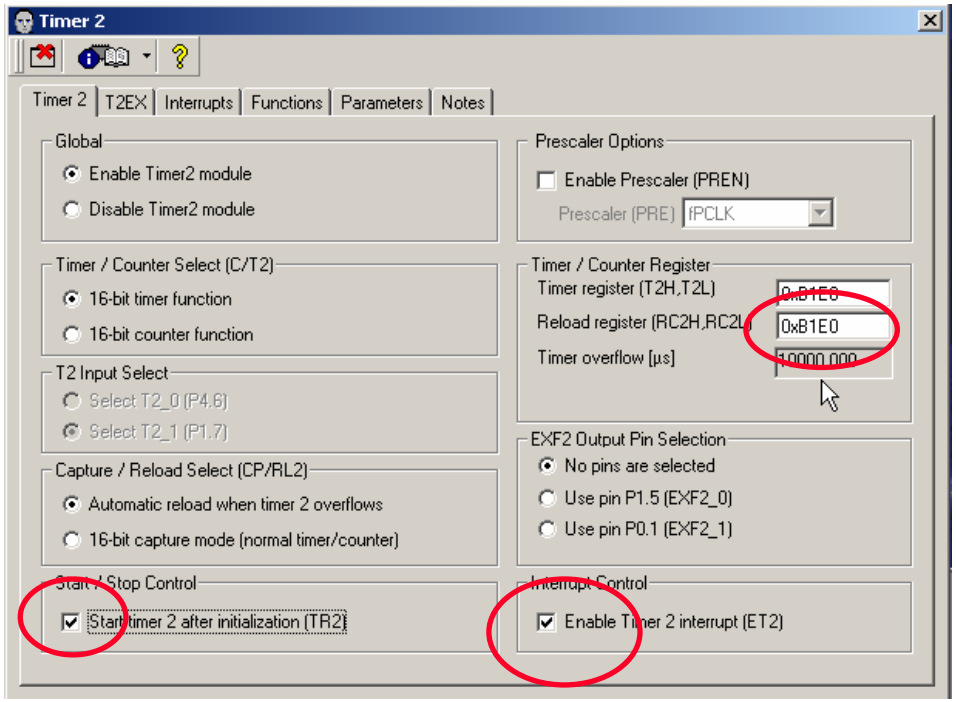
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** Setup Timer2 for generation of interrupts every 10 milliseconds





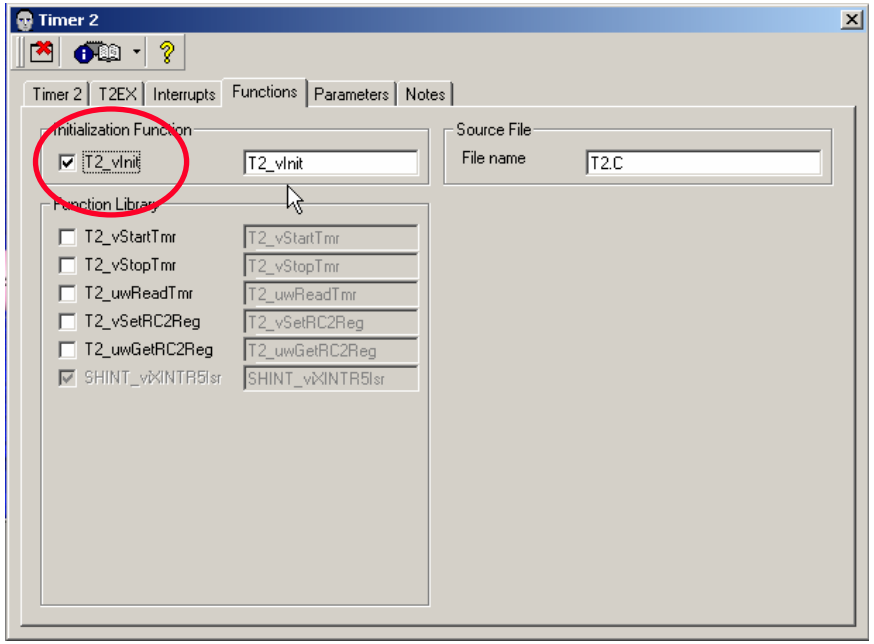
- Setup Timer 2 Reload Register to 0xB1E0
- Enable Timer 2 interrupt
- Start Timer 2 after initialization

Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** Select procedure name and functionality of interrupt initialization and interrupt service routine (ISR)



- Code procedure to initialize T2 interrupts is called T2_vInit

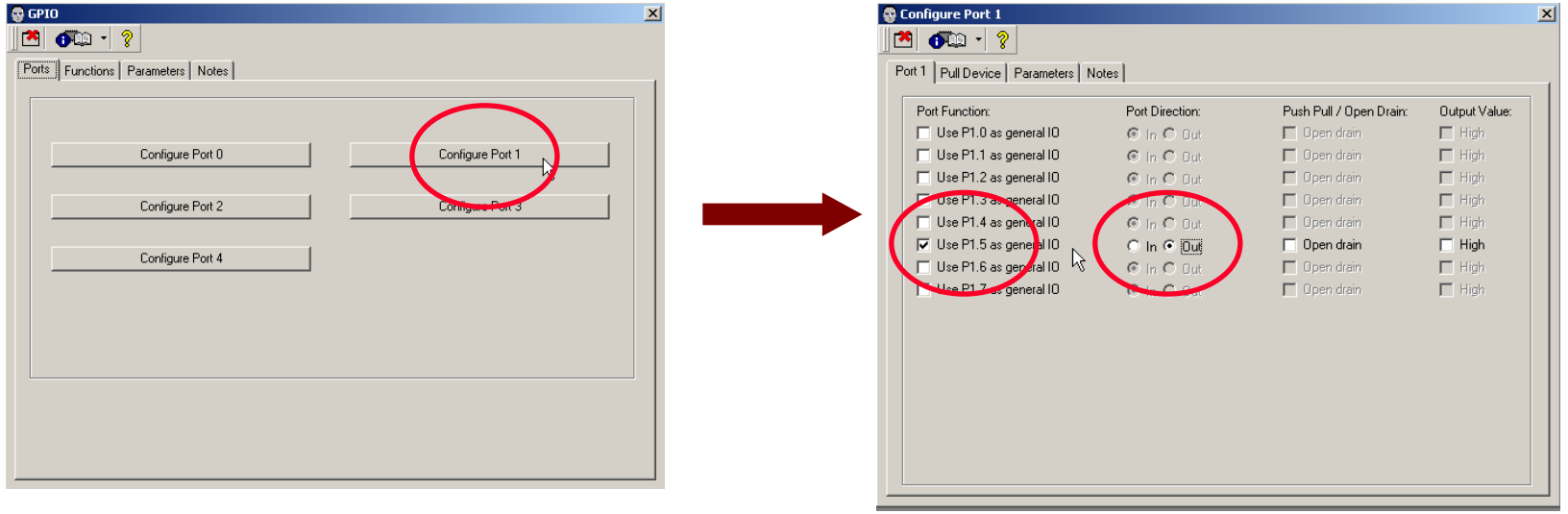
...and close that window...

Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** Initialization of the I/O port to control the LED



- Use P1.5 (LED is connected to this port pin)
- Define that port pin as output

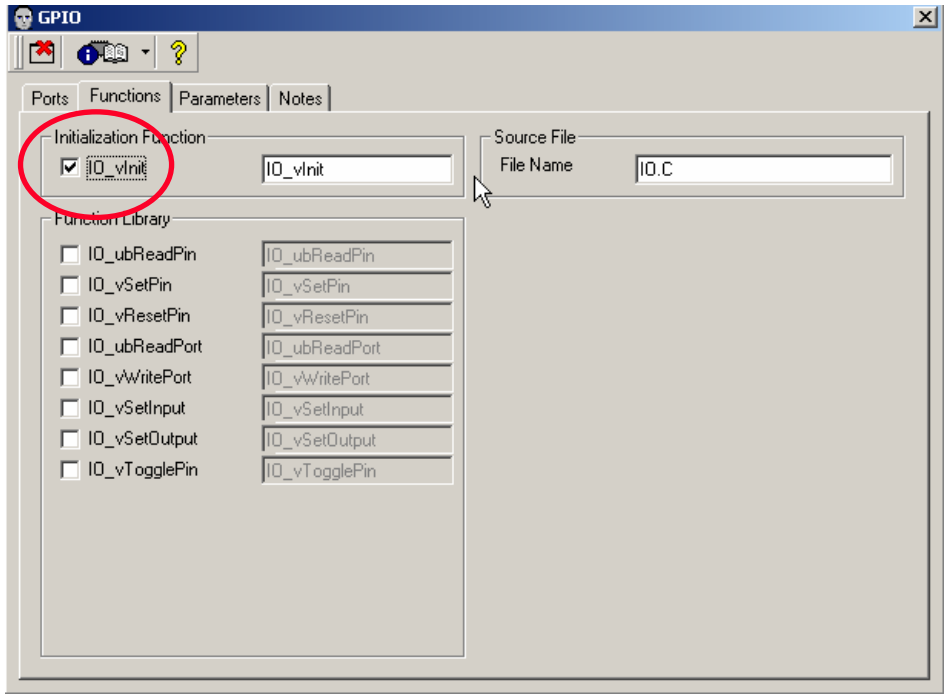
...and close that window...

Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** Select procedure name and functionality of I/O initialization.



➢ Code procedure to initialize port I/O is called IO_vInit

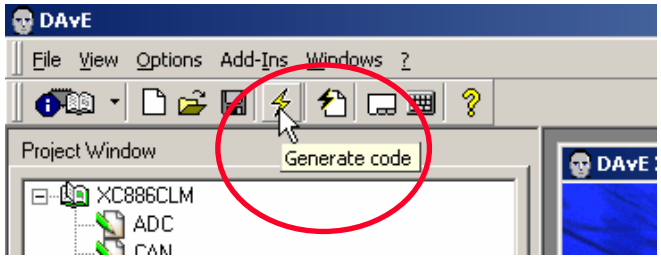
...and close that window...

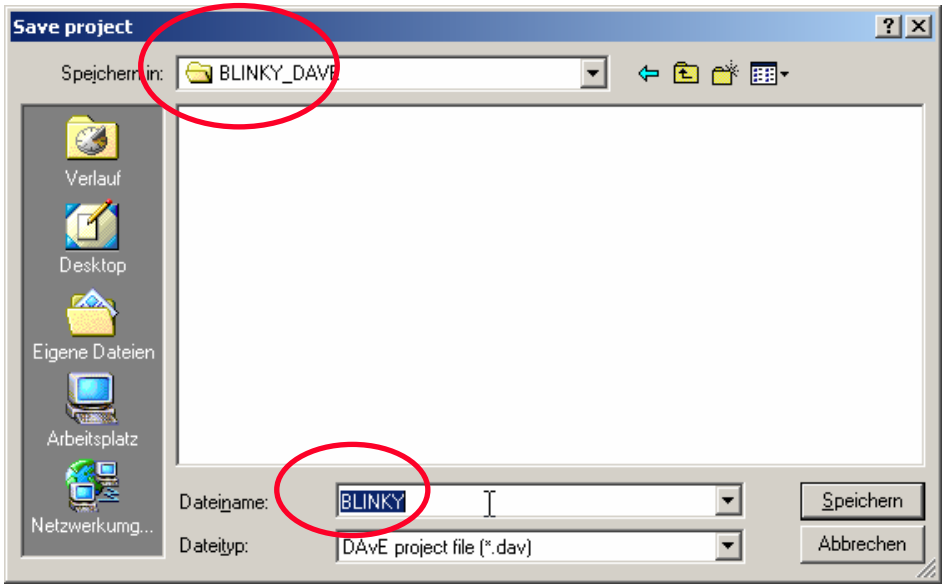
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 1: Generate an application frame using DAvE and SDCC

- **DAvE:** Generate the configured code framework





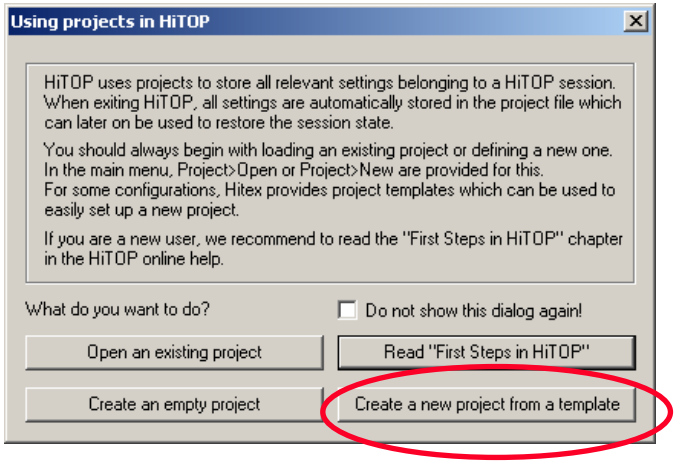
- Create DAvE project file BLINKY.dav in directory \BLINKY_DAVE and save it...
- The project framework was generated.
- Close DAvE

Embedding Software Quality

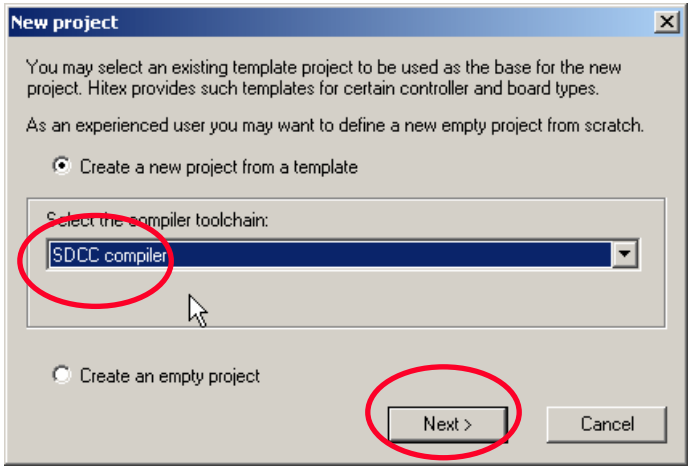
4 Steps to „BLINKY“ for USCALE

Step 2: Generate a HiTOP project framework

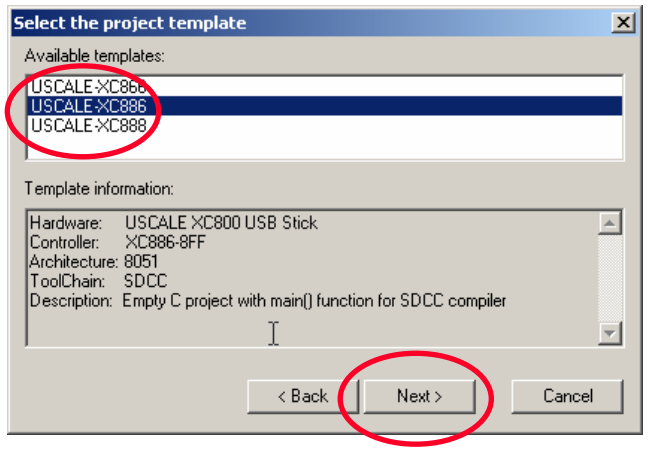
- Start HiTOP USCALE XC800



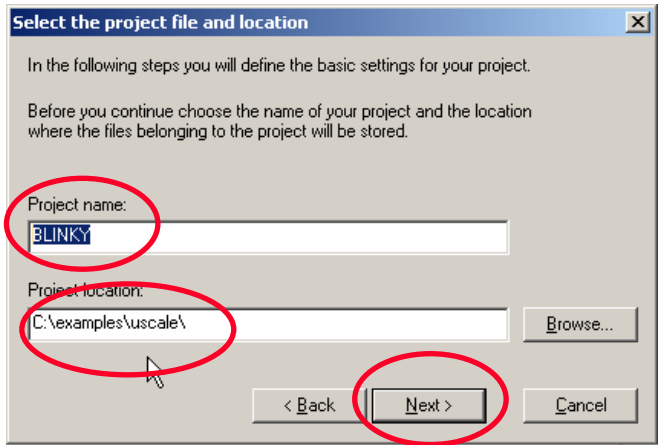
- Select the compiler



- Select the derivative



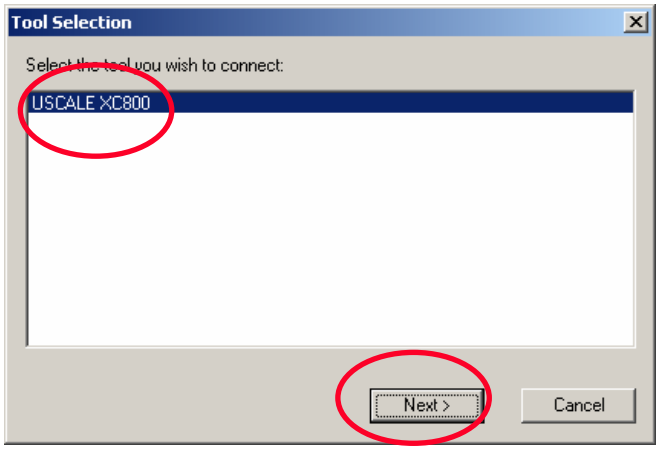
- Create HiTOP project “BLINKY.htp” in directory \example\uscale\BLINKY\



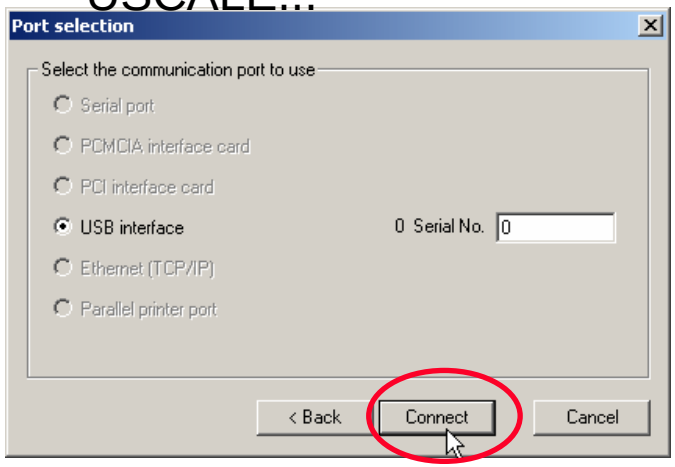
4 Steps to „BLINKY“ for USCALE

Step 2: Generate a HiTOP project framework

- Select the debugging system



- ...and connect HiTOP to USCALE...



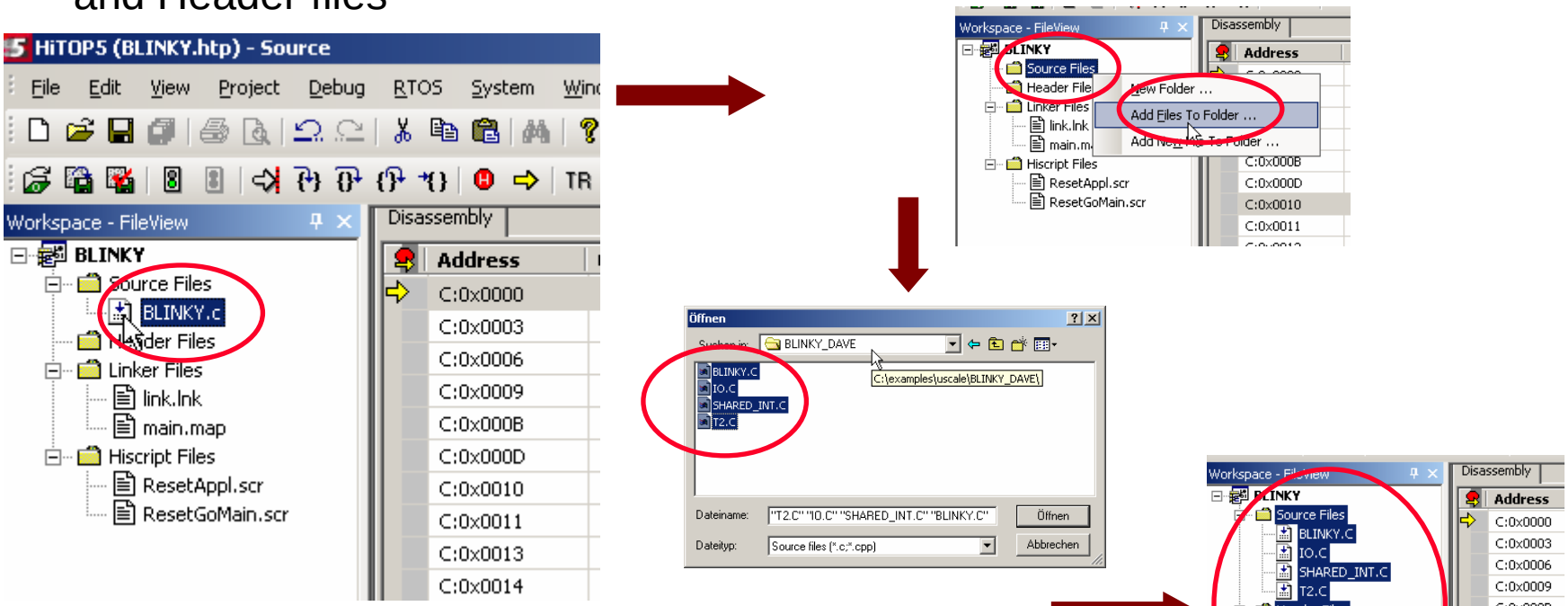
➢ HiTOP now has generated to HiTOP project framework

Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 3: Import DAvEs application framework into HiTOP IDE

- Use HiTOP´s Workspace – FileView to include DAvE´s generated Source and Header files



- Delete HiTOP´s main source module (DEL)
- Click right on “Source Files” to add source files from DAvE to folder
- Select all *.c files from directory “BLINKY_DAVE”
- Click right on “Header Files” to add header files from DAvE to folder
- Select all *.h files from directory “BLINKY_DAVE”

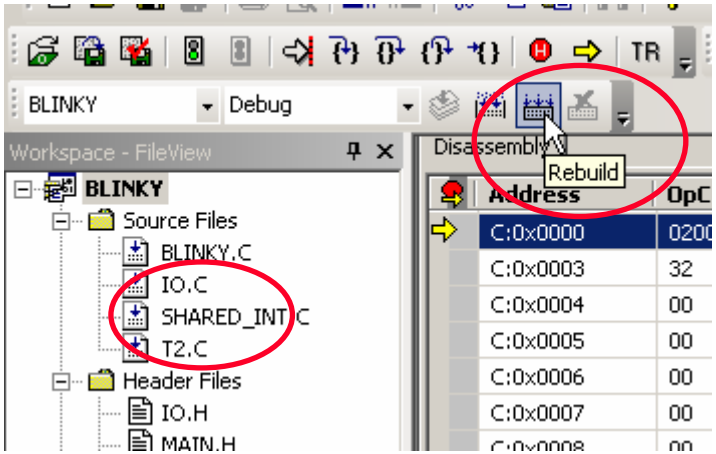
... Now import of DAvE´s application framework into HTOP is finished...

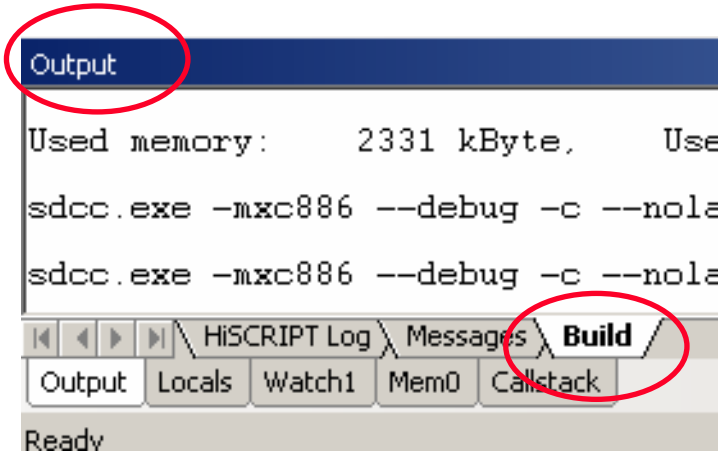
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 4: Add application code using HiTOP IDE

- Rebuild the imported application in HiTOP IDE





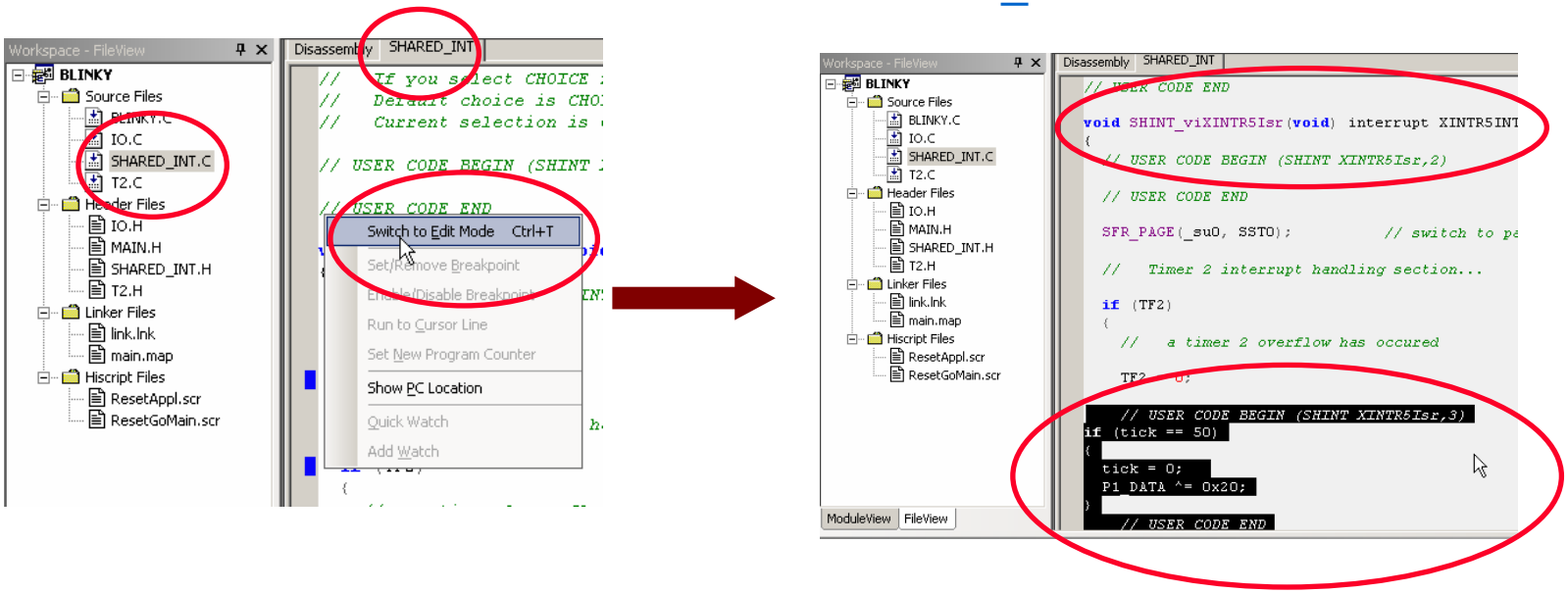
- See HiTOP´s Output window Tab “ Build” for messages generated by SDCC compile and link process....

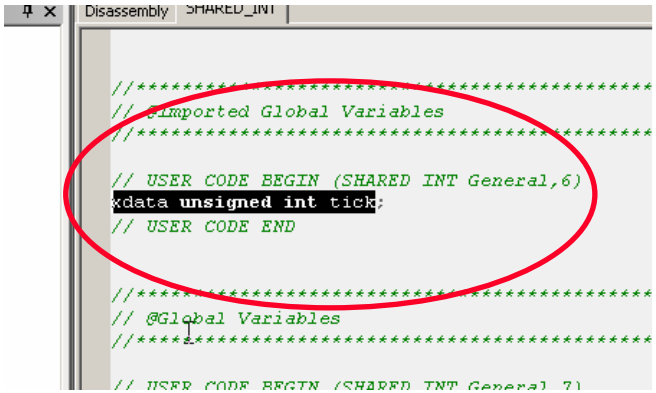
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 4: Add application code using HiTOP IDE

- Add source code to source module “SHARED_INT.C”





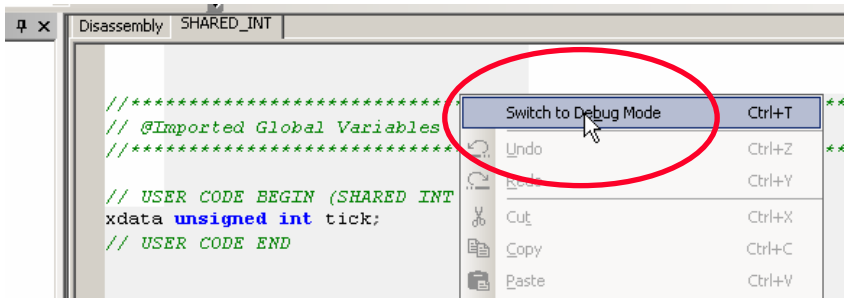
- Double click left on SHARED_INT.C to view the source file
- Click right within HiTOPs source window and switch to “Edit Mode”
- Add code to toggle LED over port 1.5 within Interrupt Service Routine
- Switch back from Edit to Debug Mode
- Rebuild and download the application

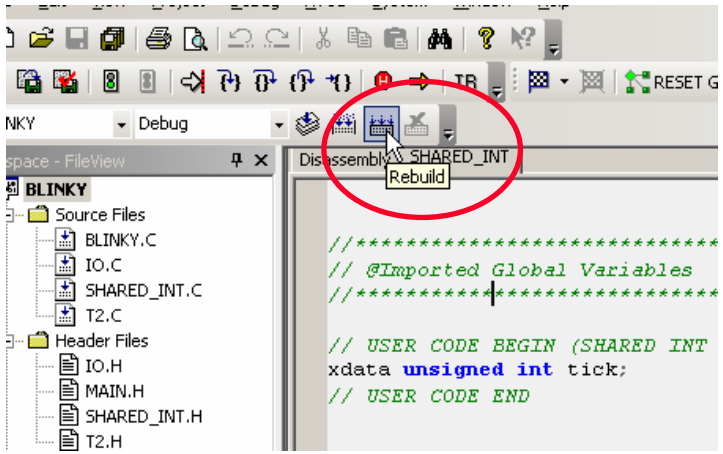
Embedding Software Quality

4 Steps to „BLINKY“ for USCALE

Step 4: Add application code using HiTOP IDE

- Rebuild application with code to toggle the LED





- Switch back from Edit to Debug Mode
- Rebuild and download the application
- Start application in HiTOP
- **If LED of USCALE blinks then application is finished**
- **If not start debugging using HiTOP.....**

Embedding Software Quality