W4 - BSPs and drivers CE 6.0

Writing BSP and drivers for Windows CE 6.0

*Windows Embedded CE*is a registered trade mark of Microsoft

Goals

- Master the cross development tools
- Bring the system on the target
- Accessing I / O
- Install kernel interrupt routines and applications
- Develop drivers and Board Support Packages

Course material

- A windows PC and an Atmel target board for two trainees
- Platform Builder for CE 6.0
- Visual Studio 2005
- A Lauterbach probe, with the Windows CE kernel awareness module
- Printed course material and labs solutions

Prerequisite

- Good knowledge of the C language
- Knowledge of application programming on Windows CE 6.0 (as described in W3 - Windows Embedded CE 6.0course) is mandatory.

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

Overview of WindowsCE

- Architecture and versions of Windows CE
- What's new in version 6.0
- Technologies and processors supported
- Embedded specific settings
- Shared source code and licensing system
- Comparison with Windows XP Embedded.
- Comparison with Windows Mobile
- Development tools for Windows Embedded CE

Windows CE architecture

- The Windows CE Kernel
- The Device Manager
- The Graphical Windowing and Events System (GWES)
- Communications services
- Timers, Watchdog timers.

- Interruptions,
  - IRQs (Interrupt ReQuest)
  - ISR (Interrupt Service Routine)
  - The IST (Interrupt Service Thread)
  - Interruption APIs
- Memory architecture.

Platform Builder in brief

- Steps to build a Windows Embedded CE platform
- Creating projects and source Workspaces
- Modules and components of the OS
- The source code configuration files
  - DIRS
  - SOURCES
  - Makefile
  - Module definition
- Build phases
  - Compile
  - Sysgen
  - Release copy
  - Make image
- Configuring Debug and Release configuration

Test and debug

- Kernel Debug
- Debug areas
- Just In Time Debugging (JIT)
- Kernel Profiler, Remote Kernel Tracker, Remote Call Profiler.
- Exception handling
- CETK Tests
- Using remote tools

Developing the Board Support Package (BSP)

- Bootloader development
- Develop the OEM Abstraction Layer (OAL)
- Configuration files

Exercise: Development of a serail line KITL
Exercise: Debug using the Lauterbach probe

Device driver development

- Introduction
- Driver types
- Stream drivers
  - interface
  - installation (static and dynamic)
- User mode drivers
- Access to physical memory
- Interrupt management
- Parameter marshalling
- Power management
- Test and debug

Exercise: Writing a fully functional button driver
- driver installation
- physical memory access
- interrupt management
- asynchronous access to the user buffer
- registration of a button as an alarm source
- writing a CETK test

Exercise: Writing a LED driver
- management of power states

Exercise: Demonstrating the use of hardware traces (captured by the Embedded Trace Macrocell) for error detection and fix, using a Lauterbach probe.