

Institute of Electronics

ECSD – Reusable Embedded Software Components



*Graphical LCD Driver based
on ECSD Components*

Project: ESCD

Start: 22.08.2008

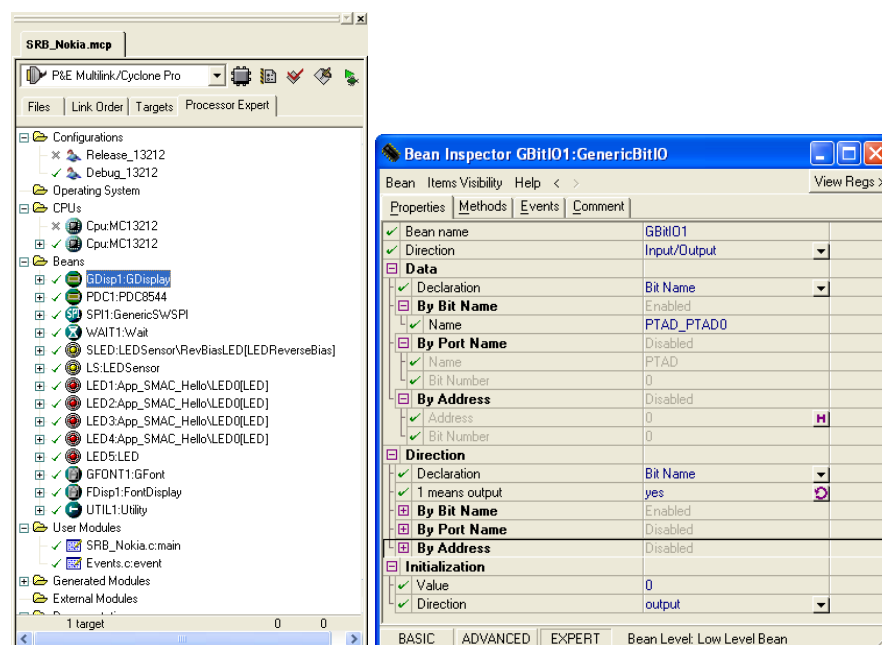
End: 31.07.2009

Project Lead:

Erich Styger, CCE

As the software amount in modern embedded systems design is increasing, the software part is getting more and more critical. In this project a foundation of reusable embedded software components have been created. This set of components has been successfully used real-time embedded systems and are used as well in student class work. An application built with this technology has won the 1st price in a worldwide contest.

To write the reusable embedded software components and drives, a scripting language together with a source code generator was used. The embedded components use properties, events and methods for the implementation of their functionality and are written in a C like scripting language. A code generator is consuming this description language and produces C code which then is compiled by a compiler. Components can inherit events, properties and methods and allow a maximum of reuse. The usage of C allows efficient code generation while still being portable between platforms.



Graphical view of embedded software components.

Contact:






































Hochschule Luzern
Technik & Architektur
Technikumstrasse 21
CH-6048 Horw

Erich Styger
T +41 41 349 33 01
erich.styger@hslu.ch

www.hslu.ch/electronics

With the usage of the UI of an IDE, the components can be easily added and configured for a complete system. The implemented hardware abstraction, together with consistent reuse of components allowed developing complex embedded applications from 8bit to 32bit in a fractional part of the usual time.

List of embedded software components (to be continued):

<i>Component</i>	<i>Category</i>	<i>Short Description</i>
 App_SMAC_Hello	Software	Sender/Receiver application
 AT25HP512	Memory	External EEPROM
 DCF77	Sensor	DCF-77 time signal decoder
 FontDisplay	HMI	Graphical Font driver
 GDisplay	HMI	Graphical Display Software Application Layer
 GenericBitIO	I/O	Generic and portable bit I/O
 GenericSPI	Communication	Generic SPI protocol implementation using bit banging
 GFont	HMI	Graphical Font Software Application Layer
 KentChLCD	HMI	Implementation of a ChLCD driver for no-power displays
 Key	HMI	Debouncing of key board switches
 LCDHTA	HMI	2x16 character LCD
 LED	HMI	A single LED
 LEDbyte	HMI	A group of 8 LED
 LEDmatrix	HMI	Display using matrix LED's
 LEDSensor	Sensor	Using a normal LED as light sensor
 LightComm	Communication	Communication Protocol using visible light and normal LED's
 LM61B	Sensor	Driver for the LM61B temperature sensor
 MaxonF2140	Actor	Driver for a Maxon DC motor
 MC13192	Transceiver	IEEE802.15.4 and ZigBee Transceiver
 MC34673	External Device	Li-Ion single coin battery charger
 MMA7260Q	Sensor	3-axis acceleration sensor
 MPR08x	Sensor/HMI	Capacitive Touch Sensor
 PDC8544	HMI	Graphical B/W LCD device
 PID_Int	Software	PID Software Control algorithm using integral operations
 QuadCounter	Sensor	Quadrature Decoder/Counter
 SimpleEvents	Software	Simple Event handling
 SMAC	Communication	IEEE802.15.4 Network stack
 SPHY	Communication	IEEE802.15.4/Zigbee Physical Layer
 SSEC	Software	Collection of multiple encryption algorithms
 Tacho	Software	Implementation of a tachometer to measure speed
 Trigger	Software	Simple triggering functionality for time triggered callbacks
 uCOS_II	RTOS	Micrium UCOS-II Operating system
 USB_CMX	Communication	USB stack
 USB_Config_CMX	Communication	USB stack configuration
 USB_HID_CMX	Communication	USB HID for keyboard/mouse/generic HID device
 Utility	Software	Safe string manipulation routines
 Wait	Software	Busy Waiting