Prototyping Physical User Interfaces

Short Course @ IT University of Göteborg

September 2003

Albrecht Schmidt
Media Informatics Group
University of Munich, Germany
http://www.medien.informatik.uni-muenchen.de/en/team/schmidt

Brief Course Outline

- **Breaking Interface Conventions?**
- **Exercise – creating a cooperative multi user game**
- **Nature and Value of Physical Prototyping**
- **Break**
- **Smart-its basics**
- **Smart-its enhanced light**
- **Lunch break**
- **Students project (afternoon) Smart-its enhanced light**
- **Smart-its enhanced light – results**
- **Building Smart-its hardware**
- **Break**
- **Developing Smart-its Software**
- **Smart-its Examples**
- **Wrap-Up**

Context Acquisition Library Structure		
Category	***Sub Categories***	***Implementation***
Architectural Frameworks	▪Attached sensing architecture. ▪Wireless single consumer architecture. ▪General wireless sensing architecture	System architectures
Hardware Library	▪Processing cores and memory units. ▪Sensor blocks ▪Communication blocks ▪Power supply blocks.	EAGLE CAD files
Software Library	▪Program Templates	Program skeletons in C and function in PIC-C
	▪Sensor drivers ▪Communication drivers ▪Timer	Drivers implemented in functions (PIC-C)
Perception Library	▪Statistical functions ▪Time domain analyis	Function in PIC-C
Backend Library	▪Serial line access ▪Network access	Variety of skeletons and functions/classes in Java, C/C++, and Visual Basic for Linux and Win32.

Software development basics

- Write a program → C-file
- Compile it for the MCU → Hex-file
- Write the executable (Hex-file) into the MCU
- Start the MCU

- Tools to start with
  - CCS PIC-Compiler
  - PIC-Start Plus

- Gerneral Intro to PIC pogramming at http://triton.cc.gatech.edu/ubicomp/394

Software Basics

Templates
- Base station (e.g. receiv2-18f.c)
- Sensor node (e.g. sensor_s-18F.c)

Drivers
- For modules or add-ons
- Implement access to sensors/actuators

Backend
- Basically reading from serial line
- Examples in java, C/C++, and Visual Basic
- For Linux and Win

Compiler CCS
- You have to know the compiler – can be tricky :)
- See www.ccsinfo.com for the newsgroups
- Programming is often dirty (e.g. global variables, …)

Compiler for MCU

Compiler CCS
- You have to know the compiler – can be tricky :)
- See www.ccsinfo.com for the newsgroups
- Programming is often dirty (e.g. global variables, …)

No in-circuit programming
- Change configuration (by physically moving processor and FRAM)
- Debugging is easier

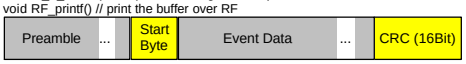
A very Simple Program

```
#include <16F876.H>
#include <STDLIB.H>
#fuses HS, NOWDT, NOBROWNOUT, NOPROTECT, NOLVP

#use delay(clock=20000000)

#use rs232(baud=115200,xmit=PIN_C2,rcv=PIN_C3)

main() {
   int x;

   x=0;

   while (1) {
        output_high(PIN_B1);
        delay_ms(200);
        output_low(PIN_B1);
        printf("counter: %i\n", x);
        x=x+1;
  }
}
```


Software - Files

core18F252.c
- Defines according to schematic,
- I2C initialization
- funtion to control LED

bim2rf3.c
- Simple RF protocol implementation, e.g.
- void reset_rf_buffer() // clear buffer, use before printf
- void to_rf_buffer(char c) // as first argument in printf
- void RF_printf() // print the buffer over RF



fr24c64.c
- functions to use the FRAM chip
- Read and write to memory

18F252.h
- Standard h-file for the PIC used in the core board
- Provided with the compiler

Lets have a look at some files…

Debugging

A Pain…
- Can be hardware, communication, system software, or application software

Had influence on design choices…
- Change configuration
  (by physically moving processor and FRAM)
- Debugging is easier that way!