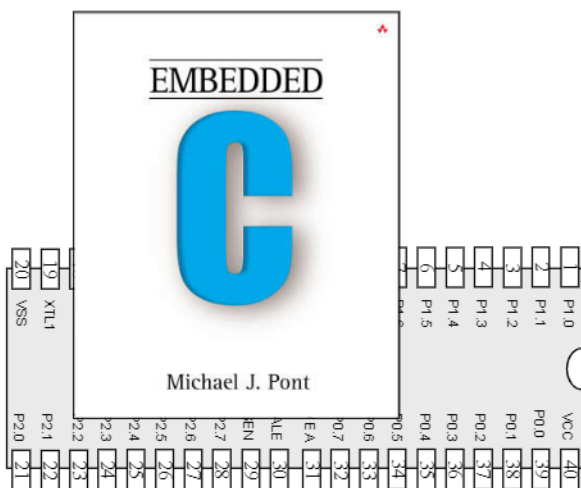


Programming Embedded Systems I

A 10-week course, using C



Michael J. Pont
University of Leicester

[v1.2]

Copyright © Michael J. Pont, 2002-2003

This document may be freely distributed and copied, provided that copyright notice at the foot of each OHP page is clearly visible in all copies.

Seminar 1: “Hello, Embedded World”	1
Overview of this seminar	2
Overview of this course	3
By the end of the course ...	4
Main course textbook	5
Why use C?	6
Pre-requisites!	7
The 8051 microcontroller	8
The “super loop” software architecture	9
Strengths and weaknesses of “super loops”	10
Example: Central-heating controller	11
Reading from (and writing to) port pins	12
SFRs and ports	13
SFRs and ports	14
Creating and using sbit variables	15
Example: Reading and writing bytes	16
Creating “software delays”	17
Using the performance analyzer to test software delays	18
Strengths and weaknesses of software-only delays	19
Preparation for the next seminar	20

Seminar 2: Basic hardware foundations (resets, oscillators and port I/O)	21
Review: The 8051 microcontroller	22
Review: Central-heating controller	23
Overview of this seminar	24
Oscillator Hardware	25
How to connect a crystal to a microcontroller	27
Oscillator frequency and machine cycle period	28
Keep the clock frequency as low as possible	29
Stability issues	30
Improving the stability of a crystal oscillator	31
Overall strengths and weaknesses	32
Reset Hardware	34
More robust reset circuits	35
Driving DC Loads	36
Use of pull-up resistors	38
Driving a low-power load without using a buffer	39
Using an IC Buffer	40
Example: Buffering three LEDs with a 74HC04	41
What is a multi-segment LED?	42
Driving a single digit	43
Preparation for the next seminar	44

Seminar 3: Reading Switches

45

Introduction	46
Review: Basic techniques for reading from port pins	47
Example: Reading and writing bytes (review)	48
Example: Reading and writing bits (simple version)	49
Example: Reading and writing bits (generic version)	51
The need for pull-up resistors	56
The need for pull-up resistors	57
The need for pull-up resistors	58
Dealing with switch bounce	59
Example: Reading switch inputs (basic code)	61
Example: Counting goats	68
Conclusions	74
Preparation for the next seminar	75

Seminar 4: Adding Structure to Your Code**77**

Introduction	78
Object-Oriented Programming with C	79
Example of “O-O C”	82
The Project Header (Main.H)	85
The Port Header (Port.H)	92
Re-structuring a “Hello World” example	96
Example: Re-structuring the Goat-Counting Example	104
Preparation for the next seminar	114

Seminar 5: Meeting Real-Time Constraints**115**

Introduction	116
Creating “hardware delays”	118
The TCON SFR	119
The TMOD SFR	120
Two further registers	121
Example: Generating a precise 50 ms delay	122
Example: Creating a portable hardware delay	126
The need for ‘timeout’ mechanisms - example	129
Creating loop timeouts	130
Example: Testing loop timeouts	132
Example: A more reliable switch interface	134
Creating hardware timeouts	135
Conclusions	137
Preparation for the next seminar	138

Seminar 6: Creating an Embedded Operating System**139**

Introduction	140
Timer-based interrupts (the core of an embedded OS)	144
The interrupt service routine (ISR)	145
Automatic timer reloads	146
Introducing sEOS	147
Introducing sEOS	148
Tasks, functions and scheduling	153
Setting the tick interval	154
Saving power	157
Using sEOS in your own projects	158
Is this approach portable?	159
Example: Milk pasteurization	160
Conclusions	174
Preparation for the next seminar	175

Seminar 7: Multi-State Systems and Function Sequences**177**

Introduction	178
Implementing a Multi-State (Timed) system	180
Example: Traffic light sequencing	181
Example: Animatronic dinosaur	189
Implementing a Multi-State (Input/Timed) system	195
Example: Controller for a washing machine	197
Conclusions	208
Preparation for the next seminar	209

Seminar 8: Using the Serial Interface	211
Overview of this seminar	212
What is 'RS-232'?	213
Basic RS-232 Protocol	214
Asynchronous data transmission and baud rates	215
RS-232 voltage levels	216
The software architecture	217
Overview	218
Using the on-chip U(S)ART for RS-232 communications	219
Serial port registers	220
Baud rate generation	221
Why use 11.0592 MHz crystals?	222
PC Software	223
What about printf()?	224
RS-232 and 8051: Overall strengths and weaknesses	225
Example: Displaying elapsed time on a PC	226
Example: Data acquisition	235
Conclusions	239
Preparation for the next seminar	240