## CONTENTS

Preface xxv  
Acknowledgments xxxi

### 1 Overview of Embedded System  
1.1 Introduction  
1.1.1 Definition of an embedded system  
1.1.2 Example systems  
1.2 System design requirements  
1.3 Embedded SoPC systems  
1.3.1 Basic development flow  
1.4 Book organization  
1.5 Bibliographic notes  

### PART I BASIC DIGITAL CIRCUITS DEVELOPMENT

### 2 Gate-level Combinational Circuit  
2.1 Overview of VHDL  
2.2 General description  
2.2.1 Basic lexical rules  
2.2.2 Library and package  
2.2.3 Entity declaration  

## CONTENTS

2.2.4 Data type and operators 13
2.2.5 Architecture body 14
2.2.6 Code of a 2-bit comparator 15
2.3 Structural description 16
2.4 Testbench 18
2.5 Bibliographic notes 19
2.6 Suggested experiments 20
  2.6.1 Code for gate-level greater-than circuit 20
  2.6.2 Code for gate-level binary decoder 20

3 Overview of FPGA and EDA Software 21
  3.1 FPGA 21
    3.1.1 Overview of a general FPGA device 21
    3.1.2 Overview of the Altera Cyclone II devices 23
  3.2 Overview of the Altera DE1 and DE2 boards 26
  3.3 Development flow 26
  3.4 Overview of Quartus II 29
  3.5 Short tutorial of Quartus II 31
    3.5.1 Create the design project 32
    3.5.2 Create a testbench and perform the RTL simulation 37
    3.5.3 Compile the project 37
    3.5.4 Perform timing analysis 39
    3.5.5 Program the FPGA device 39
  3.6 Short tutorial on the ModelSim HDL simulator 42
  3.7 Bibliographic notes 47
  3.8 Suggested experiments 47
    3.8.1 Gate-level greater-than circuit 47
    3.8.2 Gate-level binary decoder 47

4 RT-level Combinational Circuit 49
  4.1 RT-level components 49
    4.1.1 Relational operators 51
    4.1.2 Arithmetic operators 51
    4.1.3 Other synthesis-related VHDL constructs 52
    4.1.4 Summary 54
  4.2 Routing circuit with concurrent assignment statements 55
    4.2.1 Conditional signal assignment statement 55
    4.2.2 Selected signal assignment statement 58
  4.3 Modeling with a process 60
    4.3.1 Process 60
    4.3.2 Sequential signal assignment statement 60
4.4 Routing circuit with if and case statements 61
   4.4.1 If statement 61
   4.4.2 Case statement 63
   4.4.3 Comparison to concurrent statements 64
   4.4.4 Unintended memory 66
4.5 Constants and generics 67
   4.5.1 Constants 67
   4.5.2 Generics 68
4.6 Design examples 69
   4.6.1 Hexadecimal digit to seven-segment LED decoder 69
   4.6.2 Sign-magnitude adder 72
   4.6.3 Barrel shifter 74
   4.6.4 Simplified floating-point adder 75
4.7 Bibliographic notes 80
4.8 Suggested experiments 80
   4.8.1 Multi-function barrel shifter 80
   4.8.2 Dual-priority encoder 81
   4.8.3 BCD incrementor 81
   4.8.4 Floating-point greater-than circuit 81
   4.8.5 Floating-point and signed integer conversion circuit 81
   4.8.6 Enhanced floating-point adder 82

5 Regular Sequential Circuit 83
5.1 Introduction 83
   5.1.1 D FF and register 84
   5.1.2 Synchronous system 84
   5.1.3 Code development 85
5.2 HDL code of the basic storage elements 85
   5.2.1 D FF 86
   5.2.2 Register 89
   5.2.3 Register file 89
   5.2.4 SRAM 93
5.3 Simple design examples 93
   5.3.1 Shift register 94
   5.3.2 Binary counter and variant 95
5.4 Testbench for sequential circuits 98
5.5 Timing analysis 101
   5.5.1 Timing parameters 101
   5.5.2 Timing considerations in Quartus II 103
5.6 Case study 104
   5.6.1 Stopwatch 104
   5.6.2 FIFO buffer 108
## CONTENTS

5.7 Cyclone II device embedded memory module 113  
5.7.1 Overview of memory options of DE1 board 113  
5.7.2 Overview of embedded M4K module 114  
5.7.3 Methods to incorporate embedded memory module 114  
5.7.4 HDL module to infer synchronous single-port RAM 116  
5.7.5 HDL module to infer synchronous simple dual-port RAM 117  
5.7.6 HDL module to infer synchronous true dual-port RAM 119  
5.7.7 HDL module to infer synchronous ROM 120  
5.7.8 FIFO buffer revisited 121  
5.8 Bibliographic notes 122  
5.9 Suggested experiments 122  
5.9.1 Programmable square wave generator 122  
5.9.2 Pulse width modulation circuit 122  
5.9.3 Rotating square circuit 123  
5.9.4 Heartbeat circuit 123  
5.9.5 Rotating LED banner circuit 123  
5.9.6 Enhanced stopwatch 123  
5.9.7 FIFO with data width conversion 124  
5.9.8 Stack 124  
5.9.9 ROM-based sign-magnitude adder 124  
5.9.10 ROM-based temperature conversion 124  

6 FSM 127  
6.1 Introduction 127  
6.1.1 Mealy and Moore outputs 128  
6.1.2 FSM representation 128  
6.2 FSM code development 131  
6.3 Design examples 134  
6.3.1 Rising-edge detector 134  
6.3.2 Debouncing circuit 138  
6.3.3 Testing circuit 142  
6.4 Bibliographic notes 144  
6.5 Suggested experiments 144  
6.5.1 Dual-edge detector 144  
6.5.2 Alternative debouncing circuit 144  
6.5.3 Parking lot occupancy counter 144  

7 FSMD 147  
7.1 Introduction 147  
7.1.1 Single RT operation 148  
7.1.2 ASMD chart 148
7.1.3 Decision box with a register 150
7.2 Code development of an FSMD 153
  7.2.1 Debouncing circuit based on RT methodology 153
  7.2.2 Code with explicit data path components 153
  7.2.3 Code with implicit data path components 156
  7.2.4 Comparison 157
7.3 Design examples 159
  7.3.1 Fibonacci number circuit 159
  7.3.2 Division circuit 162
  7.3.3 Binary-to-BCD conversion circuit 165
  7.3.4 Period counter 168
  7.3.5 Accurate low-frequency counter 171
7.4 Bibliographic notes 174
7.5 Suggested experiments 174
  7.5.1 Alternative debouncing circuit 174
  7.5.2 BCD-to-binary conversion circuit 175
  7.5.3 Fibonacci circuit with BCD I/O: design approach 1 175
  7.5.4 Fibonacci circuit with BCD I/O: design approach 2 175
  7.5.5 Auto-scaled low-frequency counter 176
  7.5.6 Reaction timer 176
  7.5.7 Babbage difference engine emulation circuit 177

PART II BASIC NIOS II SOFTWARE DEVELOPMENT

8 Nios II Processor Overview 181
  8.1 Introduction 181
  8.2 Register file and ALU 183
    8.2.1 Register file 183
    8.2.2 ALU 184
  8.3 Memory and I/O organization 184
    8.3.1 Nios II memory interface 184
    8.3.2 Overview of memory hierarchy 184
    8.3.3 Virtual memory 184
    8.3.4 Memory protection 185
    8.3.5 Cache memory 186
    8.3.6 Tightly coupled memory 186
    8.3.7 I/O organization 186
    8.3.8 Interconnect structure 187
  8.4 Exception and interrupt handler 187
  8.5 JTAG debug module 187
  8.6 Bibliographic notes 187
  8.7 Suggested projects 188
8.7.1 Comparison of Nios II and MIPS

9 Nios II System Derivation and Low-Level Access

9.1 Development flow revisited
9.1.1 Hardware development
9.1.2 Software development
9.1.3 Flashing-LED system

9.2 Nios II hardware generation tutorial
9.2.1 Create a hardware project in Quartus II
9.2.2 Create a Nios II system and generate HDL codes
9.2.3 Create a top-level HDL file that instantiates the Nios II system
9.2.4 Compiling and programming

9.3 Nios II SBT GUI tutorial
9.3.1 Create BSP library
9.3.2 Configure the BSP using BSP Editor
9.3.3 Create user application directory and add application files
9.3.4 Build and run software
9.3.5 Check code size

9.4 System id core for hardware-software consistency

9.5 Direct low-level I/O access
9.5.1 Review of C pointer
9.5.2 C pointer for I/O register

9.6 Robust low-level I/O access
9.6.1 system.h
9.6.2 alt_types.h
9.6.3 io.h

9.7 Some C techniques for low-level I/O operations
9.7.1 Bit manipulation
9.7.2 Packing and unpacking

9.8 Software development
9.8.1 Basic embedded program architecture
9.8.2 Main program and task routines

9.9 Bibliographic notes

9.10 Suggested experiments
9.10.1 Chasing LED circuit
9.10.2 Collision LED circuit
9.10.3 Pulse width modulation circuit
9.10.4 Rotating square circuit
9.10.5 Heartbeat circuit

9.11 Complete program listing
10 Predesigned Nios II I/O Peripherals

10.1 Overview 217
10.2 PIO core 218
   10.2.1 Configuration 218
   10.2.2 Register map 221
   10.2.3 Visible register 222
10.3 JTAG UART core 222
   10.3.1 Configuration 222
   10.3.2 Register map 223
10.4 Internal timer core 224
   10.4.1 Configuration 224
   10.4.2 Register map 225
10.5 Enhanced flashing-LED Nios II system 226
   10.5.1 SOPC design 226
   10.5.2 Top-level HDL file 230
10.6 Software development of enhanced flashing-LED system 232
   10.6.1 Introduction to device driver 232
   10.6.2 Program structure of the enhanced flashing-LED system 233
   10.6.3 Main program 233
   10.6.4 Function naming convention 234
10.7 Device driver routines 234
   10.7.1 Driver for PIO peripherals 234
   10.7.2 JTAG UART 237
   10.7.3 Timer 238
10.8 Task routines 239
   10.8.1 The flashsys_init_v1() function 239
   10.8.2 The sw_get_command_v1() function 239
   10.8.3 The jtaguart_disp_msg_v1() function 240
   10.8.4 The ssseg_disp_msg_v1() function 240
   10.8.5 The led_flash_v1() function 241
10.9 Software construction and testing 242
10.10 Bibliographic notes 242
10.11 Suggested experiments 242
   10.11.1 “Uptime” feature in flashing-LED system 242
   10.11.2 Counting with different timer mode 243
   10.11.3 JTAG UART input 243
   10.11.4 Enhanced collision LED circuit 243
   10.11.5 Rotating LED banner circuit 244
   10.11.6 Enhanced stopwatch 244
   10.11.7 Parking lot occupancy counter 244
   10.11.8 Reaction timer with pushbutton switch control 244
   10.11.9 Reaction timer with keyboard control 244
10.11.10 Communication with serial port 244
10.12 Complete program listing 246

11 Predesigned Nios II I/O Drivers and HAL API 255

11.1 Overview of HAL 255
  11.1.1 Desktop-like and barebone embedded systems 256
  11.1.2 HAL paradigm 257
  11.1.3 Device classes 258
  11.1.4 HAL-compliant device drivers 259
  11.1.5 The _regs.h file 259
  11.1.6 HAL-based initialization sequence 260

11.2 BSP 261
  11.2.1 Overview 261
  11.2.2 BSP file structure 261
  11.2.3 BSP configuration 261

11.3 HAL-based flashing-LED program 265
  11.3.1 Functions using generic I/O devices 265
  11.3.2 Functions using non-generic I/O devices 267
  11.3.3 Initialization routine and main program 268
  11.3.4 Software construction and testing 269

11.4 Device driver consideration 270
  11.4.1 I/O access methods 270
  11.4.2 Comparisons 271
  11.4.3 Device drivers in this book 272

11.5 Bibliographic notes 273
11.6 Suggested experiments 273
  11.6.1 “Uptime” feature in flashing-LED system 273
  11.6.2 Enhanced collision LED circuit 274
  11.6.3 Parking lot occupancy counter 274
  11.6.4 Reaction timer with keyboard control 274
  11.6.5 Digital alarm clock 274

11.7 Complete program listing 275

12 Interrupt and ISR 277

12.1 Interrupt processing in the HAL framework 277
  12.1.1 Overview 278
  12.1.2 Interrupt controller of the Nios II processor 278
  12.1.3 Top-level exception handler 279
  12.1.4 Interrupt service routines 280

12.2 Interrupt-based flashing-LED program 280
  12.2.1 Interrupt of timer core 281
12.2.2 Driver of timer core 281
12.2.3 ISR version 1 282
12.2.4 ISR version 2 284
12.3 Interrupt and scheduling 285
  12.3.1 Scheduling 285
  12.3.2 Performance 287
12.4 Bibliographic notes 288
12.5 Suggested experiments 288
  12.5.1 Flashing-LED system with pushbutton switch ISR 288
  12.5.2 ISR-driven flashing-LED system 288
  12.5.3 “Uptime” feature in flashing-LED system 289
  12.5.4 Reaction timer with keyboard control 289
  12.5.5 Digital alarm clock 289
12.6 Complete program listing 290

PART III CUSTOM I/O PERIPHERAL DEVELOPMENT

13 Custom I/O Peripheral with PIO Cores 297
  13.1 Introduction 297
  13.2 Integration of division circuit to a Nios II system 298
    13.2.1 PIO modules 298
    13.2.2 Integration 299
  13.3 Testing 299
  13.4 Suggested experiments 302
    13.4.1 Division core ISR 302
    13.4.2 Division core with eight-bit data 302
    13.4.3 Division core with 64-bit data 303
    13.4.4 Fibonacci number circuit 303
    13.4.5 Period counter 303

14 Avalon Interconnect and SOPC Component 305
  14.1 Introduction 305
  14.2 Avalon MM interface 309
    14.2.1 Avalon MM slave interface signals 309
    14.2.2 Avalon MM slave interface properties 310
    14.2.3 Avalon MM slave timing 310
  14.3 System interconnect fabric for Avalon interface 313
  14.4 SOPC I/O component wrapping circuit 315
    14.4.1 Interface I/O buffer 315
    14.4.2 Memory alignment 318
    14.4.3 Output decoding from an Avalon MM master 318
    14.4.4 Input multiplexing to an Avalon MM master 320
14.4.5 Practical consideration
14.5 SOPC component construction tutorial
14.5.1 Avalon interfaces
14.5.2 Register map
14.5.3 Wrapped division circuit
14.5.4 SOPC component creation
14.5.5 SOPC component instantiation

14.6 Testing
14.7 Bibliographic notes
14.8 Suggested experiments
14.8.1 Division core ISR
14.8.2 Alternative buffering scheme for the division core
14.8.3 Division core with eight-bit data
14.8.4 Division core with 64-bit data
14.8.5 Fibonacci number circuit
14.8.6 Period counter

15 SRAM and SDRAM Controllers
15.1 Memory resources of DE1 board
15.2 Brief overview of timing and clock management
15.2.1 Clock distribution network
15.2.2 Timing consideration of off-chip access
15.2.3 PLL
15.3 Overview of SRAM
15.3.1 SRAM cell
15.3.2 Basic organization
15.3.3 Timing
15.3.4 IS61LV25616AL SRAM device
15.4 SRAM controller IP core
15.4.1 Avalon interfaces
15.4.2 Controller circuit
15.4.3 SOPC component creation
15.5 Overview of DRAM
15.5.1 DRAM cell
15.5.2 Basic DRAM organization
15.5.3 DRAM timing
15.6 Overview of SDRAM
15.6.1 Basic SDRAM organization
15.6.2 SDRAM timing
15.6.3 ICSI IS42S16400 SDRAM device
15.7 SDRAM controller and PLL
15.7.1 Basic SDRAM controller
CONTENTS

15.7.2 SDRAM controller IP core 364
15.7.3 SOPC PLL IP core 365

15.8 Testing system 367
  15.8.1 Testing hardware configuration 367
  15.8.2 Testing software 372

15.9 Bibliographic notes 375

15.10 Suggested experiments 375
  15.10.1 SRAM controller without I/O register 375
  15.10.2 SRAM controller speed test 375
  15.10.3 SRAM controller with Avalon MM tristate interface 376
  15.10.4 SDRAM controller clock skew test 376
  15.10.5 Memory performance comparison 376
  15.10.6 Effect of cache memory 376
  15.10.7 SDRAM controller from scratch 376

15.11 Complete program listing 377

16 PS2 Keyboard and Mouse 379

16.1 Introduction 379
16.2 PS2 receiving subsystem 380
  16.2.1 PS2-device-to-host communication protocol 380
  16.2.2 Design and code 381
16.3 PS2 transmitting subsystem 384
  16.3.1 Host-to-PS2-device communication protocol 384
  16.3.2 Design and code 385
16.4 Complete PS2 system 389
16.5 PS2 controller IP core development 391
  16.5.1 Avalon interfaces 391
  16.5.2 Register map 391
  16.5.3 Wrapped PS2 system 392
  16.5.4 SOPC component creation 393
16.6 PS2 driver 394
  16.6.1 Register map 394
  16.6.2 Write routines 394
  16.6.3 Read routines 394
16.7 Keyboard driver 396
  16.7.1 Overview of the scan code 396
  16.7.2 Interaction with host 397
  16.7.3 Driver routines 397
16.8 Mouse driver 401
  16.8.1 Overview of PS2 mouse protocol 401
  16.8.2 Interaction with host 402
  16.8.3 Driver routines 403
16.9 Test 405
16.10 Use of book’s custom IP cores 407
   16.10.1 IP core files 407
   16.10.2 Comprehensive Nios II testing system 408
16.11 Bibliographic notes 415
16.12 Suggested experiments 415
   16.12.1 PS2 receiving subsystem with watchdog timer 415
   16.12.2 Software receiving FIFO 415
   16.12.3 Software PS2 controller 415
   16.12.4 Keyboard-controlled LED flashing circuit 416
   16.12.5 Enhanced keyboard driver routine I 416
   16.12.6 Enhanced keyboard driver routine II 416
   16.12.7 Remote-mode mouse driver 416
   16.12.8 Scroll-wheel mouse driver 416
16.13 Complete program listing 417

17 VGA Controller 431
17.1 Introduction 431
   17.1.1 Basic operation of a CRT 431
   17.1.2 VGA port of the DE1 board 433
   17.1.3 Video controller 434
17.2 VGA synchronization 435
   17.2.1 Horizontal synchronization 436
   17.2.2 Vertical synchronization 437
   17.2.3 Timing calculation of VGA synchronization signals 438
   17.2.4 HDL implementation 438
17.3 SRAM-based video RAM controller 441
   17.3.1 Overview of video memory 441
   17.3.2 Memory consideration of DE1 board 442
   17.3.3 Ad hoc SRAM controller 442
   17.3.4 HDL code 446
17.4 Palette circuit 450
17.5 Video controller IP core development 451
   17.5.1 Complete video controller 451
   17.5.2 Avalon interfaces 451
   17.5.3 Register map 451
   17.5.4 Wrapped video controller 452
   17.5.5 SOPC component creation 454
17.6 Video driver 454
   17.6.1 Video memory access routines 454
   17.6.2 Geometrical model routine 456
   17.6.3 Bitmap processing routines 457
17.6.4 Bit-mapped text routines 460
17.7 Mouse processing routines 463
17.8 Testing program 464
  17.8.1 Chart plotting routine 465
  17.8.2 General plotting functions 467
  17.8.3 Strip swapping routine 469
  17.8.4 Mouse demonstration routine 469
  17.8.5 Bit-mapped text routine 470
17.9 Bitmap file processing 471
  17.9.1 BMP format overview 471
  17.9.2 Generation of BMP file 472
  17.9.3 Sprite-based design 472
  17.9.4 BMP file access 473
  17.9.5 Host-based file system 474
  17.9.6 Bitmap file retrieval routines 476
17.10 Bibliographic notes 479
17.11 Suggested experiments 480
  17.11.1 PLL-based VGA controller 480
  17.11.2 VGA controller with 16-bit memory configuration 480
  17.11.3 VGA controller with 3-bit color depth 480
  17.11.4 VGA controller with 1-bit color depth 480
  17.11.5 VGA controller with double buffering 480
  17.11.6 VGA controller with 320-by-240 resolution 480
  17.11.7 VGA controller with vertical mode operation 481
  17.11.8 Geometrical model functions 481
  17.11.9 Bitmap manipulation functions 481
  17.11.10 Simulated “Etch A Sketch” toy 481
  17.11.11 Palette lookup table circuit 481
  17.11.12 Virtual LED flashing system panel 481
  17.11.13 Virtual analog wall clock 482
17.12 Suggested projects 482
  17.12.1 Configurable VGA controller 482
  17.12.2 VGA controller using system SDRAM 482
  17.12.3 Paint program 482
  17.12.4 Video game 483
17.13 Complete program listing 484

18 Audio Codec Controller 511
18.1 Introduction 511
  18.1.1 Overview of codec 511
  18.1.2 Overview of WM8731 device 512
  18.1.3 Registers of WM8731 device 513
CONTENTS

18.2 I2C controller 516
  18.2.1 Overview of I2C interface 516
  18.2.2 HDL implementation 518
18.3 Codec data access controller 524
  18.3.1 Overview of digital audio interface 524
  18.3.2 HDL implementation 525
18.4 Audio codec controller IP core development 527
  18.4.1 Complete audio codec controller 527
  18.4.2 Avalon interfaces 529
  18.4.3 Register map 530
  18.4.4 Wrapped audio codec controller 531
  18.4.5 SOPC component creation 533
18.5 Codec driver 533
  18.5.1 I2C command routines 533
  18.5.2 Data source select routine 534
  18.5.3 Device initialization routine 534
  18.5.4 Audio data access routines 535
18.6 Testing program 536
18.7 Audio file processing 539
  18.7.1 WAV format overview 539
  18.7.2 Audio format conversion program 540
  18.7.3 Audio data retrieval routine 541
18.8 Bibliographic notes 543
18.9 Suggested experiments 543
  18.9.1 Software I2C controller 543
  18.9.2 Hardware data access controller using master clocking mode 543
  18.9.3 Software data access controller using slave clocking mode 543
  18.9.4 Software data access controller using master clocking mode 543
  18.9.5 Configurable data access controller 544
  18.9.6 Voice recorder 544
  18.9.7 Real-time sinusoidal wave generator 544
  18.9.8 Real-time audio wave display 544
  18.9.9 Echo effect 544
18.10 Suggested projects 545
  18.10.1 Full-fledged I2C controller 545
  18.10.2 Digital equalizer 545
  18.10.3 Digital audio oscilloscope 545
18.11 Complete program listing 546

19 SD Card Controller 557
  19.1 Overview of SD card 557
  19.2 SPI controller 558
21.6.4 Text panel display routines 653
21.6.5 Mouse processing routine 654
21.6.6 Main program 656
21.7 Discussion 656
21.8 Bibliographic notes 657
21.9 Suggested experiments 657
21.9.1 Hardware accelerator with one multiplier 657
21.9.2 Hardware accelerator with modified escape condition 658
21.9.3 Hardware accelerator with Q4.12 format 658
21.9.4 Hardware accelerator with multiple fractal engines 658
21.9.5 “Burning-ship” fractal 658
21.9.6 Enhanced testing program 658
21.10 Suggested projects 659
21.10.1 Floating-point hardware accelerator 659
21.10.2 General fractal drawing platform 659
21.11 Complete program listing 660

22 Direct Digital Frequency Synthesis 671
22.1 Introduction 671
22.2 Design and implementation 671
22.2.1 Direct synthesis of a digital waveform 672
22.2.2 Direct synthesis of an unmodulated analog waveform 673
22.2.3 Direct synthesis of a modulated analog waveform 674
22.2.4 HDL implementation 674
22.3 DDFS IP core development 677
22.3.1 Avalon interface 677
22.3.2 Register map 678
22.3.3 Wrapped DDFS circuit 678
22.3.4 Codec DAC integration 679
22.4 DDFS driver 680
22.4.1 Configuration routines 680
22.4.2 Initialization routine 681
22.5 Testing 681
22.5.1 Overview of music notes and synthesis 682
22.5.2 Testing program 683
22.6 Bibliographic notes 687
22.7 Suggested experiments 687
22.7.1 Quadrature phase carrier generation 687
22.7.2 Reduced-size phase-to-amplitude lookup table 687
22.7.3 Synthetic music player 687
22.7.4 Keyboard piano 688
22.7.5 Keyboard recorder 688
xxiv CONTENTS

22.7.6 Hardware envelope generator 688
22.7.7 Additive harmonic synthesis 688
22.7.8 Sample-based synthesis 688
22.8 Suggested projects 688
  22.8.1 Sound generator 688
  22.8.2 Function generator 689
  22.8.3 Full-fledged electric synthesizer 689
22.9 Complete program listing 690

References 697

Topic Index 701